ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

2(186) 2012

ТЕОРЕТИЧЕСКИЙ И ПРИКЛАДНОЙ НАУЧНО-ТЕХНИЧЕСКИЙ ЖУРНАЛ

Издается с ноября 1995 г.

УЧРЕДИТЕЛЬ
Издательство "Новые технологии"

СОДЕРЖАНИЕ

ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ И СЕТИ

Алекперов Р. К. Создание распределенных вычислительных сред на основе технологии вычислительных облаков 2
Салибекян С. М., Панфилов П. Б. Объектно-атрибутная архитектура — новый подход к созданию объектных систем 8
Докучаев А. Н. Особенности диспетчеризации сверхлегких задач в мультипроцессорных вычислительных системах реального времени 14

ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМЫ

Норенков И. П., Уваров М. Ю. Задачи обработки знаний на основе ролевой кластеризации онтологий 19
Болховитянов А. В., Чеповский А. М. Методика автоматического выделения структурных единиц в предложениях на русском языке 25
Левков А. А. Интеграция знаний и данных в больших информационных системах 29
Бочков М. В., Бойков П. Н. Способ автоматического рубрицирования неструктурированной информации сети Интернет 34

МОДЕЛИРОВАНИЕ И ОПТИМИЗАЦИЯ

Меркушева А. В., Малыхина Г. Ф., Малыхин В. М. Структуры, методы и алгоритмы адаптивной фильтрации нестационарных сигналов 37
Воловиков В. В., Сарафанов А. В. Моделирование теплового сопротивления при вынужденном конвективном теплообмене в каналах конструкций радиоэлектронной аппаратуры 44

ГЕОИНФОРМАЦИОННЫЕ СИСТЕМЫ

Бобков В. А., Мельман С. В., Май В. П. Визуализация динамики циклонов . . 49

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ В ЭКОНОМИКЕ И УПРАВЛЕНИИ

Мезенцев Ю. А., Павлов П. С. К программной реализации декомпозиционного алгоритма решения одного класса задач дискретной оптимизации с полуопределенной релаксацией 54
Топка В. В. Управление стоимостью проекта с учетом показателя его надежности . 60

ПРИКЛАДНЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ

Боровский А. С., Тарасов А. Д. Метод обработки экспертной информации на основе нечетких гиперграфов для проектирования систем физической защиты . . . 67
Набибекова Г. Ч. Применение OLAP-технологий в системах поддержки принятия решений в сфере внешней политики 73
Барбасова Т.А., Вставская Е. В., Константинов В. И. Кодирование информации методом пропуска полупериодов сетевого напряжения в системах освещения . 76
Contents 78

Приложение. Галушкин А. И. Стратегия развития современных суперкомпьютеров на пути к экзафлопным вычислениям

Информация о журнале доступна по сети Internet по адресу **http://novtex.ru/IT.**
Журнал включен в систему Российского индекса научного цитирования.
Журнал входит в Перечень научных журналов, в которых по рекомендации ВАК РФ должны быть опубликованы научные результаты диссертаций на соискание ученой степени доктора и кандидата наук.

УДК 004.324

Р. К. Алекперов, канд. техн. наук, зав. отделом,
Институт информационных технологий НАНА,
г. Баку,
e-mail: rashid@iit.ab.az

Создание распределенных вычислительных сред на основе технологии вычислительных облаков

Рассмотрены принципы создания распределенных вычислительных сред на основе технологии вычислительных облаков. Проведенный анализ показывает, что вычислительные облака обладают следующими преимуществами: увеличение доступных вычислительных мощностей, обеспечение пользователей неограниченным дисковым пространством, высокоскоростная обработка данных, оплачивание фактического использования компьютерных ресурсов и т. д.

Ключевые слова: *распределенное вычисление, вычислительные облака, вычислительная система, услуги вычислительных облаков, вычислительная среда, распределение нагрузки*

Введение

В настоящее время вычислительных мощностей персональных компьютеров бывает недостаточно для решения сложных задач, требующих мощных вычислительных ресурсов и больших объемов памяти и относящихся к разным областям науки, таким как физико-химические процессы и ядерные реакции, моделирование развития экономики, криптография, геология, создание новых лекарственных препаратов и др. Для выполнения сложных вычислений и быстрой обработки и передачи информации при решении указанных задач пользователи используют суперкомпьютеры, имеющие высокую вычислительную производительность и большие объемы памяти [1]. Большинство государств и организаций не имеет возможностей приобрести такие дорогостоящие суперкомпьютеры, однако потребность в вычислительных ресурсах у этих государств и организаций велика. Быстрое развитие информационных и сетевых технологий в последнее время привело к увеличению вычислительной мощности компьютерных сетей (КС). Таким образом, возникла идея создать распределенные вычислительные среды (РВС) из общедоступных компьютеров на базе сетей для решения сложных задач. Использование КС для решения этой проблемы уже сегодня — дело вполне реальное. Такая технология рассматривается мировым сообществом как наиболее перспективная для проведения распределенных вычислений, использующих географически рассредоточенные ресурсы [2].

Использование в ежедневной практической деятельности удаленного доступа к вычислительным системам с применением высокоскоростных каналов связи открывает новые возможности для пользователей. Такой рост возможностей получения пользователями вычислительных ресурсов и памяти привел к качественному изменению принципов организации распределенной вычислительной среды.

Модели облачных вычислений

В настоящее время в мире проводятся интенсивные исследовательские работы по созданию системы вычислительных облаков (*Cloud Computing*), с помощью которых решение задач обходится намного дешевле по сравнению с суперкомпьютерами. Такие системы, осуществляющие большие вычисления, создаются на основе компьютерных сетей, имеющих высокоскоростные каналы связи.

Cloud Computing рассматриваются как приложения, поставляемые в качестве услуги через сеть Интернет, и аппаратные и программные системы в центрах обработки данных, которые предоставляют эти услуги. Облачные вычисления являются относительно новой концепцией, она стала популярной в последнее время. Облако использует технологию виртуализации, и в сущности облачных вычислений есть логическое разделение между различными узлами, где каждый узел выступает как отдельные физические машины пользователя. В отличие от распределенных вычислений, оно подключает вместе несколько распределенных компьютеров, чтобы сформировать большой логический компьютер, который может обрабатывать большие объемы данных и вычислений. В случае облачных вычислений технология виртуализации позволяет каждый узел отображать как отдельный физический компьютер, через который пользователь может загрузить специальное программное обеспечение и операционную систему на каждом узле и настроить пользовательские правила для каждого узла [2, 3].

Идея облачных вычислений следует из параллельной обработки распределенных вычислений и Grid-компьютинга. При том, что существует сходство между ними, работают они по-разному. Хотя облачные вычисления являются развивающейся областью информатики, идея реализовалась в течение нескольких лет. Для ее обозначения используется термин "облачные вычисления", поскольку данные и приложения существуют на "облаке" веб-серверов. Концепция облачных вычислений может быть определена как обмен и использование приложений и ресурсов сетевого окружения, при этом пользователей не интересует их принадлежность [4].

Отличительные характеристики вычислительных облаков следующие:

- *доступность* — всем, из любой точки, где есть сеть Интернет, с любого компьютера, где есть браузер;
- *дешевизна* — плати столько, сколько используешь, позволь себе дорогие, мощные компьютеры и программы;
- *простота* — не требуются покупка и настройка программ и оборудования, их обновление;
- *гибкость* — неограниченность вычислительных ресурсов (память, процессор, диски);
- *широкий выбор* — программы и сервисы без установки на локальный компьютер, компьютеры любой конфигурации удаленно.

Были определены три модели облачных вычислений: открытая, частная и гибридная (рис. 1) [5].

Открытые (общественные) облака — открыты для использования широкой общественностью. Они находятся за пределами брандмауэра и полностью управляются поставщиком. Пользователями могут быть частные лица, корпорации и т. д. Операции и функции предоставляются как услуга через сеть Интернет, что обеспечивает доступ к соответствующим технологическим сервисам для пользователей, не обладающих знаниями, опытом использования и администрирования технологической инфраструктуры, лежащей в основе таких сервисов. Такие облака также называются *"внешними"*.

Основные преимущества использования общественных облаков:

- легкая и недорогая установка, потому что аппаратные средства, приложения, пропускная способность, расходы покрываются за счет поставщика;
- масштабируемость для удовлетворения потребностей;
- плата за фактическую услугу (*pay as you go*).

Амазонки Web Services и Google App Engine являются хорошими примерами общественных облачных вычислений.

Рис. 1. Модель системы облачных вычислений

Частные облака — облако размещено в пределах организации. Частные облака имеют много преимуществ перед общественными облаками, например, данные и процессы управляются в рамках организации, нет никаких дополнительных правил безопасности, юридических требований или ограничений пропускной способности, которые могут присутствовать в общественных облаках. Кроме того, с помощью частных облаков поставщики и пользователи имеют контроль над инфраструктурой и повышением уровня безопасности, поскольку доступ пользователей в используемые сети ограничен. Данное облако создается организацией для собственных сотрудников, а операции/функции предоставляются по защищенной брандмауэром внутренней сети предприятия, но не по сети Интернет. Владелец частного облака не обеспечивает совместного доступа к ресурсам других компаний, поэтому не возникает проблем, связанных с наличием нескольких владельцев. Такие облака также называются *"внутренними"*. Частные облака имеют следующие недостатки: по сравнению с общественными облаками развитие частных облаков требует вложений в оборудование и найма собственных специалистов.

Гибридные облака — сочетание открытого и частного облаков. Пользователи модели обычно держат критически важные бизнес-данные и услуги под контролем, происходит аутсорсинг менее критической обработки и информации для общественности поставщиком облака. Организация устанавливает правила и политику на основании таких факторов, как безопасность, критичность и базовая инфраструктура, с тем чтобы действия и задачи распределялись соответственно по внешним или внутренним облакам.

Технология Cloud Computing дает пользователям доступ к мощным вычислительным ресурсам и хранилищам данных, при этом местонахождение и настройки этих ресурсов для пользователя не имеют значения.

Услуги, преимущества и недостатки вычислительных облаков

Система Cloud Computing включает 11 категорий услуг. Принято классифицировать основные типы сервисов Cloud Computing следующим образом: IaaS, PaaS, SaaS [6, 7].

Инфраструктура как сервис (IaaS) — это предоставление компьютерной инфраструктуры (как правило, в форме виртуализации) как услуги на

основе концепции облачных вычислений. IaaS состоит из двух основных компонентов:

- аппаратные средства (серверы, системы хранения данных, клиентские системы, сетевое оборудование);
- операционные системы и системное программное обеспечение (средства виртуализации, автоматизации, основные средства управления ресурсами).

IaaS основана на технологии виртуализации, позволяющей пользователю оборудования делить его на части, которые соответствуют текущим потребностям бизнеса, увеличивая тем самым эффективность использования имеющихся вычислительных мощностей. То есть пользователь должен будет оплачивать всего лишь реально необходимые ему для работы серверное время, дисковое пространство, сетевую пропускную способность и другие ресурсы. Кроме того, IaaS предоставляет в распоряжение клиента весь набор функций управления на одной интегрированной платформе. Таким образом, IaaS избавляет предприятия от необходимости поддержки сложных инфраструктур центров обработки данных, клиентских и сетевых инфраструктур, а также позволяет уменьшить связанные с этим капитальные затраты и текущие расходы. Можно также получить дополнительную экономию при предоставлении услуг в рамках инфраструктуры совместного использования. Таким образом, для решения задач на этом уровне создается компьютерная инфраструктура. К услугам IaaS можно отнести Amazon S3 (Simple Storage Service), Amazon Elastic Computer Cloud (EC2), IBM Blue Cloud. Для использования услуг этого сервиса пользователь загружает на свой компьютер соответствующий веб-браузер и для решения задачи обращается к облакам.

Платформа как сервис (PaaS) — это предоставление интегрированной платформы для разработки, тестирования, развертывания и поддержки веб-приложений как услуги. Для разворачивания веб-приложений клиенту не нужно приобретать оборудование и программное обеспечение, нет необходимости организовывать их поддержку. Доступ для клиента может быть организован на условиях аренды. Такой подход имеет следующие достоинства:

- масштабируемость;
- отказоустойчивость;
- виртуализация;
- безопасность.

Масштабируемость PaaS предполагает автоматическое выделение и освобождение необходимых ресурсов в зависимости от числа обслуживаемых приложением пользователей. PaaS как интегрированная платформа для разработки, тестирования, разворачивания и поддержки веб-приложений позволит весь перечень операций по разработке, тестированию и разворачиванию веб-приложений выполнять в одной интегрированной среде, исключая тем самым затраты на поддержку отдельных сред для отдельных этапов. Сервис PaaS — виртуальная платформа, дающая возможность пользователям использовать операционные системы и приложения специализированных программ (Apache, My SQL и т. д.), размещенных в виртуальных серверах (состоящих из физических серверов). Примерами сервиса PaaS являются IBM IT Factory, Google App Engine, Force.com.

Программное обеспечение как сервис (SaaS) — модель продажи программного обеспечения, при которой поставщик разрабатывает веб-приложение и самостоятельно управляет им, предоставляя заказчикам доступ к программному обеспечению через сеть Интернет. В данном случае основное преимущество модели SaaS для клиента состоит в отсутствии затрат, связанных с установкой, обновлением и поддержкой работоспособности оборудования и программного обеспечения, работающего на нем. Сервис SaaS характеризуется следующим:

- приложение приспособлено для удаленного использования;
- одним приложением могут пользоваться несколько клиентов;
- оплата за услугу взимается либо как ежемесячная абонентская плата, либо на основе суммарного объема транзакций;
- поддержка приложения входит уже в состав оплаты;
- модернизация приложения может проводиться обслуживающим персоналом плавно и прозрачно для клиентов.

С точки зрения разработчиков программного обеспечения сервис SaaS позволяет эффективно бороться с нелицензионным использованием программного обеспечения благодаря тому, что клиент не может хранить, копировать и устанавливать программное обеспечение. По сути, программное обеспечение в рамках SaaS можно рассматривать в качестве более удобной и выгодной альтернативы внутренним информационным системам. К программным сервисам, используемым на этом уровне, можно отнести Google Apps, Google Docs, Microsoft Software Services (E-mail, video-conference), Salesforce.com (CRM — система управления взаимосвязями клиентов, ERP — система управления ресурсами предприятия) и т. д.

Схема обработки запроса пользователя в системе Cloud Computing показана на рис. 2 [8].

Можно отметить три основных фактора технологии Cloud Computing, привлекающих пользователей [3]:

- неограниченное количество вычислительных ресурсов, т. е. освобождение пользователей от прогнозирования количества нужных ресурсов и их предварительного заказа;
- отсутствие больших расходов на первых этапах проектов;

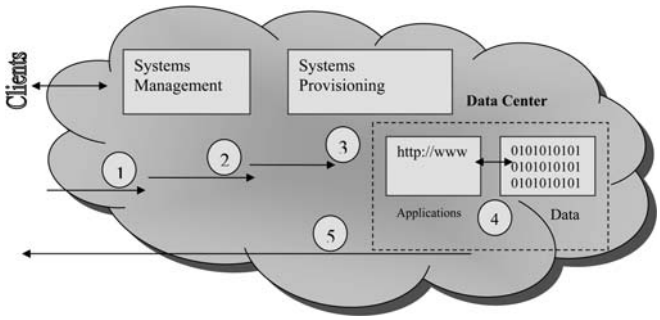


Рис. 2. Блок-схема обработки запроса пользователя в системе Cloud Computing:
1 — клиент передает запрос на доступ к услуге; *2* — система управления находит нужные ресурсы; *3* — система выделения ресурсов находит нужные ресурсы; *4* — после выделения необходимых вычислительных ресурсов обрабатывается запрос на доступ к услуге; *5* — результаты передаются клиентам

- плата за фактическую услугу (*pay as you go*).

Проводимые исследования показали, что у вычислительных облаков есть как преимущества, так и недостатки [2].

Преимущества:

- уменьшение требований к вычислительным ресурсам и ресурсам памяти персональных компьютеров, подключенных к сети Интернет;
- обеспечение пользователей неограниченными вычислительными ресурсами и ресурсами памяти;
- плата за фактическое использование вычислительных ресурсов и ресурсов памяти;
- высокоскоростная обработка данных;
- уменьшение расходов на аппаратное и программное обеспечение (ПО), услуги и электроэнергию;
- обеспечение безопасности хранения данных;
- эффективное использование устройства дисковой памяти (данные и программы хранятся в вычислительных облаках);
- постоянное обновление пользовательских программ;
- совместное редактирование файлов и табличных документов пользователями, объединившимися в группу.

Недостатки:

- зависимость хранения данных пользователя от компаний, оказывающих Cloud Computing услуги;
- создание новых монополистов (облаков);
- наличие вопросов надежности, безопасности каналов связи;
- в этой области не разработаны методы и стандарты, гарантирующие качественные услуги;
- компьютер пользователя должен быть постоянно подключен к сети Интернет;
- скорость канала связи должна быть высокой;
- в случае исчезновения данных пользователей восстановление их невозможно;
- на выполнение некоторых программ может быть затрачено больше времени, чем на выполнение этих программ на локальных компьютерах.

Организация услуги на базе технологий вычислительных облаков

Проводимые исследования показывают, что 60...80 % вычислительных ресурсов и памяти компьютеров таких больших компаний, как Intel, IBM, Google и т. д., используются достаточно эффективно. Но технология Cloud Computing дает возможность вычислительные ресурсы компьютеров компаний использовать еще более эффективно.

В настоящее время многие крупные компании — Microsoft, Google, IBM, Oracle/Sun, Amazon и более мелкие фирмы, конкурируя друг с другом, заняты разработкой своих облачных сервисов и инструментов для их создания. Имеется тенденция к интеграции "корпоративных облаков" в единое доступное пользователю облако. К компаниям, оказывающим услуги в сети Cloud Computing, можно отнести, например, Google, Amazon, IBM, Microsoft [9].

Компания Google в 2008 г. для создания вебприложений и выполнения хостинговых услуг создала платформу Google App Engine на основании технологии Cloud Computing. В настоящее время предложенная компанией Google онлайновая почтовая услуга Gmail основана на технологии Cloud Computing. Размещение в вычислительных облаках прокси-компьютеров, осуществляющих электронные почтовые услуги, даст возможность для проверки точности использующейся информации и документов, предотвратит потерю данных, обеспечит защиту от постороннего вмешательства, осуществит поиск уязвимостей системы.

Компания Amazon, начиная с 2006 г., создав платформу Amazon Web Services (AWS), предложила Интернет-пользователям такие услуги, как Amazon Simple Storage Service (S3) — хранение данных в памяти серверов, и Amazon Elastic Compute Cloud (EC2) — использование вычислительных ресурсов на многочисленных масштабируемых серверах. Этот сервис выделяет пользователю виртуальный персональный сервер (ECR Compute Units — EC2 CU), производительность которого равна производительности процессора AMD Opteron или Intel Xeon (32-го или 64-го разряда). Кроме того, для выполнения вычислений с помощью этой услуги пользователям предлагаются восемь видов виртуальных серверов (характеристики некоторых из них приведены ниже):

- Small Instance (один EC2 CU 32-го разряда, оперативная память 1,7 Гбайт, дисковая память 160 Гбайт);
- Large Instance (два EC2 CU 64-го разряда, оперативная память 7,5 Гбайт, дисковая память 850 Гбайт);

- Extra Large Instance (четыре EC2 CU 64-го разряда, оперативная память 15 Гбайт, дисковая память 1,69 Тбайт);
- High-Memory Quadruple Extra Large Instance (двадцать шесть EC2 CU 64-го разряда, оперативная память 68,4 Гбайт, дисковая память 1,69 Тбайт).

Клиент может арендовать виртуальную машину на одной из 12 предлагаемых операционных систем. Также ему предлагаются восемь различных конфигураций, самая мощная из которых, High-Memory Quadruple Extra Large Instance, состоит из 26 стандартных компьютерных модулей ECU, каждый из которых предоставляет расчетные параметры, эквивалентные процессору 2007 Xeon с частотой 1...1,2 ГГц. Пользоваться сервисом можно одноразово, если, например, потребуется протестировать работу базы данных на компьютерах разной мощности и понять, какую конфигурацию выбрать в итоге. Стоимость аренды виртуальных машин варьируется в диапазоне $0,085...2,00 в час для UNIX-систем и обойдется в $0,12...2,48 для работы под Windows. Мощности можно зарезервировать сроком на один год. В этом случае единовременно выплачивается сумма от $227 до $5300, виртуальные машины оказываются в вашем постоянном распоряжении. Если виртуальные машины никак не задействованы, плата не начисляется. Услуга Amazon S3 дает возможность пользователю или организации получить 50...500 Тбайт памяти. Месячная цена 1 Гбайт памяти приблизительно составляет $0,15. Кроме того, оплата осуществляется за фактически использованные ресурсы. По расчетам журнала Forbes, эти услуги принесли фирме большие доходы.

Компании IBM и Microsoft тратят миллионы долларов на создание системы Cloud Computing. Фирма IBM планирует создать систему "вычислительные облака", состоящую из 13 центров данных, расположенных на территориях различных стран (США, Китай, Япония, Франция, Турция и т. д.), которая обойдется ей в $300 млн. Это даст возможность в случае выхода из строя компьютеров организаций, пользующихся услугами этой системы, восстановить информацию, хранящуюся в компьютерах, за короткое время (в течение 2...6 ч) [10].

Компания Microsoft намерена запустить в продажу операционную систему Windows Azure, работающую на основе технологии вычислительных облаков. Новая платформа будет работать на основе группы серверов, координируемых посредством сети Интернет, и будет предлагать пользователям услуги по использованию вычислительных ресурсов и ресурсов памяти. Windows Azure — операционная система и набор инструментов фирмы Microsoft, обеспечивающий поддержку облачных вычислений ("ОС в облаке"). Важно подчеркнуть, что Windows Azure обеспечивает хранение, использование и модификацию данных и запуск программ только на компьютерах центров обработки данных Microsoft. Никакого программного обеспечения, кроме веб-браузера, на пользовательских компьютерах не требуется.

С точки зрения пользователя существуют две категории приложений — внутренние (*on-premises applications*), исполняемые на компьютере пользователя, и облачные (*cloud applications*), фактически исполняемые в среде Windows Azure на компьютерах центра обработки данных. На пользовательском компьютере могут быть установлены операционные системы Windows и, возможно, другие. Независимо от этого через веб-браузер пользователь получает доступ к Windows "в облаке" — Windows Azure. Функционирование Windows Azure основано на веб-сервисах .NET. Windows Azure для хранения данных обеспечивает доступ к аналогу СУБД Microsoft SQL Server "в облаке" — SQL Azure. Основные компоненты Windows Azure — вычисления (Compute), память (Storage) и интерфейс (fabric). Все компоненты — вычисления, память и интерфейс — являются веб-сервисами .NET. Сервис вычисления выполняет пользовательские облачные приложения, сервис памяти хранит пользовательские данные, сервис интерфейса обеспечивает общие средства управления приложениями, использующими облачную платформу.

Компании SAP и Oracle предлагают услугу централизованного использования программных приложений, основанную на сервисной платформе "как услуга ПО" (SaaS) технологии Cloud Computing.

Предложенные фирмой Intel вычислительные облака предлагают Интернет-пользователям и организациям вычислительные ресурсы, необходимые для решения сложных задач. Услугами вычислительных облаков, созданных компанией Intel, широко пользуются такие известные зарубежные фирмы, как Facebook, Tencent, Baidu и Oracle.

Системы Cloud Computing, в отличие от существующих систем, являются более гибкими. Например, если увеличится число запросов к веб-ресурсам компаний, пользующихся услугами Cloud Computing, и, соответственно, объем трафика, то имеется возможность арендовать большее число серверов, и наоборот, если число запросов уменьшится, например ночью, то можно уменьшить число арендованных серверов.

По прогнозам экспертов компании Gartner group, 80 % компаний, входящих в список Fortuna 1000, в 2012 г. будут пользоваться услугами Cloud Computing. Технология Cloud Computing является очень привлекательной для новых учреждений и компаний малого бизнеса, которые не могут найти необходимые инвестиции для создания своих ИТ-инфраструктур.

Многие эксперты предполагают, что технология Cloud Computing в ближайшие пять лет заново сформирует ИТ-инфраструктуру. С помощью этой

технологии пользователи через сеть Интернет будут широко использовать услуги вычислительных облаков (вычислительные ресурсы, ПО, данные и т. д.) различного компьютерного оборудования (персональный компьютер, ноутбук, смартфоны, коммуникаторы и т. д.).

Высокая подготовленность вычислительных облаков и охватывание ими большой территории обеспечивают им преимущество при использовании. По прогнозам IDS (International Data Corporation — аналитический центр, исследующий рынок информационных технологий, Фремингем, Массачусетс, США), расходы, затраченные на создание вычислительных облаков, с $16 млрд в 2008 г. дойдут до $42 млрд в 2012 г. В 2012 г. 8,5 % годовых расходов в области ИТ придется на долю вычислительных облаков [11].

В результате развития технологии вычислительных облаков обработка и сохранение в памяти данных на серверах предприятий и организаций с экономической точки зрения станут невыгодными. Кооперативные пользователи по установленным ценам смогут взять в аренду у провайдеров вычислительные облака.

При создании систем Cloud Computing надо также обратить внимание на политические аспекты. Так, расположение многих известных компаний, оказывающих хостинговые услуги, на территории США упрощает контроль за информацией тех пользователей, которые пользуются услугами этих компаний. Многие эксперты считают, что пользование услугами Cloud Computing приведет к потере контроля над информацией, передаваемой по сети. А это означает ослабление военных и разведывательных возможностей США

Многие государства уже понимают, что зависимость их трафика от управления зарубежными странами создает проблемы в вопросах обеспечения национальной безопасности. Во избежание этого специалисты либо предлагают установить системы Cloud Computing, управляемые в согласованном порядке несколькими государствами, либо создать системы Cloud Computing, находящиеся под контролем каждой страны. В настоящее время первое предложение используется шире, потому что это обходится гораздо дешевле.

Хранимой в облаках информацией можно пользоваться в любой точке мира. Но это может противоречить законам некоторых стран о защите конфиденциальности данных. Например, на основании законодательства стран Европейского союза (ЕС) некоторые виды профессиональных данных не могут передаваться за пределы ЕС. В связи с этим компанией Amazon и другими были разработаны требования для пользователей, имеющих возможность использовать сервис хранения данных в странах ЕС.

Таким образом, компании и отдельные пользователи, не покупая мощные и дорогие компьютеры, серверы и ПО, пользуясь услугами вычислительных облаков и взяв в аренду вычислительные ресурсы и ресурсы памяти по низким ценам, смогут решить необходимые им задачи.

Заключение

Проведенный анализ показывает, что при создании РВС на основе технологии вычислительных облаков приходится учитывать множество факторов: модели, надежность, безопасность, производительность компьютеров, услуги вычислительных облаков, скорость передачи каналов связи, интеграцию вычислительных ресурсов компьютеров разных организаций. Исследования показали, что создание распределенных вычислительных систем на базе технологий вычислительных облаков имеет следующие преимущества: уменьшение требований к вычислительным ресурсам и ресурсам памяти персональных компьютеров, доступ пользователей к неограниченным вычислительным ресурсам и т. д. При этом главная цель заключается в эффективном использовании простаивающих вычислительных ресурсов таких компаний, как Google, Amazon, IBM и т. д., предоставляющих клиентам вычислительные мощности и дисковое пространство.

Список литературы

1. **Воеводин В. В., Воеводин Вл. В.** Параллельные вычисления. СПб.: БХВ — Петербург, 2002. 608 с.

2. **Алгулиев Р. М., Алекперов Р. К.** Вычислительные облака: современное состояние, проблемы и перспективы // Телекоммуникации. 2010. № 9. С. 15—25.

3. **Armbrust M., Fox A., Griffith R., Josep A.** Above the Clouds: A Berkeley View of Cloud Computing. URL: http://www.eecs.berkeley.edu

4. **Scale M. S.** Cloud computing and collaboration. URL: http://www.emeraldinsight.com

5. **Rimal B. P., Choi E.** A Conceptual Approach for Taxonomical Spectrum of Cloud Computing // Proceedings of the 4th International Conference on Ubiquitous Information Technologies & Applications. ICUT '09. Fukuoka, Japan. 2009.

6. **Джонс Т.** Cloud Computing и Linux (платформы и приложения для Cloud Computing). URL: http://www.ibm.com

7. **Dikaiakos M., Pallis G., Katsaros D., Mehra P., Vakali A.** Cloud Computing — Distributed Internet Computing for IT and Scientific Research // IEEE Internet Computing. 2009. N 9. P. 10—13.

8. **Ференц В.** Так что же такое Cloud Computing. URL: http://www.cio-world.ru

9. **Циферов И.** Cloud Computing: что там за "облаками". URL: http://www.itnews-com.ua

10. **Ильин Ю.** IBM инвестирует "вычислительные облака". URL: http://www.pcnews.ru

11. **Левит А.** Готовы ли вычислительные облака к выходу в массы? URL: http://www.osp.ru

УДК 004.438, 004.08, 004.274

С. М. Салибекян, ст. преподаватель,
Московский институт электроники и математики,
e-mail: salibek@yandex.ru,
П. Б. Панфилов, канд. техн. наук, доц.,
Московский институт электроники и математики,
e-mail: panfilov@miem.edu.ru

Объектно-атрибутная архитектура — новый подход к созданию объектных систем

Утверждается, что наиболее популярная на сегодняшний день объектно-ориентированная парадигма программирования (ООП) не удовлетворяет современным требованиям к вычислительной технике: повторное использование кода ограничено, принцип инкапсуляции оказался неудобным для распределенных и параллельных вычислительных систем, не обеспечивается изоморфизм программы и т. д. Предлагается другая (объектно-атрибутная) парадигма, являющаяся дальнейшим развитием ООП, преодолевающая недостатки предшественника, и обеспечивающая полноценную работу вычислительной системы в режиме управления данными (dataflow).

Ключевые слова: *парадигма программирования, объектно-атрибутная парадигма, объектно-ориентированная парадигма, изоморфизм программы и данных, инкапсуляция, параллелизм вычислений, интеллектуальные вычислительные системы, распределенная вычислительная система*

Кризис объектно-ориентированного программирования

В настоящее время наибольшую популярность приобрела объектно-ориентированная парадигма программирования (ООП), основанная на фреймовой модели [3], разработанной Марвином Минским. Она имеет массу достоинств: удобная для человека абстракция программного кода и данных, возможность повторного использования программного кода; создание и легкая модификация готовых библиотек стандартных компонентов и т. д. И еще в 90-е годы прошлого века казалось, что данная парадигма способна решить буквально все проблемы в области программирования... Однако уже в 2000-е годы эйфория эта прошла — оказалось, что ООП не оправдала возложенные на нее надежды [5]: повторное использование кода оказалось ограниченным, принцип инкапсуляции неудобен для реализации распределенных вычислительных систем, не обеспечивается легкая стыковка программных блоков, разработанных различными группами программистов; не в полном объеме обеспечивается изоморфизм программы (т. е. безболезненное радикальное изменение программы); не получается организовать полноценную динамическую модификацию объектной программы (для модификации ОО-программы, к сожалению, требуется перекомпиляция кода — динамическая модификация, т. е. без перезагрузки, в современных объектных языках невозможна); не реализуется полноценный режим управления вычислительным процессом с помощью потока данных (*dataflow*). И все больше и больше ученых и программистов-профессионалов начинают утверждать, что современное программирование сейчас находится в тупике, ибо ни одна парадигма не удовлетворяет современным требованиям к вычислительной технике.

В настоящей статье предлагается разработанная автором (Салимбекяном С. М.) новая архитектура вычислительной системы (ВС), открывающая новый подход для реализации объектных систем (объектно-атрибутная (ОА) парадигма) [1, 2]. Данный подход обладает намного бóльшими возможностями для объектного программирования и создания интеллектуальных систем, чем общепринятое в настоящее время ООП [4] и реализует принцип управления вычислительным процессом с помощью потока данных (dataflow).

Введение в ОА-парадигму программирования

Для начала дадим определения нескольким понятиям, на которых основывается ОА-архитектура ВС.

Информационная пара (ИП) (атрибутированные данные) — совокупность нагрузки (данных или ссылки на данные) и ярлыка/атрибута (уникального идентификатора), описывающего нагрузку. Указатель, хранящийся в нагрузке ИП, может ссылаться на информационные конструкции любой сложности (переменные, массивы, списки, другие ИП и т. д.): тип данных определяется по ярлыку.

Функциональное устройство (ФУ) — это виртуальное (реализованное программным способом) или реальное (реализованное аппаратно) устройство, осуществляющее обработку данных. ФУ имеет внутренние регистры (набор регистров будем именовать контекстом ФУ) и может исполнять некий набор милликоманд, описываемый алгоритмом функционирования этого устройства.

Милликоманда (мК) — это ИП, где ярлык указывает функциональному устройству, каким образом следует обрабатывать прикрепленную к нему нагрузку.

Шина данных-атрибута (ШДА) — виртуальная (или реальная) шина, по которой ФУ обмениваются между собой милликомандами (рис. 1).

Капсула — это множество информационных пар, служащих для описания определенного объекта (с помощью капсулы обеспечивается абстракция

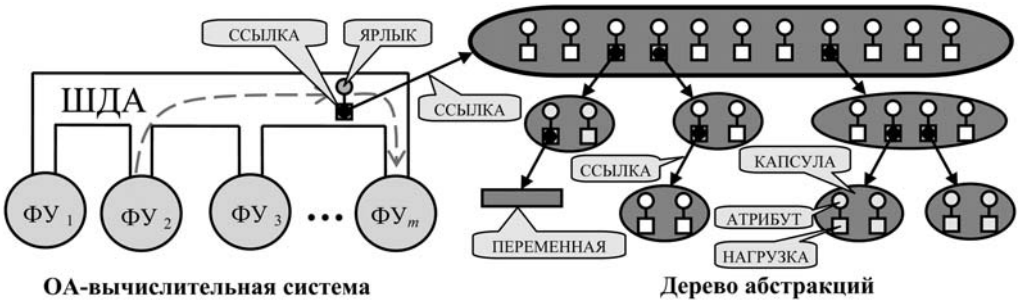


Рис. 1. Работа ВС с ОА-деревом абстракций

данных). Каждая ИП, входящая в капсулу, задает один из критериев описываемого объекта.

Например, на ОА-языке можно описать следующий объект:

Параллелепипед {*Длина*=10 *Высота*=15 *Ширина*=20},

где {} — данные знаки задают начало и конец описания капсулы; = — знак сопоставления атрибута и информационной нагрузки (перед "=" — обозначение атрибута, после — нагрузка);

Параллелепипед — имя информационной капсулы.

ОА-дерево абстракций (смысловой граф). В качестве нагрузки в ИП может выступать не только константа, но и ссылка на переменную, капсулу или иную информационную структуру, благодаря чему появляется возможность синтезировать смысловой ОА-граф, состоящий из множества капсул, связанных ссылками (рис. 1). Смысловой ОА-граф служит для описания сколь угодно сложного объекта подобно тому, как это делается в ООП. Капсула может содержать не только данные, но и программу (миллипрограмму), которая состоит из последовательности милликоманд, помещенных в капсулу.

ОА-дерево абстракций несколько напоминает собой структуру класса в ООП, однако ОА-подход имеет намного большую гибкость, так как механизм капсул и ссылок в качестве нагрузок ИП чем-то подобен детскому конструктору, из которого можно собрать сколь угодно сложную информационную структуру в реальном времени (т. е. во время выполнения вычислительного процесса). Как известно, ООП изначально строится на основе фреймовой концепции (концепция была разработана Марвином Минским в 70-е годы прошлого века [3]): фреймовая модель была лишь дополнена возможностью ссылки не только на другой фрейм, но и на процедуру или функцию (методы объекта). Совокупность фреймов, связанных между собой ссылками, хоть и дает возможность описывать любые реальные или абстрактные объекты, однако не обладает достаточной гибкостью, чтобы свободно изменять свою структуру непосредственно во время выполнения вычислительного процесса. Как правило, фреймовая структура (в ООП классы объектов) создается еще до начала выполнения программы и далее не меняется, что существенно сужает возможности ООП для создания интеллектуальных систем. В ОА-парадигме используется совершенно другой подход: информационные структуры, описывающие объекты, синтезируются не заранее, а создаются непосредственно во время работы ВС по заранее заложенным программистом (или экспертом) правилам — информационные структуры образуются путем группировки ИП в капсулы и настройки ссылок между капсулами (синтез ОА-дерева абстракций) (см. рис. 1).

В ОА-системе, работающей по принципу *dataflow*, алгоритм задается не с помощью последовательности команд, а посредством описания обмена милликомандами между ФУ (такой обмен описывается программистом с помощью ОА-языка программирования). По атрибуту милликоманды ФУ идентифицирует нагрузку (данные или ссылку), пришедшую к нему по ШДА, и, исходя из атрибута, принимает решение, каким образом ему следует ее обрабатывать (указатель в нагрузке может ссылаться на информационные объекты любой сложности (см. рис. 1)). Каждое ФУ имеет набор внутренних регистров (контекст), который используется для хранения промежуточных данных. Например, ФУ АЛУ (арифметико-логическое устройство) имеет следующий контекст: аккумулятор, флаг нулевого результата и флаг знака результата. Для выполнения операции сложения на ШДА необходимо выдать следующую последовательность милликоманд, описанных на ОА-языке:

Bus{*ALU.Set*=2, *ALU.Add*=2, *ALU.Pop*=*x*},

где *Bus* — указание на то, что последовательность милликоманд, расположенных в капсуле, должна быть выдана на *Шину* (ШДА); = — знак сопоставления ярлыка и нагрузки; *ALU* — имя ФУ; *ALU.Set, ALU.Add, ALU.Pop* — расширенные милликоманды для устройства АЛУ (в расширенной милликоманде указывается ФУ, которое должно исполнять милликоманду и атрибут нагрузки: перед точкой стоит обозначение ФУ, которому передается милликоманда, после точки — ярлык передаваемой нагрузки): *ALU.Set* — записать число из нагрузки милликоманды в аккумулятор ФУ АЛУ; *ALU.Add* — сложить значение из аккумулятора с числом из на-

грузки и записать результат вычисления обратно в аккумулятор; *ALU.Pop* — записать значение из аккумулятора в ячейку памяти с адресом, хранящимся в нагрузке милликоманды; *x* — ячейка памяти, куда АЛУ выдает результат своих вычислений.

ФУ могут быть реализованы как программным, так и аппаратным способом. При программной реализации ФУ могут создаваться и уничтожаться непосредственно во время работы ВС (создание виртуального ФУ — по сути это резервирование области в оперативной памяти компьютера для размещения там контекста ФУ). Например, при программной реализации ФУ АЛУ описание контекста оформляется с помощью структуры/записи:

```
struct AluContext{int Accumulator; //Аккумулятор
bool FZero, FLager; } // Флаги нуля и знака результата
```

а алгоритм работы АЛУ можно оформить с помощью следующей процедуры на языке *C*:

```
void ALU(void *Context, int millicomand, void *Load)
{AluContext *temp;
temp=(AluContext*)Context;
switch(millicomand)
  {case 0: // Сброс (Сброс)// в скобках мнемоника милликоманды
     ((AluContext*)Context)->Accumulator=0;
  case 1: // Установить значение в аккумуляторе (Уст)
  if(Load==NULL) ((AluContext*)Context)-> Accumulator=0;
  else ((AluContext*)Context)-> Accumulator=*PVariant(Load);
  case 2: // Сложение (Сум)
  if (Load!=NULL)
    {((AluContext*)Context)->Accumulator=
    ((AluContext*)Context)->Accumulator+*PVariant(Load);
    if(((AluContext*)Context)->Accumulator>=0)
       ((AluContext*)Context)->FLager=true;
    else ((AluContext*)Context)->FLager=false;
    if (((AluContext*)Context)->Accumulator==0)
         ((AluContext*)Context)->FZero=true;
    else ((AluContext*)Context)->FZero=false;}
  case 4: // Выдать содержимое аккумулятора (Выд)
if(Load!=NULL)*PVariant(Load)=((AluContext*)Context)->Accumulator;}
  // и т. д.....................
  }
```

где *Context* — ссылка на контекст ФУ, т. е. на структуру данных, которая содержит виртуальные регистры (таким образом, с помощью одной процедуры реализации логики работы ФУ можно управлять сразу несколькими ФУ одного типа, лишь меняя адрес в параметре *Context*);

millicomand — индекс милликоманды (по данному индексу ФУ определяет тип данных и их назначение);

Load — ссылка на нагрузку милликоманды (данная ссылка может указывать как на переменную, так и на любую информационную конструкцию).

Данный интерфейс, состоящий из трех полей, одинаков для всех типов ФУ, благодаря чему обеспечивается полная универсальность вызова процедуры реализации логики работы ФУ (в отличие от классических процедур и функций, которые имеют свой индивидуальный интерфейс — перечисление передаваемых и возвращаемых параметров). Как видно, логика работы ФУ довольно проста, так что не составит особого труда реализовывать ФУ не только в программном исполнении (виртуальные ФУ), но и в аппаратном.

Для полноценного функционирования ОА-системы необходимо ввести ФУ нескольких типов: *Шина* (специализированное ФУ, которое обеспечивает передачу ИП между различными ФУ); *АЛУ*, *ДиспетчерКапсул* (создание, удаление и модификация ИП, группировка ИП в капсулы); *УстройствоВводаВывода*, *ДиспетчерСписков* (осуществляет формирование ОА-списков и поиск по списку, исходя из заданных условий); *МатричноеАЛУ*, *Автомат* и т. д. Например, ФУ *Автомат* предназначено для того, чтобы выдавать на ШДА последовательность милликоманд, расположенных в капсуле. В состав *Автомата* входит указатель на текущую милликоманду (аналог счетчика команд в фоннеймановском компьютере), которая выдается на *Шину*. Для осуществления условных переходов в контекст *Автомата* включен регистр адреса условного перехода, где хранится адрес, на который следует переходить в случае выполнения логического условия.

На рис. 2 представлен фрагмент капсулы с милипрограммой, которая задает вывод результата вычисления ФУ АЛУ на консоль (экран): *Bus{ALU.Pop=tetmp OutConsole.OutPut}*. На первом такте считывающая головка *Автомата* (указатель на адрес текущей выдаваемой на ШДА милликоманды, который входит в контекст данного ФУ) указывает на милликоманду *ALU.Pop=temp* ("записать результат вычислений в ячейку памяти с адресом *temp*", причем *temp* — это адрес нагрузки следующей милликоманды). *OutConsole.OutPut =temp(0)*, где *temp* задает ссылку на нагрузку, (0) — указывает начальное значение, хранимое в нагрузке. На первом такте *Автомат* считывает первую милликоманду и выдает ее на *Шину*; *Шина* передает милликоманду на устройство *ALU*, а *ALU* записывает результат вычисления в ячейку памяти, адрес которой записан

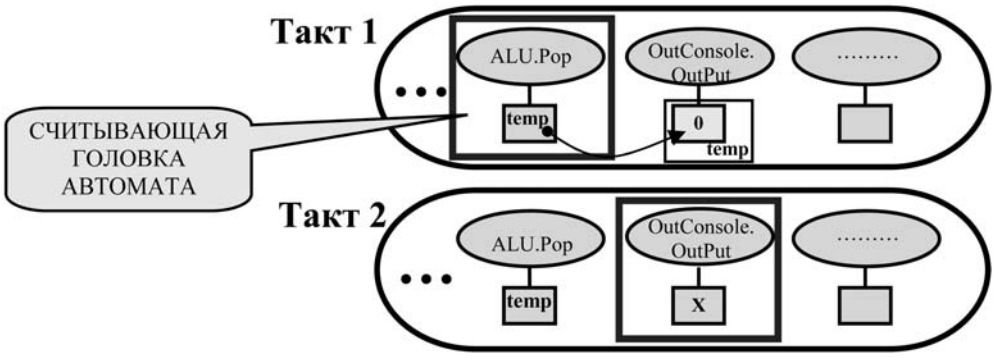


Рис. 2. Работа Автомата в ОА-системе

в нагрузке милликоманды, т. е. в *temp* (нагрузку следующей милликоманды). На втором такте работы *Автомата* указатель (считывающая головка) передвигается на следующую милликоманду, и далее вторая милликоманда выдается *Автоматом* на *Шину*; *Шина* передает ее на ФУ *OutConsole*, которое считывает из нагрузки значение, записанное в нее на предыдущем такте, и выводит его на консоль. Однако, как мы покажем дальше, в ОА-системе, обеспечивающей модель работы ВС в режиме *dataflow* и распределенное управление вычислительным процессом, ФУ *Автомат*, которое работает по командному принципу, можно и не применять.

Преимущества и недостатки ОА-архитектуры

ОА-архитектура в полном объеме обеспечивает принцип управления вычислительным процессом с помощью потока данных: в ОА-системе даже можно обойтись без привычного для классической парадигмы программирования оператора условного перехода "if". Как несложно заметить, в ОА-архитектуре функциональным устройством управляют не команды извне, а именно поток атрибутированных данных. Решение же о том, каким образом данные обрабатывать, принимает само ФУ (ФУ выполняет операцию в том случае, когда к нему приходят все необходимые для выполнения операции данные) — в классической же архитектуре команда для ФУ задается "извне", т. е. явно описывается в программе: вызов процедур и функций. К тому же, различия между данными и программным кодом стираются и благодаря тому, что в ОА-архитектуре полностью совпадают форматы атрибутированных данных и миллипрограммы: милликоманда — это по сути та же ИП, а миллипрограмма — это капсула с набором ИП. Поэтому капсулы с миллипрограммой можно без труда встраивать в ОА-граф, превращая его в мощную базу знаний. В ООП же в основном применяется управление с помощью команд (методы, входящие в состав объектов, представляют собой последовательность классических команд), что существенно ограничивает гибкость ВС, снижает ее изоморфность и не позволяет эффективно осуществлять распараллеливание вычислений.

Управление вычислительным процессом в ОА-системе децентрализованное: алгоритм задается не центральным устройством управления (например, в фон-неймановской архитектуре ВС — это ядро процессора), а описанием обмена информацией между ФУ. ФУ, получив результат выполнения команды, может либо пассивно ожидать прихода по ШДА запроса на выдачу результата, либо само через ШДА передавать результаты вычисления другим ФУ и тем самым осуществлять управление другими ФУ. Получается, что одни ФУ выдают результаты своей работы другим ФУ, те, в свою очередь, приняв все необходимые милликоманды, выполняют вычисления по заложенной в них программе, передают результаты следующим ФУ и т. д. Вычислительный процесс, таким образом, чем-то напоминает процесс ядерного деления, где высвободившиеся из ядер нейтроны попадают на другие атомы, вызывая их деление и порождая новые нейтроны, только в ОА-системе в качестве атомов выступают ФУ, а в качестве нейтронов — милликоманды, передаваемые по ШДА. Распределенность управления вычислительным процессом позволяет легко проводить распараллеливание вычислительного процесса и реализовывать распределенные ВС.

К преимуществам ОА-архитектуры можно отнести удобный способ абстракции данных и программного кода, используемый не только для описания (как в ООП), но и для распознания объектов: как уже говорилось ранее, абстракции не создаются перед выполнением программы, как в ООП, а синтезируются по заранее описанным правилам уже в процессе выполнения программы, что существенно повышает гибкость вычислительного процесса, и в результате ОА-система оказывается наделенной полноценным интеллектом: у ВС появляется возможность описания/распознания заранее неизвестных программисту объектов и самообучения (т. е. самостоятельное изменение системой правил синтеза информационных конструкций). Синтез же информационных конструкций в ОА-архитектуре осуществляется от простого к сложному. Итак,

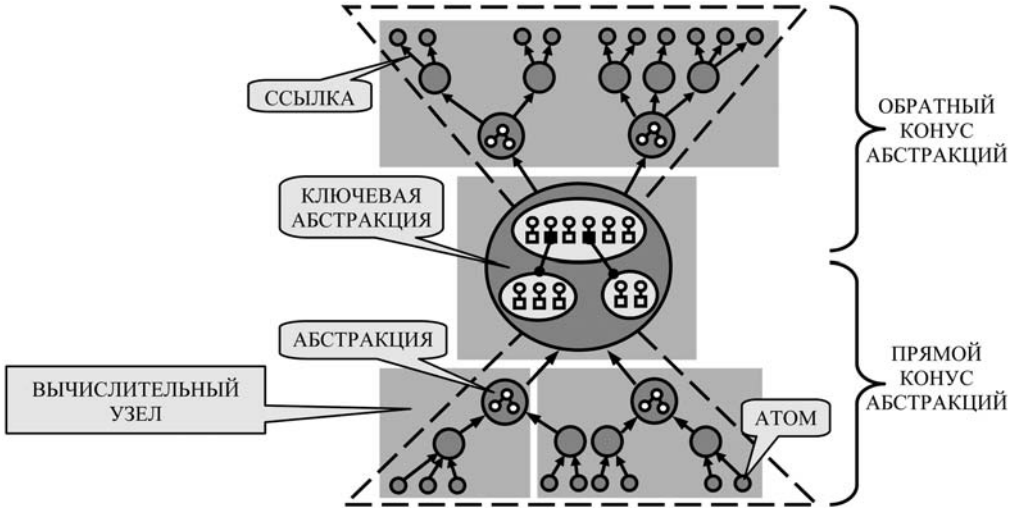


Рис. 3. Прямой и обратный конусы абстракций

ОА-система, осуществляющая распознание объектов "внешнего" мира, распадается на несколько уровней абстракции. Процесс распознания начинается с так называемых информационных атомов — элементарных (неразложимых) абстракций (атом в большинстве случаев представляет собой единственную ИП). Так, для систем автоматизации атомами будут являться сигналы, считанные с датчиков и снабженные ярлыками, идентифицирующими их. Для систем распознавания изображений атомы — описания отдельных пикселей изображения (координаты на экране и код цвета пикселя). Для систем смыслового анализа текста атомами будут являться отдельные символы, составляющие анализируемый текст (рис. 3).

Например, при смысловом анализе текста на первом уровне абстракции происходит лексический анализ, т. е. формирование из исходных символов слов. Формированием информационных конструкций для передачи на более высокий уровень абстракции занимаются ФУ, которые реализуют алгоритм работы первого уровня абстракций. Далее выделенные слова передаются на следующий уровень абстракции, где уже другие ФУ проводят синтаксический анализ, пришедших к ним лексем. Информационные конструкции, сформированные на этом уровне, передаются на уровень выше для смыслового анализа и т. д. На каждом последующем уровне информационные конструкции (абстракции) становятся все более и более сложными и их количество уменьшается (сложная абстракция представляет собой довольно большой смысловой ОА-граф). На вершине этого так называемого конуса абстракций находится ключевая абстракция, полностью описывающая смысл анализируемого ОА-системой объекта "внешнего мира".

Если же необходимо из ключевой абстракции осуществить синтез более простых информационных конструкций, то применяется так называемый обратный конус абстракций (т. е. синтез от сложного к простому). Например, если требуется перевести текст с одного языка на другой, то используются сразу и прямой (распознание смысла текста на исходном языке), и обратный (для синтеза текста уже на другом языке) конусы абстракций (рис. 3). ОА-конус абстракций весьма легко реализуется на распределенной ВС: части конуса абстракций относительно независимы друг от друга и их можно расположить на различных вычислительных узлах (рис. 3). К тому же, в отличие от распространенных ныне интерфейсов распределенных ВС Corba и DCOM, в ОА-системе нет необходимости описывать громоздкие шаблоны вызова удаленных процедур (язык IDL), так как все параметры для ФУ передаются по отдельности в виде милликоманд, что значительно облегчает процедуру обмена информацией между вычислительными узлами.

Разработанная парадигма в полной мере поддерживает принцип изоморфизма (безболезненного изменения программы), так как любую классическую процедуру можно реализовать с помощью ФУ: параметры такой процедуры передаются посредством милликоманд. ФУ для других участников информационного процесса представляет собой "черный ящик", общаться с которым можно с помощью милликоманд; модификация (дальнейшее усовершенствование) ФУ заключается в расширении набора милликоманд (для совмещения со старыми версиями программ алгоритм работы ФУ следует модернизировать таким образом, чтобы не изменять функционал старых милликоманд). Так, в классических процедурах и функциях параметры передаются единовременно, и при изменении интерфейса (перечня параметров) процедуры или функции программисту приходится исправлять всю программу, т. е. выискивать в коде все вызовы процедуры и подставлять в них новые параметры. Но благодаря милликомандному принципу, изме-

нение числа операндов и последовательность их передачи для ФУ не будут критически влиять на всю ОА-программу. Изоморфизм обеспечивается и на уровне структур данных: модификация ОА-графа, заключающаяся в добавлении в информационные капсулы новых ИП, не влияет на алгоритмы поиска по графу для других ФУ. Допустим, поиск по одной капсуле выполняют сразу два ФУ, и мы добавляем в капсулу ИП, которая является критерием поиска всего лишь для одного из них. Тогда второе ФУ просто не будет воспринимать новую ИП, так как ярлык ИП не является для него критерием поиска. И получается, что изменение структуры данных и добавление новой информации не вызывает ошибок в программе в целом. Следует отметить, что ни процедурная, ни объектно-ориентированная парадигмы изоморфизма данных не обеспечивают: любое изменение формата данных может вызвать ошибки по всей программе.

Изоморфизм, который обеспечивает ОА-парадигма, не только помогает облегчить труд программиста, но и является необходимым качеством для создания интеллектуальных систем, ведь в процессе распознания объектов и самообучения программе приходится довольно существенно изменять базу знаний, и модификации эти не должны нарушать целостности всей системы.

К недостаткам ОА-парадигмы, естественно, следует отнести избыточность данных (ведь данные необходимо снабжать ярлыками, которые требуют выделения дополнительно памяти) и сниженное по сравнению с классическим программированием быстродействие программы (ведь на анализ ярлыков требуется дополнительное машинное время). Правда, снижение быстродействия компенсируется легкостью организации параллельных вычислений в ОА-архитектуре, так как ОА-система в отличие от классической фон-неймановский ВС является параллельной по сути: ФУ работают асинхронно (их синхронизация обеспечивается через обмен информацией между ФУ по ШДА) и потому могут довольно легко функционировать в параллельном режиме.

Заключение

Работоспособность предложенной архитектуры подтверждается тем, что в настоящий момент создана ОА-среда программирования. В среде реализованы 50 классов ФУ и ОА-язык программирования (компилятор ОА-языка реализован на базе ОА-архитектуры); имеется возможность не только программирования, но и управления ОА-системой в реальном времени, т. е. коррекция алгоритма работы системы без перезагрузки ОА-программы; реализован распределенный режим работы ОА-системы (т. е. работа единой программы на ВС, состоящей из различных вычислительных узлов, объединенных между собой линиями связи). С помощью ОА-среды программирования решен ряд практических задач.

В заключение можно перечислить положительные стороны ОА-архитектуры:

- удобная структуризация (абстракция) как данных, так и программы;
- изоморфизм как на уровне данных, так и на уровне программы;
- "врожденный" параллелизм вычислений (каждое ФУ работает асинхронно и потому может функционировать параллельно с другими ФУ);
- обеспечивается распределенное управление вычислительным процессом;
- легко реализуется на распределенной ВС (программист не разрабатывает изолированный код для каждого вычислительного узла системы и не описывает обмен данными между узлами, а проектирует целостную программу, которая потом будет выполняться на всей распределенной ВС [2]);
- ОА-система может реализовываться как программно, так и аппаратно;
- при программной реализации обладает легкой переносимостью с одной аппаратной платформы на другую (для того чтобы ОА-программа заработала на новой аппаратной платформе, необходимо лишь закодировать для новой машины подпрограммы реализации логики работы ФУ, что делается довольно легко);
- имеет хорошую масштабируемость как на программном, так и на аппаратном уровнях;
- обладает удобной технологией создания программы: ОА-программа хорошо декомпозируется, части программы весьма легко стыкуются между собой, а благодаря изоморфизму гарантируется безболезненное радикальное изменение программы, что позволяет легко добавлять к уже готовой программе ранее непредусмотренные новые функции.

Список литературы

1. **Салибекян С. М.** Принципы милликомандной архитектуры как основа построения высокопроизводительных адаптивных вычислительных систем // Автоматизация и современные технологии. 2002. № 5.

2. **Салибекян С. М., Панфилов П. Б.** ОА-архитектура построения и моделирования распределенных систем автоматизации // Автоматизация в промышленности. 2010. № 11.

3. **Минский М.** Фреймы для представления знаний: пер. с англ. М.: Энергия, 1979.

4. **Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж.** Приемы объектно-ориентированного программирования. Паттерны проектирования. СПб.: Питер, 2001.

5. **Gabriel R.** Objects Have Failed: Notes for a Debate (retrieved 17 May 2009). URL: http://www.dreamsongs.com/Files/ObjectsHaveFailed.pdf

УДК 004.031.43:004.272.43:004.451.42

А. Н. Докучаев, аспирант,
БГТУ "ВОЕНМЕХ" им. Д. Ф. Устинова,
г. Санкт-Петербург,
e-mail: a.n.dokuchaev@gmail.com

Особенности диспетчеризации сверхлегких задач в мультипроцессорных вычислительных системах реального времени

Рассматриваются современные методы планирования, применяющиеся в вычислительных системах реального времени с архитектурой симметричного мультипроцессирования. Особое место уделяется вопросу постановки задачи диспетчеризации так называемых "сверхлегких задач" в условиях высокой нагрузки на процессор и недостаточного объема предоставляемого процессорного времени, а также при учете влияния эффекта Далла.

***Ключевые слова**: система реального времени, симметричное мультипроцессирование, глобальная мультипроцессорная приоритетно-управляемая диспетчеризация, эффект Далла, диспетчеризация "сверхлегких задач"*

Введение

Современный этап эволюции вычислительной техники можно характеризовать усилением внимания к мультипроцессорным и многоядерным архитектурам. В то время как темпы роста производительности отдельных вычислительных модулей в устройствах резко снизились, тенденция к увеличению их общего числа лишь набирает силу. Сегодня достаточно сложно удивить кого-либо даже 32-ядерными микропроцессорными модулями, поддерживающими архитектуру симметричного мультипроцессирования, особенно если речь идет о сфере встраиваемых систем и систем реального времени (СРВ).

В данном аспекте вопросы эффективного распределения вычислительных ресурсов процессора между потребителями (выполняющимися в системе задачами) становятся особенно актуальными. Стоит отметить, что в подавляющем большинстве случаев программное обеспечение (ПО) СРВ составляют как задачи реального времени, так и задачи общего назначения. Обязательным требованием, предъявляемым к СРВ на этапе проектирования, является гарантирование успешного выполнения каждой из них. Если для задач общего назначения данное требование зависит от субъективных представлений разработчиков и особенностей конкретной системы, то для задач реального времени оно четко формулируется: необходимо строго соблюдать все крайние сроки завершения. В СРВ за выполнение обозначенного требования отвечает такой программный компонент, как диспетчер (или планировщик), характеризующийся реализуемыми алгоритмами планирования.

В 1978 г. S. K. Dhall и C. L. Liu в работе [1] подвергли анализу систему, имеющую конфигурацию задач реального времени, при которой система не являлась диспетчеризируемой даже при стремлении к нулю суммарной нагрузки на процессоры. Данное явление объясняется назначением высоких приоритетов задачам с малым периодом появления и слабой нагрузкой на процессор при частотно-монотонной диспетчеризации, что приводит к невозможности выполнения задач, потребляющих большее количество процессорного времени, но возникающих реже. Таким образом, значительное число сравнительно быстро завершающихся задач с малыми периодами может рассматриваться в качестве препятствия для диспетчеризации остальных задач. Данное явление получило название "эффект Далла" и обусловило необходимость классификации задач по объему потребляемого процессорного времени, а также подразделения их на тяжелые и легкие. Было доказано, что данное явление имеет место как при рассмотрении однопроцессорной вычислительной техники, так и мультипроцессорной. Очевидно, это является весьма существенным вопросом при учете темпов увеличения числа вычислительных модулей на кристалле в современной микропроцессорной технике.

В данной статье рассмотрены основные подходы к вопросу преодоления эффекта Далла, применяемые на современном этапе развития науки и техники. Для данных подходов отмечен ряд существенных недостатков, определяющих границы их применимости, а также сформулирован взгляд автора на возможные пути снижения степени влияния обозначенных особенностей.

Преодоление эффекта Далла в современных методах мультипроцессорной диспетчеризации

Некоторое время считалось, что эффект Далла является своеобразным барьером для диспетчеризируемости и не может быть преодолен. Современная наука, тем не менее, доказала, что природа данного явления кроется в одновременном потреблении процессорного времени задачами с принципиально различными объемами требуемых вычислительных ресурсов и не зависит от числа имеющихся вычислительных модулей. В данном аспекте различными учеными был сформулирован подход к диспетчеризации задач в СРВ с позиции их классификации

по указанному признаку и обоснована необходимость рассмотрения специальных алгоритмов диспетчеризации, представленных в виде совокупности правил, подчеркивающих характерные особенности задач конкретного вида. Методы глобальной мультипроцессорной диспетчеризации, позволяющие успешно преодолевать ограничения, обусловленные наличием эффекта Далла, а также укладывающиеся в данную концепцию, были предложены, в частности, в работах [2—4]. Стоит отметить, что упомянутые в этих работах методы диспетчеризации RM_US [2] и SM_US [3] относятся к классу алгоритмов со статическим назначением приоритетов, в то время как RMZL [4] использует динамическую модель назначения и может в некоторой мере считаться их преемником.

Диспетчеризация со статическими приоритетами с точки зрения реализации и применения является наиболее простым методом планирования, обладающим также наименьшей трудоемкостью. Можно утверждать, что в данном аспекте при возникновении необходимых условий для применения методов данного класса предпочтение разработчиков СРВ будет отдано именно им с достаточно высокой долей вероятности.

В рамках статической диспетчеризации методами RM_US [2] и SM_US [3] авторы рекомендуют подразделять задачи на два непересекающихся подмножества, характеризующихся предложенным ими критерием классификации. При этом во избежание проявления эффекта Далла подмножество задач, требующее наибольшего объема процессорного времени, наделяется приоритетным преимуществом. Очевидно, подобное выделение некоторого подмножества задач может негативно сказаться на диспетчеризации задач, не вошедших в эту группу.

Учитывая указанную особенность, в качестве альтернативы предложенным методам рассматриваются динамические алгоритмы диспетчеризации. В данном аспекте метод RMZL [4], являясь примером реализации принципов динамической диспетчеризации, обладает рядом достоинств:

- в общем случае суммарное число наборов задач, успешно диспетчеризируемых динамическим методом, может превышать значение данного показателя для статических методов;
- реализация динамической диспетчеризации осуществляется существенно сложнее и связана с ощутимо большими накладными расходами на этапе исполнения;
- приоритетное преимущество подмножества так называемых "тяжелых" задач для данного класса алгоритмов диспетчеризации несущественно, поскольку приоритеты носят исключительно динамический характер.

Тем не менее, на фоне даже столь очевидных достоинств метод имеет ряд особенностей, оказывающих негативное влияние:

1. Поскольку при динамическом повышении приоритетов дисциплиной диспетчеризации используется условие равенства запаса времени нулю, имеется большая вероятность пропуска крайнего срока завершения задач. Данное обстоятельство, очевидно, может иметь исключительное значение при использовании метода в СРВ. Исходное утверждение базируется на том факте, что в любой момент задача с нулевым запасом времени может быть вытеснена на неопределенный интервал времени при межзадачном взаимодействии.

2. Особенностью планировщика, реализующего динамическое управление приоритетами, является необходимость выполнения некоторого объема вычислений по определению моментов времени, требующих немедленного повышения приоритетов задач (особенно при учете предыдущего замечания). Если за отправную точку брать предложенный авторами метод определения времени отклика задач [4], то учитывая его динамическое применение на этапе выполнения, можно судить о том, что объем подобных вычислений значительно превосходит затраты, требуемые для реализации статических методов.

3. Очевидно, непрерывно проводить указанные вычисления, без существенного влияния на производительность системы не представляется возможным. В данном аспекте возникает вопрос об оптимальном выборе точек перепланирования, соответствующих моментам активизации планировщика. Авторы метода в заключении к работе [4] отмечают, что вопрос о методах детектирования событий перепланирования в данный момент не решен. Таким образом, с определенной долей уверенности можно утверждать, что практическое применение данного метода затруднительно и связано с высокоточным определением флуктуаций времени исполнения задач. Очевидно, при учете данной особенности нельзя пренебрегать характерной чертой метода, указанной в п. 1.

К вопросу о диспетчеризации сверхлегких задач в условиях симметричного мультипроцессирования

Ввиду наличия отмеченных особенностей при динамической мультипроцессорной диспетчеризации применение упомянутых ранее методов планирования со статическими приоритетами для эффективного преодоления эффекта Далла является актуальным. Как уже отмечалось, данные методы наделяют приоритетным преимуществом подгруппу наиболее ресурсоемких объектов планирования, что негативно сказывается на диспетчеризации легких задач.

Ввиду того, что высокая ресурсоемкость задачи трактуется как превышение некоторого порогового

значения величиной отношения требуемого ей процессорного времени к периоду, нет никаких гарантий того, что тяжелые задачи предоставят процессорное время легким задачам до истечения отведенного им интервала времени. Задачи, чей крайний срок завершения и период выполнения (в подавляющем большинстве случаев эти параметры отождествляют) не превышают времени исполнения наименее ресурсоемкой тяжелой задачи либо превышают незначительно, имеет смысл именовать сверхлегкими. Наличие хотя бы одной сверхлегкой задачи при условии равенства или превышения числом тяжелых задач числа имеющихся процессоров даже при наилучшем фазировании (не говоря уже о критическом сценарии исполнения, соответствующем случаю наихудшего фазирования задач в системе) приводит к совершенной недиспетчеризируемости всей системы. Для систем жесткого реального времени данный результат зачастую характеризуется наличием ошибок при проектировании по части выбора аппаратного обеспечения или принципов диспетчеризации.

Недавние исследования в данной области позволяют утверждать, что общее число наборов задач, не способных успешно выполняться по причине наличия сверхлегких объектов диспетчеризации, достаточно велико при сравнительно высокой суммарной нагрузке на вычислительные модули в мультипроцессорной системе. Результаты оценки числа описанных явлений при генерации наборов задач методом UUniFast-Discard [5] для систем с четырьмя и восемью процессорами представлены на рис. 1 и рис. 2 соответственно. Указанный метод генерации наборов задач является мультипроцессорно-ориентированным преемником широко известного алгоритма UUniFast [6].

На рис. 1 и 2 параметрами *n*, *U* и *C* обозначены соответственно: число системообразующих задач реального времени, суммарный объем нагрузки на процессоры (выраженный в процентном отношении к предельно допустимой нагрузке на все вычислительные модули) и число неуспешных экспериментов (выраженное в процентном отношении к общему числу экспериментов в серии), в которых зафиксировано изучаемое явление.

В данном аспекте можно утверждать, что обсуждение вопроса преодоления приоритетной "дискриминации" сверхлегких задач имеет полное право на существование.

В качестве простейшего примера влияния на выполнимость СРВ сверхлегких задач достаточно рассмотреть диспетчеризацию СРВ с архитектурой симметричного мультипроцессирования, имеющей два процессора. Рассмотрим набор задач с конфигурацией:

- задачи *A* и *Б*, классифицируемые как тяжелые, со временем выполнения не менее 500 мкс и наивысшими приоритетами, статично назначенными алгоритмом диспетчеризации (во избежание проявления эффекта Далла);
- сверхлегкая задача *B* с крайним сроком завершения в 200 мкс.

В этом случае система не сможет успешно выполняться по причине пропуска крайнего срока завершения сверхлегкой задачей *B*.

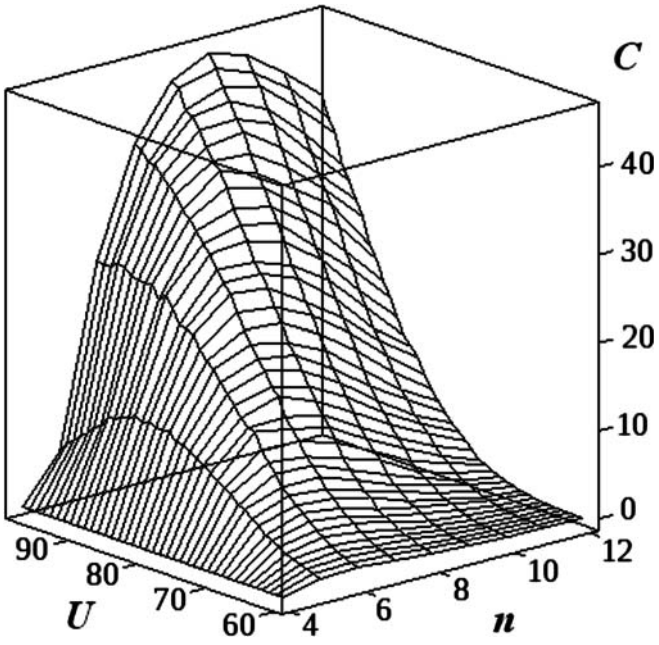


Рис. 1. Оценка числа отказов при диспетчеризации, обусловленных наличием сверхлегких задач в СРВ с четырьмя процессорами

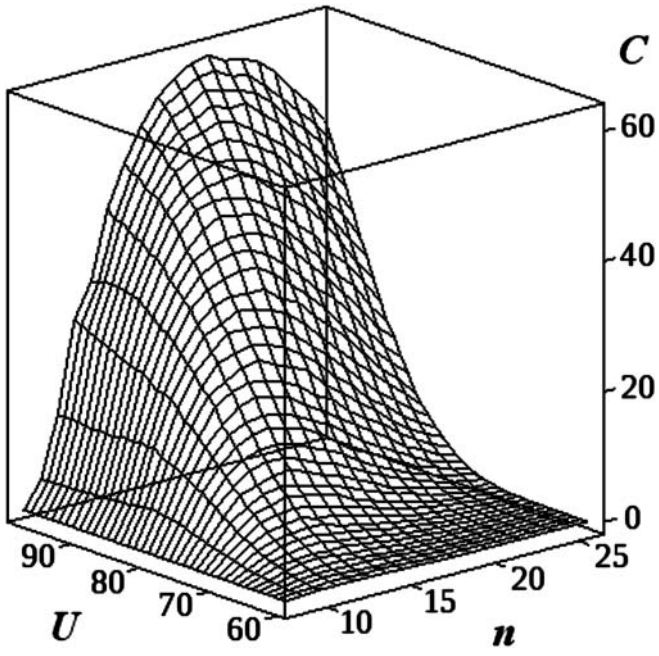


Рис. 2. Оценка числа отказов в диспетчеризации, обусловленных наличием сверхлегких задач в СРВ с восемью процессорами

Подходы к диспетчеризации сверхлегких задач и их применение на практике

Очевидно, диспетчеризация сверхлегких задач описанными методами статического планирования при указанных условиях не может быть успешно реализована по причине недостатка предоставляемых им ресурсов процессорного времени, что обусловлено неоспоримой принадлежностью задач данного класса к низкоприоритетному подмножеству.

Наиболее очевидным подходом к планированию подобных задач, судя по всему, является статическое повышение приоритетов при достижении числом тяжелых задач общего числа имеющихся вычислительных модулей. Сверхлегкие задачи в данном случае будут исключены из подмножества объектов диспетчеризации, характеризующихся недостаточным объемом предоставляемого процессорного времени. Таким образом, все системообразующие задачи реального времени можно будет разделить на три непересекающиеся группы: тяжелые, легкие и сверхлегкие, причем последние должны обладать существенным приоритетным преимуществом.

Данный подход в простейшем случае, очевидно, полностью исключает возможность влияния тяжелых задач на диспетчеризацию сверхлегких при допущении их независимости и отсутствии между ними взаимных блокировок на разделяемых ресурсах. Тем не менее, его применение на практике осложняется тем фактом, что здесь, по сути, происходит возврат к проблеме исключения негативного влияния эффекта Далла, поскольку вновь возникают предпосылки для его появления (возникновение высокоприоритетных задач, потребность в вычислительных ресурсах у которых ниже, чем у любой тяжелой задачи). Немаловажен также факт невозможности снижения степени приоритетной дискриминации между легкими и тяжелыми задачами.

Другим подходом к диспетчеризации сверхлегких задач при учете влияния обсуждаемой особенности является изменение принципа потребления процессорного времени тяжелыми задачами. В данном аспекте приемлемо рассмотрение концепции принудительных периодических блокировок на некоторый промежуток времени для наиболее приоритетных тяжелых задач. Группа тяжелых задач в этом случае обязуется периодически, на протяжении всего периода своего существования, предоставлять менее приоритетным задачам определенный (на этапе проектирования) квант процессорного времени, именуемый периодом блокировки. Очевидно, данный подход обусловливает некоторое увеличение накладных расходов, связанных с переключением контекстов задач (выгрузка текущего окружения вытесняемой задачи и загрузка окружения задачи, которая избрана для дальнейшего исполнения на процессоре) и перепланированием. Однако ввиду возможности широкого варьирования параметрами его применение тем актуальнее, чем менее существенны затраты на перепланирование (стоимость переключения контекстов задач может существенно разниться в зависимости от используемого аппаратного обеспечения и архитектуры). Параметрами, определяющими границы применимости метода, являются периоды блокирования и вытеснения тяжелых задач, а также общее число задач, применяющих концепцию принудительных блокировок.

Наряду с предоставлением дополнительных вычислительных ресурсов процессора сверхлегким задачам подход также позволяет несколько сгладить приоритетное неравенство между тяжелыми и легкими задачами. Применение указанного принципа выполнения тяжелых задач позволяет полностью исключить возможность проявления эффекта Далла в мультипроцессорной системе реального времени.

В качестве альтернативного пути при изучении особенностей диспетчеризации сверхлегких задач имеет смысл рассматривать объединение нескольких тяжелых задач в группу с одновременным запретом их миграции между процессорами. Данные меры позволят перераспределить процессорное время между менее приоритетными задачами, поскольку объем параллельно потребляемых ресурсов тяжелыми задачами существенно снизится. Очевидно, для реализации данного подхода может потребоваться обеспечение независимости между задачами в группе, что обусловливает необходимость проведения особенно тщательного отбора претендентов на объединение в группу. Методом назначения приоритетов задачам в группе вполне может являться даже классический алгоритм диспетчеризации RMS [7], однако во избежание проявления эффекта Далла правило приоритетного преимущества перед легкими задачами должно соблюдаться. Очевидно, применение указанного метода обусловит переход к частично раздельной диспетчеризации, где для ряда задач допустим запрет в перемещении между процессорами. В отличие от раздельной диспетчеризации методы глобального планирования не ограничивают миграцию задач.

Последний подход позволяет минимизировать число переключений контекстов задач на фоне снижения влияния тяжелых задач на все исполняющиеся в системе объекты диспетчеризации реального времени. Здесь, так же как и в предыдущем методе диспетчеризации, эффект Далла не имеет силы, поскольку приоритетное преимущество тяжелых задач сохраняется. Тем не менее, обеспечение независимости (или, по крайней мере, ослабление зависимости) задач, объединенных в группу, может в ряде случаев стать непреодолимым препятствием для применения подхода на практике.

Выводы и предпосылки для дальнейшего развития идей

Достижения мировой науки в области диспетчеризации задач реального времени за последние десятилетия пополнились множеством эффективных решений, ориентированных, в том числе, и на мультипроцессорные архитектуры. Обнаружение принципиально новых подходов в данном вопросе сейчас является событием столь редким, что интерес к методам повышения эффективности существующих решений со временем лишь возрастает. В данном аспекте особенно заманчивыми с точки зрения применения на практике являются исследования в области расширения границ применимости методов статической диспетчеризации.

В представленной статье рассмотрены современные методы преодоления эффекта Далла — известной особенности диспетчеризации, характерной для разнородных с точки зрения потребления процессорного времени задач. Как выяснилось, при диспетчеризации в мультипроцессорных СРВ на успешность исполнения набора задач может влиять наличие так называемых сверхлегких задач. В предложенных подходах к диспетчеризации задач данного класса имеются принципиальные отличия, определяющие возможность их применения в конкретных ситуациях. С очевидностью можно утверждать, что конечный выбор в любом случае должен проводиться в контексте учета особенностей конкретной архитектуры, аппаратных особенностей и требований к разрабатываемой системе. Дальнейшие исследования в данном вопросе, очевидно, должны обозначить границы применимости представленных в статье методов диспетчеризации с учетом их особенностей и недостатков.

В качестве эксперимента в рамках аппарата компьютерного моделирования была проведена оценка влияния сверхлегких задач на диспетчеризацию в мультипроцессорных системах реального времени при генерации наборов задач с равномерно распределенными параметрами. В результате было выявлено, что данная величина не может считаться пренебрежимо малой и требует детального изучения.

Науке известно множество примеров, в которых привнесение минимальных изменений в логику выполнения задач позволяло обеспечить общую успешность диспетчеризации задач в мультипроцессорной СРВ. Данное утверждение в первую очередь касается общепринятых методов преодоления эффекта Далла. В данном аспекте актуальность рассмотрения диспетчеризации сверхлегких задач очевидна.

Список литература

1. **Dhall S. K., Liu C. L.** On a Real-Time Scheduling Problem // Operating Research. 1978. Vol. 26. № 1. P. 127—140.
2. **Andersson B., Baruah S., Jonsson J.** Static-priority scheduling on multiprocessors // Proc. of the 22nd IEEE Real-Time Systems Symposium, December 2001. P. 193—202.
3. **Andersson B.** Global Static-Priority Preemptive Multiprocessor Scheduling with Utilization Bound 38 % // Proc. of the 12th International Conference on Principles of Distributed Systems, December 2008. P. 73—88.
4. **Kato S., Takeda A. and Yamasaki N.** Global Rate-Monotonic Scheduling with Priority Promotion // Technical Report CMU-ECE-TR10-05, May, 2010.
5. **Davis R. I., Burns A.** Priority assignment for global fixed priority preemptive scheduling in multiprocessor real-time systems // Proc. of the 30th IEEE Real-Time Systems Symposium (RTSS 2009), December 2009. P. 398—409.
6. **Bini E., Buttazzo G.** Measuring the performance of schedulability tests // Real-Time Systems. 2005. Vol. 30. № 1—2. P. 129—154.
7. **Liu C. L., Layland J. W.** Scheduling algorithms for multiprogramming in a hard-real-time environment // Journal of the ACM. 1973. Vol. 20. № 1. P. 46—61.

УДК 004.89

И. П. Норенков, д-р техн. наук, проф.,
e-mail: norenkov@wwwcdl.bmstu.ru,
М. Ю. Уваров, инженер,
МГТУ им. Н. Э. Баумана

Задачи обработки знаний на основе ролевой кластеризации онтологий*

Предлагается для решения задач извлечения знаний из документов и их обработки использовать метод ролевой кластеризации онтологий, в соответствии с которым концепты онтологии распределяются между ролевыми кластерами и при поиске документов, содержащих сведения о принятии решений, классификации и аннотировании документов, формировании запросов на поиск информации используются сложные концепты, образуемые из понятий разных кластеров. Приведены примеры применения метода.

Ключевые слова: *извлечение знаний, обработка знаний, кластеризация онтологий, классификация и аннотация документов, поддержка принятия решений*

Введение

Извлечение знаний из текстовых документов и их обработка являются неотъемлемой частью большинства интеллектуальных процессов, выполняемых человеком в различных сферах деятельности. К числу основных задач извлечения знаний из документов и их обработки, называемых задачами TextMining, относятся классификация, аннотирование и упорядочение документов, информационный поиск, поддержка принятия решений (ППР), выявление нечетких дубликатов публикаций. Слабая структурированность представленных в документах знаний обусловливает трудности формализации этих задач, поэтому применяют неавтоматизированные способы их решения, что часто приводит к неэффективности результатов и принимаемых решений.

Актуальность создания автоматических или полуавтоматических методов решения задач TextMining нашла отражение во многих публикациях последнего времени.

Один из подходов к решению задач TextMining основан на выявлении в текстах документов фрагментов, содержащих ключевые слова или фразы, характерные для определенной предметной области [1], или характеризующие конкретные события, ситуации, факты [2]. Ими могут быть фамилии людей, даты, адреса, названия организаций, географических пунктов и т. п. Полуавтоматическое аннотирование документов и извлечение нужных данных при этом происходит на основе предварительного обучения системы TextMining, выполняемого пользователем [3]. В других подходах к автоматизации аннотирования используют прикладные онтологии. Аннотации составляются из терминов концептов, найденных в текстах документов [4—7].

Для задач классификации необходимо предварительное выделение рубрик (тематических кластеров), перечень которых либо получается путем обработки обучающей выборки, состоящей из документов, специально отобранных пользователями-экспертами [8], либо задается в соответствии со списком имеющихся частных (задачных) онтологий [9]. При классификации оцениваются степени принадлежности документов тематическим кластерам, определяемые суммами взвешенных частот появления концептов частных онтологий в документе [10].

Проблемы информационного поиска связаны со стремлением повысить его эффективность. Для оценки релевантности документа запросу используют ряд показателей и подходов [11], в том числе онтологический подход, например, связывая онтологию с ГРНТИ [12]. В работах [13, 14] релевантность определяется на основе анализа графовых моделей запроса и документа. В работе [15] предлагается использовать персонифицированные онтологии, формируемые на основании анализа поведения пользователя в процессе поиска в сети Интернет.

Выявление нечетких дубликатов статей выполняется в целях определения плагиата и устранения повторяющихся или близких по содержанию документов из результатов информационного поиска. Как отмечено в работе [16], актуальной остается задача выявления устойчивых словосочетаний в анализируемых документах и оценки степени их совпадения.

Упорядочение документов обычно требуется при формировании учебных материалов и различного рода отчетной документации. Создание электронных учебных пособий из отдельных учебных модулей на основе онтологий рассмотрено в работе [17].

Наиболее трудной для автоматизации является задача принятия решений. Для отдельных узкоспециализированных приложений успешно применя-

*Работа выполнена при финансовой поддержке РФФИ (12-07-002222).

ется подход, основанный на правилах (RBR) [18], реализуемый преимущественно в экспертных системах. Для извлечения сведений о решениях, описываемых в документах, используют подход, основанный на прецедентах (CBR) [19]. При этом обработка найденных по запросу документов в целях выявления решения осуществляется вручную.

Однако в настоящее время требования к эффективности решения большинства задач TextMining удовлетворяются лишь в малой мере. Необходима бóльшая степень автоматизации решения задач извлечения знаний из документов и их обработки, в том числе на основе онтологического подхода. Одним из путей развития онтологического подхода к решению задач TextMining является предлагаемый в статье метод ролевой кластеризации онтологий.

Ниже после определения основных используемых понятий рассматриваются примеры применения метода ролевой кластеризации онтологий к решению ряда задач извлечения знаний из текстовых документов.

Основные понятия метода ролевой кластеризации онтологий

Сложным концептом $\mathbf{k}_{сл}$ назовем упорядоченное подмножество простых концептов:

$$\mathbf{k}_{сл} = (k_1, k_2, ..., k_n). \quad (1)$$

Простой концепт k_i — концепт, являющийся элементарным понятием в пределах данной онтологии.

Простые концепты выполняют ту или иную роль в составе сложного концепта. Например, ролями могут быть "объект", "свойство", "действие", "средство", тогда сложные концепты суть отношения типа "свойство—объект", "действие—объект", "средство—действие—объект" и т. п.

Ролевая кластеризация онтологии заключается в распределении простых концептов по кластерам в зависимости от выполняемой ими роли в сложных концептах. Возможны различные варианты кластеризации, определяемые выбором ролей (кластеров), подмножеством используемых простых концептов и их соответствием выбранным ролям. Поскольку в разных сложных концептах некоторый простой концепт может выполнять различные роли, то такой концепт может быть включен в более чем один кластер.

Тип отношения (тип сложного концепта), называемый паттерном, задается последовательностью ролей составляющих его простых концептов. Для каждого паттерна (1) эта последовательность, в свою очередь, задает упорядочение самих кластеров. Пусть $\mathbf{A}_i$ — i-й ролевой кластер, k_j — j-й концепт. Если $k_q \in \mathbf{A}_i$, $k_p \in \mathbf{A}_j$ и $i < j$, то сложный концепт образуется при условии $k_q \rangle k_p$, где $\rangle$ — знак предшествования, т. е. концепт k_q занимает позицию впереди концепта k_p в сложном концепте (1). Например: если роль a_3 кластера $\mathbf{A}_3$ — свойство, роль a_4 кластера $\mathbf{A}_4$ — объект, то свойство "скорость" и объект "автомобиль" образуют сложный концепт "скорость автомобиля", но не "автомобиль скорости".

Таблица 1

$\mathbf{A}_i$	$\mathbf{A}_j$	$\mathbf{A}_k$	$\mathbf{A}_m$
алгоритм	верификация	полоса пропускания	заряд
дедукция	визуализация	потребность	запрос
диакоптика	влияние	поток	защита информации
допущение	восстановление	приоритет	знания
идея	выбор	производительность	кортеж
индукция	выделение	работоспособность	криптоанализ
мера	вычисление	разрез	криптография
метод	генерация	реентерабельность	кроссовер
методика	использование	скорость	кубит
методология	испытание	робастность	кэш-память
модель	исследование	связанность	линия связи
подход	квантификация	система	ЛИСП
проблема	коммутация	состояние	логистика
предположение	распознавание	спектр	индекс
способ	распределение	структура	индексный файл
средство	распространение	функция	инновация

В базовом варианте кластеризации выделены роли a_i — средство, a_j — действие, a_k — свойство, a_m — объект и соответственно кластеры $\mathbf{A}_i$, $\mathbf{A}_j$, $\mathbf{A}_k$, $\mathbf{A}_m$. Примеры концептов, принадлежащих кластерам $\mathbf{A}_i$, $\mathbf{A}_j$, $\mathbf{A}_k$, $\mathbf{A}_m$-прикладной онтологии "Информационные технологии", приведены в табл. 1. Примером паттерна $a_i\ a_j\ a_m$ может служить "метод исследования линии связи", паттерна $a_j\ a_k\ a_m$ — "расчет производительности компьютера" и т. п.

Задачи извлечения и обработки знаний основаны на распознавании экземпляров паттернов в текстах документов. При этом в качестве терминов принимаются основы слов, допускаются расстояния между концептами k_i в составе паттерна не более заданного порога d, обычно d не превышает значений одно или два слова.

Аннотирование документов

Автоматическое аннотирование документа заключается в выделении в его тексте паттернов заданных типов, совокупность которых может быть принята в качестве сокращенной аннотации. Более содержательная аннотация получается как последовательность предложений из текста, в состав которых входят найденные экземпляры паттернов. Целесообразно использовать полуавтоматическое аннотирование, при котором вручную выполняется стилистическое редактирование автоматически полученной аннотации.

Таблица 2

1. М. А. Кожевников, Ю. В. Марчук, О. Н. Хамидулина, А. И. Монтиле, И. А. Погосян. Разработка средства поддержки диагностики ортопедической патологии на основе дискриминантного анализа клинико-анамнестических данных	
В работе затронут вопрос аналитической поддержки процесса дифференциальной диагностики патологии шейного отдела позвоночника (ШОП). Представлены технология выявления типа нарушения ШОП, а также механизм формирования структурной патологии верхнешейного отдела позвоночника. Разработаны и апробированы алгоритмы информационно-программной поддержки на основе адаптации дискриминантного анализа (ДА). Выявлена несостоятельность ДА как средства многомерного анализа данных в вопросах поддержки диагностики ортопедической патологии без предварительной его адаптации.	Теоретическим вопросам в части математического моделирования медицинской диагностики и применения моделирования для **решения актуальных практических задач, а также проблемам в области разработки медицинских информационных систем** поддержки диагностики посвящены работы, в основном касающиеся вопросов теоретического характера, строгих постановок задачи, применения математического анализа для разработки средств постановки диагноза. Авторами **предложена балльная система** оценки качественных показателей состояния опорно-двигательного аппарата. **Предложена система** оценки показателей, характеризующих жалобы пациента и его анамнез. Разработанная на основе результатов **исследования информационно-интеллектуальная система** поддержки диагностики (свидетельство об официальной регистрации программы для ЭВМ № 2007614539) позволяет учесть особенности предметной области и повысить качество диагностических мероприятий.
2. А. С. Клещев, М. А. Князева. Интернет-система управления информацией о преобразованиях программ	
Представлена интернет-система управления информацией о преобразованиях программ. В качестве формализма представления знаний используется онтология предметной области преобразований программ. Информационное обеспечение и доступ к системе осуществляется специализированным банком знаний о преобразованиях программ.	Основной **проблемой в сфере науки о преобразованиях программ** является невозможность своевременно выполнять компьютерные эксперименты. Единственным **средством проведения подобных экспериментов являются оптимизирующие компиляторы.** В сфере разработки оптимизирующих компиляторов основными проблемами являются разнородность систем понятий и **моделей в области преобразования программ,** невозможность макетирования оптимизирующих компиляторов и их быстрое моральное старение. Таким подходом является интеллектуальная система, моделирующая **процесс преобразования программ,** управляемый знаниями. Работа выполнена при финансовой поддержке ДВО РАН, инициативный научный проект "Интернет-система управления информацией о преобразованиях программ", при финансовой поддержке РФФИ, проект **"Исследование возможностей коллективного управления в семантическом вебе** информационными ресурсами различных уровней общности" (06-070-89071-а). **Средства визуализации историй преобразований программ** и отчетов по качеству процесса оптимизации программ. Для решения научных, практических и образовательных **проблем в области преобразования программ** предложена концепция интеллектуальной системы управления знаниями об оптимизации программ. Редактор ИРУО используется и как инструментальное **средство для создания редакторов банка знаний.** Разработка макетов оптимизирующих компиляторов открывает возможность для проведения **исследований стратегий и наборов преобразований программ** в таких компиляторах. Для лучшего усвоения знаний разработан учебно-методический комплекс по дисциплине "Теория вычислительных процессов и структур II. Теория и **методы трансляции**" и **"Оптимизация программ"** для студентов, обучающихся по специальности 351500 — математическое обеспечение и администрирование информационных систем
3. Д. А. Васинев, Д. Л. Жусов, М. С. Цынгаев, И. Ю. Ватулин. Предложения по повышению защищенности Ethernet-сетей от атак канального уровня	
В статье рассмотрены существующие способы защиты от атак данного класса и предложен вариант защиты на основе проверки правильности заполнения ARP-таблицы на серверах, клиентских рабочих станциях, а также коммуникационном оборудовании	Для обеспечения безопасности ИВС необходимо решение следующих частных задач: **анализ известных реализаций атак канального уровня** на сегмент сети Ethernet и существующих способов защиты от атак данного класса; разработка модели защиты сегмента сети Ethernet от атак канального уровня; разработка программного прототипа, позволяющего повысить защищенность сегмента сети Ethernet от атак канального уровня. На основе **анализа известных реализаций атак канального уровня** на сегмент сети, а также существующих способов и средств защиты от них была разработана функциональная модель защиты Ethernet-сети от компьютерных атак канального уровня. Построена функциональная **модель процесса защиты Ethernet-сети от атак канального уровня**, учитывающая особенности их защиты

Окончание таблицы 2

4. Г. Ф. Малыхина, А. В. Меркушева. Совместное использование нейронной сети и вейвлет-преобразования при анализе нестационарного сигнала	
Представлены некоторые приложения на основе совместного использования нейронной сети и вейвлет-преобразования (ВП) в задачах анализа нестационарных сигналов. Первая задача относится к системе мониторинга речевого сигнала, в которой выполнение ВП используется для сжатия речевого сигнала, идентификации распределения вероятности ВП сигнала и вейвлет-фильтрации с адаптивным порогом. Алгоритм нейронной сети используется для распознавания типа фрейма сигнала (речь/пауза) и управления двумя фазами: определения порогов (на фреймах паузы) и вейвлет-фильтрации речевого сигнала на основе этих порогов (на фреймах речи). С помощью такого метода достигается адаптация к изменению уровней шума. Разложение сигнала обеспечивается на основе вейвлет-пакета, частотные субполосы которого аппроксимируют критические полосы персептуальной модели. Получены новые соотношения для вычисления порогов на основе экспоненциально-степенного распределения, которые являются обобщением выражений Крамера для распределения максимальной величины Гауссовской переменной. Это позволяет фильтровать (в вейвлет области) шумы самого общего типа. Во второй задаче предложен алгоритм для определения уровней свободного газа в трубопроводе коммерческой нефти. Метод основан на бесконтактном радиационном плотномере. Показания плотномера обрабатываются нейронной сетью, которая выполняет идентификацию в потоке нефти фреймов, информационно значимых для вейвлет-преобразований сигнала: временных интервалов с газом и интервалов, не содержащих газа	Сложные сигналы, подлежащие контролю, **обработке и анализу в измерительной информационной системе** (ИИС), как правило, отражают нестационарные процессы. Ниже будут рассмотрены две прикладные задачи, решение которых основано на совместном использовании методов **вейвлет-преобразования и нейросетевого алгоритма для идентификации типа фреймов** контролируемых сигналов. Нейросетевой **алгоритм анализа речевого сигнала** в области его вейвлет-отображения. Детектирование, как **задача бинарного распознавания, реализуется нейронной сетью** в форме двухслойного перцептрона. При обучении персептрона использовано два способа: на основе алгоритма с обратным распространением ошибки и на основе **оптимизации нелинейной функции отображения сети** вход—выход. **Поиск новых форм анализа измерительной информации.** Вторая составляющая **метода — применение нейронной сети** для определения типов интервалов регистрации (с чистой нефтью или с примесью газообразной фракции). Выполнено **исследование, позволившее выбрать структуру многослойного персептрона** и обосновать метод обучения, имеющего лучшую скорость сходимости. Другой подход к обучению персептрона для детектирования типа интервала основан на трактовке этой фазы как **оптимизации нелинейной функции отображения сети** вход—выход. Он использован в моделирующей программе при исследовании характеристик **метода обработки информации**
5. В. В. Грибова, Н. Н. Черкезишвили. Развитие онтологического подхода для автоматизации разработки пользовательских интерфейсов с динамическими данными	
В данной работе предлагается концепция автоматизации разработки пользовательских интерфейсов с динамическими данными, которая является дальнейшим развитием онтологического подхода. Интерфейсы с динамическими данными допускают формирование наборов входных/выходных данных динамически во время работы приложения, тогда как для интерфейсов со статическими данными требуется точное определение наборов входных/выходных данных во время их разработки. Интерфейсы с динамическими данными дают более высокую степень гибкости по сравнению с интерфейсами со статическими данными, что дает принципиально новые возможности их использования и расширяет круг применения таких интерфейсов. Однако современные средства автоматизации разработки интерфейсов поддерживают проектирование и автоматическую генерацию только интерфейсов со статическими данными. Предлагаемая в работе концепция направлена на устранение их недостатков	Для **автоматизации разработки пользовательских интерфейсов** в настоящее время используются построители WIMP-интерфейсов, моделеориентированные средства и средства, основанные на онтологическом подходе. Однако все перечисленные **средства ориентированы на автоматизацию разработки пользовательских интерфейсов** со статическими данными; автоматизация разработки интерфейсов с динамическими данными не поддерживается. Такие интерфейсы либо полностью реализуются на языках программирования (C++, Java, Pascal и др.), либо некоторые компоненты интерфейса реализуются с использованием специализированных **средств автоматизации разработки, остальные реализуются на языках** программирования. В результате **проектирование, реализация и особенно сопровождение пользовательских интерфейсов** оказываются чрезвычайно трудоемкими. Целью данной работы является представление концепции **автоматизации разработки интерфейсов** с динамическими данными. Предлагаемая в работе концепция **автоматизации разработки интерфейсов** с динамическими данными является развитием онтологического подхода. Основными положениями автоматизации **проектирования, реализации и сопровождения пользовательского интерфейса** на основе онтологического подхода являются: **проведение анализа профессиональной деятельности, связанной с разработкой пользовательского интерфейса:** выделение групп специалистов, осуществляющих разработку и **сопровождение пользовательского интерфейса, а также систем понятий,** которые они используют в своей работе; **построение онтологий пользовательского интерфейса,** необходимых для того, чтобы в их терминах разработчики интерфейса могли определять и модифицировать структуру конкретной модели пользовательского интерфейса; **разработка модели пользовательского интерфейса в терминах онтологий,** которая является конкретизацией онтологий пользовательского интерфейса; **построение алгоритма автоматического преобразования модели интерфейса в программный код,** который управляется онтологиями пользовательского интерфейса, при этом характеристики конкретной модели являются входными данными для этого алгоритма

В табл. 2 представлены результаты аннотирования нескольких статей из научно-технического журнала "Информационные технологии". Левая колонка таблицы содержит авторскую аннотацию, правая колонка — автоматически составленную из предложений с экземплярами отношений заданных типов. Использовались следующие паттерны: "действие—свойство—объект", "средство—действие—объект", "средство—действие—свойство—объект", "действие—средство—действие—объект", "действие—действие—объект" и "средство—действие—действие—объект". Расстояние d принято равным 2. Экземпляры паттернов в таблице выделены полужирным шрифтом. Для некоторых из аннотаций в правой колонке выполнено редактирование (стилистическая правка и удаление некоторых предложений). Автоматически составленные аннотации признаны корректными. Отметим, что первая из аннотируемых статей посвящена проблеме использования информационных технологий в медицине. Поскольку нами использовалась онтология, относящаяся только к информационным технологиям, то и в аннотации акцент оказался смещенным именно на вопросы информационных технологий.

Классификация документов

Классификация документов выполняется на основе предварительно выбранных рубрик, для которых должны быть разработаны частные онтологии. Степень R_{kl} принадлежности k-го документа l-й рубрике оценивается по n_{kl} — числу появлений концептов заданных типов из l-й частной онтологии в k-м документе.

В экспериментах классификация выполнялась по отношению к набору статей из научно-технического журнала "Информационные технологии". Примеры полученных значений степеней принадлежности R_{kl} показаны в табл. 3. Частные онтологии соответствовали рубрикам, часть из которых приведена в левой колонке табл. 3. При определении n_{kl} учитывались указанные выше типы сложных концептов (те же, что и при получении аннотаций табл. 2). Степени принадлежности определялись по формуле

$$R_{kl} = n_{kl}/N_k,$$

где N_k — объем k-го документа (в тысячах слов).

Таблица 3

Частная онтология	Номер статьи							
	1	2	3	4	5	6	7	8
Компьютеры	0,4		0,5		**19,8**	1,3		1,2
Сети		0,5	**22,1**	0,7		1,2		1,2
Программная инженерия		**46,8**			0,7		0,8	
Безопасность информации			**5,2**	0,4				
Базы данных						**17,1**		
Интеллектуальные системы	0,9	**9,4**		**3,2**			0,8	
Управление знаниями	**9,3**	**9,3**		0,4				
Компьютерная графика								**24,3**
Генетические алгоритмы							**10,7**	
Синтез расписаний		0,4				**5,5**	**12,3**	

Примечание. Названия статей 1—5 приведены в табл. 2. Названия остальных статей: 6. Проблема оптимизации запросов в многомерной распределенной СУБД. 7. Применение аппарата генетических алгоритмов для моделирования и формирования расписания операций в многопрофильной клинике. 8. Описание виртуальной модели изделия в индустрии моды с позиции геометрического моделирования формы.

В табл. 3 полужирным шрифтом выделены значения R_{kl} для рубрик l, к которым следует относить k-ю статью. Очевидно, что некоторые статьи относятся одновременно к более чем одной рубрике, как, например, статья 7.

Информационный поиск

Применение ролевой кластеризации онтологий для информационного поиска имеет следующие преимущества.

1. Формирование содержательных сниплетов, в качестве которых принимаются экземпляры паттернов.

2. Автоматическое формирование расширенных запросов. В расширенном запросе ключевые словосочетания представлены экземплярами паттернов, формируемыми на основе одного или нескольких заданных ключевых слов. Используемые при этом

Таблица 4

Подкластеры			
Создание	Применение	Исследование	Принятие решения
проектирование, конструирование, создание, разработка, оптимизация, планирование, преобразование, развитие, сборка, синтез, совершенствование, сочетание, улучшение, формирование, построение, находка, модификация	использование, обслуживание, применение, сопровождение, управление, реализация, функционирование	анализ, верификация, декомпозиция, изучение, исследование, моделирование, определение, поиск, оценка, прогнозирование, прототипирование, тестирование, обоснование, идентификация	выбор, разработка, обоснование, предложение, размещение, решение, формирование, описание, рассмотрение, альтернатива, раскрытие, позиционирование, сопоставление, генерация, построение, модификация

ролевые кластеры могут относиться к тому или иному аспекту проблемы. Например, пусть кластер "действие" разделен на подкластеры с ролями "создание", "применение", "исследование", "продажа" и т. п. В табл. 4 приведены примеры концептов из некоторых подкластеров.

Достаточно задать некоторое ключевое слово A из кластера "объект" и одно из слов из определенного подкластера, и система сформирует множество экземпляров паттернов в виде отношений "концепты заданного подкластера — A".

3. По экземплярам паттернов, найденным в анализируемых документах, можно оценить степень близости их содержания и, следовательно, установить дублирование или выявить плагиат.

Поддержка принятия решений (ППР)

С помощью метода ролевой кластеризации удается отобрать документы, перспективные, с точки зрения содержания в них искомого решения. Подход напоминает известный метод поиска по характерным фразам, но онтологии позволяют не задавать в исходном виде такие фразы, а динамически их конструировать из представителей разных кластеров.

Так, на места кластеров "объект", "действие", "средство" были помещены кластеры А1, А2, А3:

— А1, включающий концепты: вариант, альтернатива, метод, решение, подход, критерий, целевая функция;

— А2, состоящий из концептов: множество, совокупность, набор, выбор, сравнение, сопоставление, новый, оригинальный;

— А3, состоящий из концептов: предложение, описание, разработка, оценка.

Далее осуществлялся информационный поиск в целях выделения документов, имеющих отношение к теме ППР. В результате в исследованных документах были автоматически найдены фразы (сниплеты): "Поставлена задача **выбора эффективных вариантов** энергосиловых установок летательных аппаратов и вариантов конструкции детонационного двигателя, которая сводится к задаче гипервекторного ранжирования", "метод анализа иерархий (МАИ), — простое и удобное средство, позволяющее структурировать проблему, оценивать построенный **набор альтернатив** по каждому из заданных критериев", "В работе проводится **сравнение методов** k-средних, PAM и обобщенного метода k-средних для данной задачи", "**Предложен новый подход** к решению указанных проблем и описана разработанная технология быстрого вычисления характеристик сложных технических объектов" и т. д., свидетельствующие о перспективности документов с этими фразами. Подобный поиск выполняется среди множества документов, предварительно отобранных по показателю R_{kl} принадлежности к определенной предметной области.

Заключение

Онтологии широко используются в интеллектуальных системах для решения актуальных задач управления знаниями. Одновременно с разработкой онтологий целесообразно выполнять ролевую кластеризацию концептов. Использование метода ролевой кластеризации позволяет успешно решать большинство задач извлечения знаний из текстовых документов и их обработки.

Список литературы

1. **Li Y., Zhang L., Yu Y.** Learning to Generate Semantic Annotation for Domain Specific Sentences // In K-CAP 2001 Workshop on Knowledge Markup & Semantic Annotation. 2001.
2. **Muslea I.** Extraction Patterns for Information Extraction Tasks: A Survey // In. AAAI-99 Workshop on Machine Learning for Information Extraction. 1999.
3. **Kushmerick N., Weld D., Doorenbos R.** Wrapper Induction for Information Extraction // In Proc. 15th Int. Joint Conf. AI. 1997. P. 729—735.
4. **Fernández M., Vallet D., Castells P.** Automatic Annotation and Semantic Search from Protégé // In 8th International Protege Conference, Madrid, Spain. 2005.
5. **Handschuhl L. S., Staabl L. S., Ciravegna F.** S-CREAM — Semi-automatic CREAtion of Metadata // In Proc. of the European Conference on Knowledge Acquisition and anagement EKAW-2002. Madrid: Springer, 2002.
6. **Kiryakov A., Popov B., Terziev I., Manov D., Ognyanoff D.** Semantic Annotation, Indexing, and Retrieval // Journal of Web Sematics. Is. 1. 2005. P. 47—49.
7. **Castells P., Fernandez M., Vallet D.** An Adaptation of the Vector-Space Model for Ontology-Based Information Retrieval // IEEE Transactions on Knowledge and Data Engineering 19(2), Special Issue on Knowledge and Data Engineering in the Semantic Web Era. February 2007. P. 261—272.
8. **Шабанов В. И., Андреев А. М.** Метод классификации текстовых документов, основанный на полнотекстовом поиске // Труды РОМИП'2003. СПб.: НИИ Химии СПбГУ, 2003. С. 52—71.
9. **Nagarajan M., Sheth A., Aguilera M., Keeton K., Merchant A., Uysal M.** Altering Document Term Vectors for Classification — Ontologies as Expectations of Co-occurrence // 16th World Wide Web Conference, 2007. P. 1225—1226.
10. **Толчеев В. О.** Методы выявления информативных признаков в задаче классификации текстовых документов // Информационные технологии. 2005. № 8. С. 14—21.
11. **Manning C., Raghavan P., Schütze H.** Introduction to Information Retrieval. Cambridge University Press, 2008. 544 p.
12. **Вдовицын В. Т., Лебедев В. А.** Технологии систематизации и поиска электронной научной информации с применением онтологий // Информационные ресурсы России. 2010. № 5.
13. **Baziz M., Boughanem M., Pasi G., Prade H.** An Information Retrieval Driven by Ontology from Query to Document Expansion // Conference RIAO2007. Pittsburgh PA, U.S.A. 2007.
14. **Карпенко А. П.** Оценка релевантности документов онтологической базы знаний // Наука и образование. Электронное научно-техническое издание. 2010. № 9.
15. **Calegari S., Pasi G.** Ontology-Based Information Behaviour to ImproveWeb Search // Future Internet. 2010. N 2. P. 533—558.
16. **Толчеев В. О.** Анализ проблемы и разработка процедуры выявления нечетких дубликатов научных статей по библиографическим описаниям // Информационные технологии. 2011. № 2. С. 17—21.
17. **Норенков И. П.** Документальные базы знаний на основе онтологий // Информационные технологии. 2011. № 2. С. 11—16.
18. **Ligeza A.** Logical Foundations for Rule-Based Systems // Series Studies in Computational Intelligence. V. 11. Springer-Verlag, 2006. 329 p.
19. **Aamodt A., Plaza E.** Case-Based Reasoning: Foundational Issues, Methodological Variations and System Approaches // AI Commutations. IOS Press. 1994. V. 7, 1. P. 39—59.

УДК 519.76; 811

А. В. Болховитянов, аспирант,
МГТУ им. Н. Э. Баумана,
e-mail: alex.daiv@gmail.com,
А. М. Чеповский, канд. техн. наук, доц.,
НИУ ВШЭ

Методика автоматического выделения структурных единиц в предложениях на русском языке

Предлагаются математические модели, предназначенные для анализа предложений на русском языке в целях выделения именных групп и причастных оборотов. Алгоритм выделения причастных оборотов основан на модели глагольного управления в русском языке. Предлагаемые алгоритмы предназначены для уменьшения возможных связей в дереве синтаксического разбора, что позволяет применять такого рода анализ в промышленных задачах обработки текстовой информации.

Ключевые слова: *обработка естественно-языковых текстов, именная группа, глагольное управление, когнитивный семантический анализ*

Введение

Задачей семантического анализа является создание представления текста, позволяющего соотносить информацию, содержащуюся в тексте, с системой знаний. Формирование такого промежуточного представления является целью когнитивного анализа текста [1]. Для решения задачи когнитивного моделирования текста требуется создание математических моделей, позволяющих решать инженерные задачи обработки текстов.

Такого рода задачи хорошо изучены и имеют решения, использующиеся в промышленных системах обработки текстовой информации, для языков с достаточно простой синтаксической структурой, например, для английского языка, в котором порядок слов фиксирован. В то же время для русского языка при наличии развитой системы флексий порядок слов может существенным образом варьироваться. Это в свою очередь порождает множество синтаксических связей при анализе и не позволяет применять методы синтаксического анализа в промышленных системах. Для решения указанных проблем применяют ряд методов, одним из которых является выделения единых синтаксических единиц — именных групп и причастных оборотов. Это позволяет значительно сократить число возникающих синтаксических связей.

В проекте "Автоматическая обработка текста" (АОТ) была разработана собственная система синтаксического анализа, способная выделять именные группы [2]. Однако ее основной недостаток заключается в том, что она основана на шаблонах, описываемых регулярным языком. Использование такого рода шаблонов ограничивает глубину построения именных групп. Предлагаемая в настоящей работе модель позволяет выделять именные группы произвольной сложности и глубины.

Формирование промежуточного представления требует информацию о синтаксической структуре анализируемого текста. Результат синтаксического анализа представляется в виде дерева синтаксических зависимостей [3], которое может быть получено на основе связей по управлению между составными элементами текста. Таким образом, перед нами встает задача установления синтаксических связей по управлению между синтаксическими единицами предложения.

В данной работе предлагаются математические модели и алгоритмы, позволяющие решать (в некотором допустимом для реальных систем обработки текстовой информации приближении) задачи построения синтаксических связей по управлению между элементами текста.

Модель для выделения причастных оборотов основана на модели глагольного управления в русском языке [4]. На основе этой модели был составлен словарь [5], который и был взят за основу при построении математической модели глагольного управления в разрабатываемой системе.

Математическая модель морфологического анализа

В данном разделе приводятся математические модели и алгоритмы, предназначенные для выделения именных групп и причастных оборотов из предложений на русском языке.

Введем вспомогательные множества, используемые в дальнейшем:

- множество частей речи $PoS = \{NOUN, ADJ, VERB, CNUM, PART\}$. Элементами множества являются части речи русского языка: *NOUN* — имя существительное, *ADJ* — имя прилагательное, *VERB* — глагол, *CNUM* — количественное числительное, *PART* — причастие;
- множество падежей русского языка $GC = \{N, G, D, A, I, P\}$. Элементы множества *N* — именительный падеж, *G* — родительный падеж, *D* — дательный падеж, *A* — винительный падеж, *I* — творительный падеж, *P* — предложный падеж;
- множество родов русского языка $Genders = \{M, F, N\}$, где *M* — мужской род, *F* — женский род, *N* — средний род;
- множество показателей числа $Numbers = \{SG, PL\}$, где *SG* — единственное число, *PL* — множественное число.

Слово w можно представить в виде списка результатов морфологического анализа, сгруппированных по частям речи. Функция $MA(w)$ задает соответствие между множеством слов и множеством вариантов морфологического разбора:

$$MA{:}w \to (G_1, \ldots, G_2),$$

где $G_i = (GI_1, \ldots, GI_k)$, $GI_j = (pos, gc, gender, number)$, $pos \in PoS$, $gc \in GC$, $gender \in Genders$, $number \in Numbers$, $\forall j \in [1, k-1]$: $pos_j = pos_{j+1}$, GI_j — это конкретный морфологический разбор слова w. Групп морфологических разборов G_i может быть больше одной из-за омонимии в русском языке. Таким образом, результаты морфологического анализа слова w могут быть представлены в разгруппированном виде:

$$MA(w) = (GI_1, \ldots, GI_p).$$

Такое представление результатов морфологического анализа слова мы будем использовать при дальнейшем описании моделей и алгоритмов.

Выделение именных групп

Определение 1. Именная группа — это группа, у которой главное слово — существительное.

Введем ряд вспомогательных конструкций. Подчинительную синтаксическую связь мы будем обозначать в виде пары $(w, RL(w))$, где w — слово, а $RL(w)$ — список подчиненных ему слов. Множество подчинительных синтаксических связей мы будем обозначать R.

На предварительном этапе работы алгоритма для каждого слова w из предложения в множество подчинительных связей R добавляется пустой список () подчиненных слов, т. е. $\forall w$: $R = R \cup (w, ())$.

Входными данными алгоритма выделения именных групп является предложение, состоящее из слов: $Sent = w_1, \ldots, w_n$, где $w_i \in \Sigma^+$, при этом Σ — алфавит русского языка. Алгоритм состоит из трех этапов:

1. Установление подчинительных синтаксических связей в паре (lw, rw), lw — левое слово, rw — правое слово в паре слов. Например, если есть предложение $Sent$, состоящее из слов w_1, w_2, w_3, то можно выделить пару (w_1, w_2), в которой $lw = w_1$, $rw = w_2$.
2. Установление синтаксических связей внутри конструкций с однородными членами.
3. Выделение именных групп.

Ниже приведены основные этапы алгоритма выделения именных групп.

Установление подчинительных синтаксических связей

Подчинительные синтаксические связи устанавливаются в соответствии с описанными ниже тремя правилами:

1. Если $\exists GI_i \in MA(lw)$: $pos_i = NOUN$ и $\exists GI_j \in MA(rw)$: $pos_j = NOUN$ и $gc_j = G$, то добавить отношение подчинения для слова lw: $RL(lw) = RL(lw) \cup (rw)$.
2. Если $\exists GI_i \in MA(lw)$: $pos_i = ADJ$ и $\exists GI_j \in MA(rw)$: $pos_j = NOUN$ и $gc_i = gc_j$, $gen_i = gen_j$ и либо $num_i = num_j$, либо $num_i = SG$, то добавить отношение подчинения для слова rw: $RL(rw) = RL(rw) \cup (lw)$.
3. Если $\exists GI_i \in MA(lw)$: $pos_i = CNUM$ и $\exists GI_j \in MA(rw)$: $pos_j = NOUN$ и $gc_j = G$ и либо $gc_i = N$, либо $gc_i = G$, то добавить отношение подчинения для слова lw: $RL(lw) = RL(lw) \cup (rw)$.

Установление синтаксических связей внутри конструкций с однородными членами

Введем множество связок C. Элементы этого множества в предложении располагаются между однородными членами предложения.

В соответствии с приведенными далее условиями формируется список $S = (s_1, \ldots, s_m)$ однородных существительных:

- существительные разделены связками из множества C;
- существительные имеют совпадающие или пересекающиеся значения падежа, т. е. $\forall s_i \in S$, $\exists s_j \in S$: $gc_i = gc_j$.

Осуществим две последовательные проверки, предназначенные для изменения отношений синтаксического подчинения для слов, входящих в список однородных существительных S.

1. Если последние два существительных s_{n-1}, $s_n \in S$ разделены союзом "и" и $\exists s_l \in RL(s_n)$: $\exists GI_j \in MA(s_l)$: $pos_i = NOUN$, то если $\forall s_k \in RL(s_{n-1})$, $\nexists GI_i \in MA(s_k)$: $pos_i = NOUN$, то $r_{n-1} = (s_{n-1}, RL(s_{n-1}) \cup s_l)$. Если $\exists GI_i \in MA(s_k)$: $pos_i = NOUN$, то завершить эту часть алгоритма и перейти к следующей фазе.
2. Если $\forall s_i \in S$, $\exists GI_i \in MA(s_i)$: $gc_j = G$ и если $\exists r_p \in R$: $s_1 \in RL(w_p)$ и $\exists GI_j \in MA(w_p)$: $pos_j = NOUN$, то $\forall r_q = (w_q, RL(w_q))$, если $s_1 \in RL(w_q)$, то $RL(w_q) = RL(w_q) \cup (s_2, \ldots, s_m)$.

Выделение именных групп

На основе множества подчинительных синтаксических связей R построим ненаправленный граф $G = (V, E)$, $E = E_q \cup E_u$. Вершинами в этом графе являются слова предложения, дуги — подчинительные синтаксические связи. Различают два типа ребер: q-ребра и u-ребра. Если $e_{q_l} = (v_i, v_j)$, то $\exists r_p \in R$: $v_j \in RL(v_i)$, и если $e_{u_l} = (v_i, v_j)$, то $\exists r_p \in R$: $v_j \in RL(v_i)$, $pos_{v_j} = ADJ$.

Поиск именных групп осуществляется с использованием построенного графа. Именная группа — это путь $p = (v_1, \ldots, v_k)$, удовлетворяющий одному из следующих условий:

1. Если $k = 2$, то $e = (v_1, v_2)$, $e \in E_q$.
2. Если $k > 2$, то $\exists e_i \in E_u$ и $\exists e_j \in E_q$.

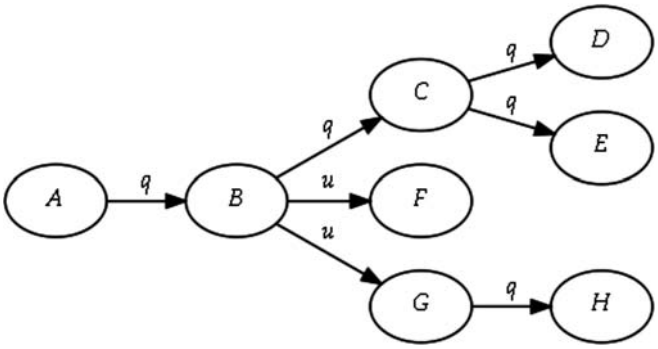


Рис. 1. Пример графа зависимостей

Выделенные в результате работы алгоритма именные группы образуют множество $NG = \{p_1, ..., p_f\}$.

На рис. 1 приведен пример графа подчинительных синтаксических связей с обозначенными *q*- и *u*-ребрами.

Выделение причастных оборотов

Алгоритм выделения причастных оборотов работает с результатами работы алгоритма выделения именных групп, а именно с множеством *NG*. Грамматическая информация именной группы определяется по главному слову именной группы. Алгоритм основывается на том предположении, что причастие имеет такую же модель управления, как и глагол, от которого это причастие образовано.

Введем вспомогательные определения, необходимые в дальнейшем для формулировки алгоритма выделения причастных оборотов.

Определение 2. Ограничение — это кортеж $r = (preposition, gc, type\}$, где *preposition* — это предлог, с которым глагол употребляется в предложении, *gc* — падеж зависимой синтаксемы (именной группы), *type* — тип зависимой синтаксемы (свободная, связанная или обусловленная).

Определение 3. Глагольное управление — это отображение, ставящее в соответствие каждому глаголу список ограничений:

$$VG: v \rightarrow (r_1, ..., r_n).$$

Для некоторых типов причастий, например для страдательных причастий, возникают самостоятельные ограничения, дополняющие список ограничений, полученных причастием от глагола, от которого оно образовано.

Таким образом, для причастий вводится понятие управления причастий.

Определение 4. Управление причастий — это отображение, ставящее в соответствие каждому причастию список ограничений:

$$PG: p \rightarrow (r_1, ..., r_k).$$

Ограничения для причастия *p* получаются из ограничений глагола *v*, от которого рассматриваемое причастие было образовано, и дополнительных ограничений $(r_1, ..., r_l)$, характерных для данного типа причастия:

$$PG(p) = VG(v) \cup (r_1, ..., r_l).$$

Причастный оборот — это причастие и управляемая им именная группа, удовлетворяющая одному из ограничений из списка ограничений для заданного причастия. Поиск зависимой синтаксемы для причастия осуществляется в обе стороны, так как причастие может располагаться как справа, так и слева по отношению к зависимым словам.

Таким образом, мы приходим к алгоритму выделения причастных оборотов: если $\exists w_i: \exists GI_j \in MA(w_i)$: $pos_j = PART$ и $\exists r_p \in PG(w_i)$ и $\exists n_q \in NG$: $prep_p \in n_q$, $gc_p = gc_q$, то именная группа n_q и причастие w_i образуют причастный оборот.

Определение 5. Глагольная группа — это группа, у которой главным словом является глагол или отглагольная часть речи (например, причастие).

То есть в глагольной группе устанавливается связь между глаголом (или причастием) и иными сущностями, которыми могут выступать, в частности, именные группы. Тем самым появляется возможность осуществлять когнитивный анализ текстов на естественном языке.

Пример

Рассмотрим пример применения предложенных моделей и методов к решению задачи когнитивного анализа текстов на естественном языке, а именно — к решению задачи выделения именных групп. Далее в выделенных структурных единицах слова приведены в нормальной форме. На дугах графов, приведенных на рис. 3 и 4, указан тип синтаксического отношения.

Пример 1. Исследование методов морфологического и синтаксического анализа

Граф зависимостей для этого предложения представлен на рис. 2. В соответствии с алгоритмом выделения именных групп были получены следующие именные группы:

- анализ морфологический;
- анализ синтаксический;
- метод анализ морфологический;

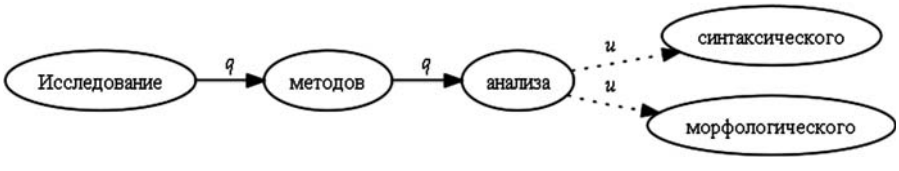


Рис. 2. Граф зависимостей для предложения из примера 1

- метод анализ синтаксический;
- исследование метод анализ;
- исследование метод анализ морфологический;
- исследование метод анализ синтаксический.

Оценка качества

Для оценки качества выделения именных и глагольных групп были использованы данные из Национального корпуса русского языка [6]. Для оценки качества предлагаемых моделей и алгоритмов было выбрано несколько примеров предложений с синтаксической разметкой из синтаксического корпуса.

Пример 2. Рынок автоматизации продуктовых торговых сетей в России

Применив алгоритмы выделения именных групп и причастных оборотов к данному предложению, были получены следующие результаты:
- сеть торговый;
- сеть продуктовый;
- сеть продуктовый торговый;
- сеть продуктовый торговый в;
- автоматизация сеть продуктовый торговый;
- автоматизация сеть продуктовый торговый в;
- рынок автоматизация;
- рынок автоматизация сеть продуктовый торговый;
- рынок автоматизация сеть продуктовый торговый в.

Синтаксическая структура предложения, полученная из синтаксического корпуса НКРЯ [6], приведена на рис. 3. Как можно видеть, именные группы, выделенные нашей системой, полностью согласуются с синтаксической разметкой анализируемого предложения.

Рис. 3. Синтаксическая структура предложения из примера 2

Пример 3. Сотрудники, обслуживающие заведение, входят в категорию риска

Применение предлагаемых моделей и алгоритмов к этому предложению позволяет получить следующие результаты:
- обслуживающий заведение;
- категория риск;
- в категория;
- в категория риск;
- входить в;
- входить в категория;
- входить в категория риск.

Синтаксическая структура предложения, полученная из синтаксического корпуса НКРЯ, приведена на рис. 4. Как можно видеть, выделенные нашей системой именные группы и причастный оборот полностью согласуются с синтаксической разметкой анализируемого предложения.

Из приведенных примеров видно, что предлагаемые модели и алгоритмы способны решать задачу частичного синтаксического анализа и тем самым позволяют решать задачу когнитивного семантического анализа. Особенностью разработанных алгоритмов является их применимость при построении промышленных систем анализа текстовой информации, обусловленная высокой скоростью работы предлагаемых методов.

В работе [7] на научных текстах достигается точность 88 %. Наши эксперименты, проведенные на корпусе новостных сообщений, показали, что предлагаемые алгоритмы способны обнаруживать именные группы с точностью 95 %. Точность выделения причастных оборотов составляет 98 %, что согласуется с результатами предварительного теоретического исследования в области анализа моделей глагольного управления.

Рис. 4. Синтаксическая структура предложения из примера 3

Заключение

В данной работе были рассмотрены математические модели, предназначенные для решения в некотором допустимом для реальных систем обработки текстов приближении широкого класса задач когнитивного анализа текстовой информации. Алгоритмы, сформулированные в терминах предложенных моделей позволяют эффективно и с высокой степенью точности обнаруживать зависимые синтаксические конструкции, что в свою очередь позволяет реализовать эффективный алгоритм синтаксического анализа текстов на русском языке.

Список литературы

1. **Croft W., Cruse D. A.** Cognitive Linguistics (Cambridge Textbooks in Linguistics). Cambridge University Press, 2004. 372 p.
2. **Ножов И. М.** Морфологическая и синтаксическая обработка текста (модели и программы) // Дисс. ... канд. техн. наук. М., 2003.
3. **Jurafsky D., Martin J. H.** Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Prentice Hall, New Jersey, second edition, 2008. 1024 p.
4. **Болховитянов А. В., Гусев С. В., Чеповский А. М.** Модель и база знаний глагольного управления в предложениях на русском языке // Информационные технологии. 2011. № 12. С. 37—39.
5. **Золотова Г. А.** Синтаксический словарь: Репертуар элементарных единиц русского синтаксиса. М.: Едиториал УРСС, 2006. 440 с.
6. **Национальный** корпус русского языка. URL: http://ruscorpora.ru/ (дата обращения: 25.04.2011).
7. **Белоногов Г. Г., Кузнецов Б. А.** Языковые средства автоматизированных информационных систем. М.: Наука, 1997. 248 с.

УДК 004.82

А. А. Левков, канд. техн. наук, докторант,
Уфимский государственный авиационный технический университет, г. Уфа

Интеграция знаний и данных в больших информационных системах

Предлагается способ организации онтологии в виде явно заданной иерархии типизированных концептов, что позволяет структурировать хранимые факты базы знаний на основе реляционной схемы данных и ее процедурных расширений, получаемых с помощью прямого преобразования онтологии без потери импеданса моделей.

Ключевые слова: *семантическая модель, реляционная система, дескрипционная логика, трансляция моделей, иерархически-реляционная схема данных*

Введение

В настоящее время стремительно растет число интеллектуальных информационных систем (ИИС), оперирующих большими объемами данных, основанными на сложных знаниях. Типичной архитектурой таких систем является комбинация онтологического и реляционного подходов, где метаданные сформулированы в виде терминологических аксиом (*TBox*+*RBox*) дескрипционной логики (ДЛ), а данные (*ABox*) хранятся в реляционной СУБД [1]. Использование выражений ДЛ позволяет выполнять операции вывода знаний, а реляционная СУБД обеспечивает работу с хранимой информацией.

Основной сложностью построения таких моделей является трансляция онтологической модели и ДЛ-запросов на реляционный базис [2]. От эффективности преобразований существенно зависит общая эффективность работы всей системы в целом. Задача трансляции усложняется большим числом диалектов ДЛ, каждый из которых определяет свои дополнительные конструкторы и правила работы с онтологической моделью, независимо от сложности реализации данных конструкций на основе реляционного исчисления. В целом, для подобных *sparql*-систем характерна сложная вычислимость и низкое быстродействие [3], связанные либо с прямой трансляцией концептов в реляционные сущности [4], либо с прямой трансляцией реляционных сущностей в концепты ДЛ [5] (в зависимости от первообразной системы).

В данной статье предлагается рассмотреть онтологическую модель в роли системы ограничения целостности реляционной модели. Построение такой интегрированной модели позволит существенно повысить эффективность систем, интегрирующих данные и знания за счет явного хранения *ABox* в соответствии с *TBox*+*RBox* [6].

Ограничение онтологической модели

Одной из основных трудностей реализации *ABox*-хранилища со стороны реляционных СУБД является игнорирование вопроса хранения данных самой онтологической моделью — рассматриваемая как математическая абстракция, онтологическая модель не определяет способов и методов организации *ABox*-составляющей [7].

Для эффективной организации хранения данных необходимо пересмотреть способ описания терминологических аксиом, адаптировав описание таким образом, чтобы совокупность концептов явно отражала информацию в реляционных сущностях, что

позволило бы существенно упростить верификацию целостности и согласованности данных (соответствия *ABox*-выражений *TBox*+*RBox*) и упростить выполнение *ABox*-запросов.

Первоочередной задачей согласования онтологического подхода является разделение выражений *TBox*+*RBox* на два отдельных множества: на множество, определяющее явную структуру хранимых данных, и множество, определяющее дополнительные ограничения целостности. То есть необходимо разделить концепты, являющиеся первичными в системе, и вторичные концепты, определенные на первичных, "уточняющие" их, но не добавляющие новых данных, а определяющие дополнительные семантические правила интерпретации и именования уже существующих данных.

Физический смысл различий между первичными и вторичными концептами заключается в том, что первичные концепты определяют, какие именно классы и объекты с атрибутами составляют область описания онтологии, в то время как вторичные концепты, комбинируя первичные концепты, определяют дополнительные знания об их взаимодействии, но не вносят нового фактологического материала в систему — не определяют новых сущностей и новых областей данных.

Например, в онтологии определены следующие терминологические аксиомы:

$$\text{"Документ"} \sqsubseteq \text{"Абстракт"} \quad (1)$$

$$\text{"Состояние"} \sqsubseteq \text{"Абстракт"} \quad (2)$$

$$\text{"Пользователь"} \sqsubseteq \text{"Абстракт"} \quad (3)$$

$$\text{"Приходный документ"} \sqsubseteq \text{"Документ"} \quad (4)$$

$$\text{"Расходный документ"} \sqsubseteq \text{"Документ"} \quad (5)$$

То есть существуют первичные концепты "Абстракт", "Документ", "Приходный документ", "Расходный документ", "Пользователь" и "Состояние", и концепты "Документ" и "Состояние" связаны ролью "В состоянии", где *ABox*, относящийся к концепту "Состояние", определен как {'Открыт', 'Закрыт'}. Тогда становится возможным ввести вторичный концепт "Открытый документ", который определяется как:

$$\text{"Открытый документ"} \equiv \text{"Документ"} \sqcap \exists\ \text{"В состоянии"}.\{\text{'Открыт'}\} \quad (6)$$

Таким образом, вторичные концепты просто задают "псевдонимы" для составных первичных концептов и удобны в качестве средства инкапсуляции логики, не оказывая влияния на структуру и данные онтологической модели.

В целом, специфика онтологического подхода заключается в моделировании систем реального мира как множества пересекающихся, но в высокой степени самостоятельных "семантик" *TBox*+*RBox*, каждая из которых отражает определенный угол зрения на моделируемую систему. Основным конструктором для описания модели служит конструктор вложенности концептов $\sqsubseteq$. Семантика такого оператора не определена и может изменяться в зависимости от точки зрения составителя описания. Например, в классической онтологии возможно определение аксиомы:

$$\text{"Открытый документ"} \sqsubseteq \text{"Документ"} \quad (7)$$

Данная аксиома является верной, однако очевидно, что выражения (1)—(5) и (7) несут разную семантическую нагрузку, хотя и определяются одним оператором вложенности концептов. Установить это отличие возможно, только проведя полный анализ всего *TBox*+*RBox* (только обнаружив аксиому (6) и подобные ей).

При реализации системы хранения данных наиболее значащей является семантика родово-видового наследования концептов, так как именно она определяет состав атрибутов концепта, а следовательно, и реляционной сущности. Именно данная семантика оператора вложенности концептов позволяет выделить первичные концепты *TBox*+*RBox* и может быть явным образом транслирована в систему взаимосвязанных реляционных сущностей.

Для явного построения таксономии концептов и ролей предлагается ввести в онтологическое описание новый оператор $\prec$, определяющий иерархию наследования концептов ("Предок" $\prec$ "Потомок"), т. е. сформировать собственный диалект ДЛ с явными таксономиями — *ALT*-диалект (*AttributeLanguagewithTaxonomy*). Построение *ALT*-онтологии должно начинаться с формирования таксономии концептов и ролей на основе определения концептов, связанных только данным оператором, без использования конструкторов конъюнкции, дизъюнкции и кванторов в соответствии со следующими правилами.

***ALC*-выражение**	***ALT*-выражение**
$A \doteq B \sqcup C$	$A \prec B,\ A \prec C$
$A \doteq B \sqcap C$	$B \prec A,\ C \prec A$

Опираясь на данные правила, становится возможным на онтологическом уровне сформулировать структурную иерархию концептов, которая может быть транслирована в реляционные сущности, связанные вторичными ключами, и определять состав атрибутов в сущностях.

Вторичные концепты определяются на основе первичных и могут задаваться как с помощью операторов конъюнкции и дизъюнкции, так и кванторов. Таким образом, все выражения с кванторами определяют только вторичные концепты и не могут определять первичные концепты. То есть вторичные концепты определяют правила, которым должна подчиняться реляционная модель и которые не могут быть выражены в виде реляционных ограничении целостности — с помощью ключей.

Примером преобразования терминологических аксиом (1)—(6) по заданным правилам могут служить следующие выражения:

"Документ" $\prec$ "Абстракт" (8)

"Состояние" $\prec$ "Абстракт" (9)

"Пользователь" $\prec$ "Абстракт" (10)

"Приходный документ" $\prec$ "Документ" (11)

"Расходный документ" $\prec$ "Документ" (12)

При таком подходе выражения (8)—(12) и (7) имеют разную формулировку, согласно их разному семантическому смыслу. Таким образом, применение явного оператора наследования в онтологической модели позволяет более точно описывать модель предметной области.

Другой важной особенностью онтологического подхода является необязательность ограничения ролей концептами, что позволяет создавать "всеобщие" роли, которые могут быть применены к любому концепту, в то время как в реальном мире роли связывают только конкретные типы объектов и роли всегда могут быть ограниченными концептами, т. е. для каждой роли должны быть сформулированы выражения типа $\forall R.\top \sqsubseteq C_1$. Важно, что такое ограничение является двусторонним: оно применяется как для предшественников, так и для последователей любой роли т. е. все роли должны иметь инверсную интерпретацию и ограничение $\forall R^{-}.\top \sqsubseteq C_2$. В общем случае, данное ограничение многоместных ролей и определяет ограничение каждого места роли принадлежностью к какому-либо концепту. Таким образом, необходимо ввести понятие ограничение роли, которое должно формулироваться аналогично утверждению об индивидах, но задавать ограничение на каждое место роли, имеющее в общем случае следующий вид: $R[C_1, C_2, ..., C_n]$. В данном случае в роли концептов-ограничителей могут выступать как первичные, так и вторичные концепты.

В качестве примера можно использовать концепты "Документ", "Состояние" и "Пользователь" и роль "Находится в состоянии". Как было показано выше, роль "Находится в состоянии" отражает связь именно состояния документа и не должна использоваться для отражения состояния пользователя: такое использование будет ошибочным. В данном случае при определении роли должно быть использовано ограничение:

"Находится в состоянии"["Документ", "Состояние"] (13)

С помощью подобных выражений становится возможным определять ограничения на уровне каждого места роли и не требуется дополнительных введений инверсных ролей и их сложных ограничений.

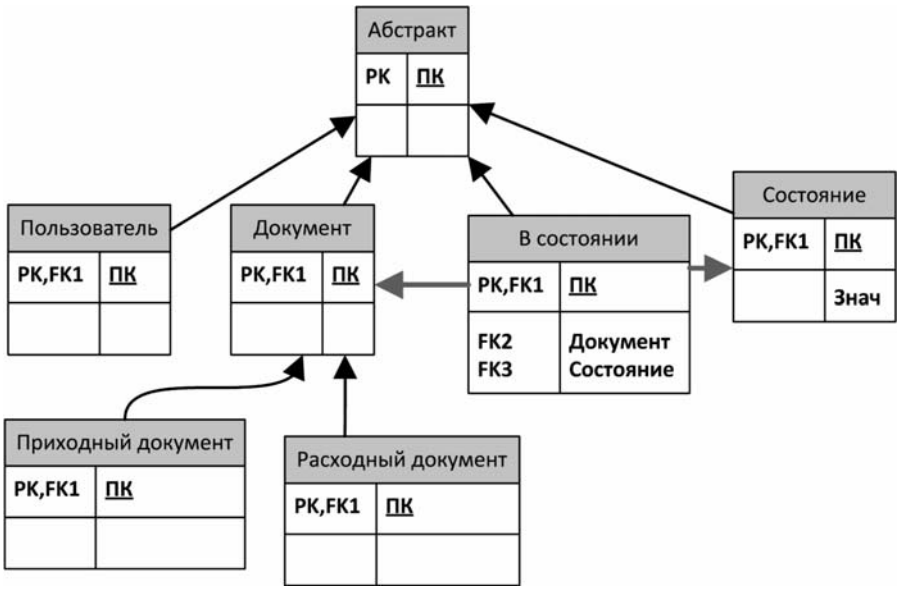
Таким образом, в данном разделе предложено введение дополнительных ограничений на онтологической модели, которые упрощают реализацию системы хранения и верификации *ABox*-утверждений и в виде реляционных структур и операторов.

Ограничение реляционной модели

Для организации эффективного хранения данных (которое подразумевает не только само хранение, но и операции по манипуляции с ними — выборку, добавление, изменение и удаление) необходимо обеспечить наиболее полное соответствие реляционной модели онтологическому описанию. Со стороны онтологической модели для этих целей предлагается ввести разделение на первичные и вторичные концепты и организовать таксономию концептов, определяющую их родово-видовую иерархию [8]. Такое построение модели наиболее эффективно с точки зрения хранения данных — оно позволяет избежать дублирования информации при организации ее хранения и может быть с минимальными усилиями перенесено на реляционную модель без нарушения правил нормализации реляционных сущностей.

При осуществлении трансляции предложенной онтологической модели в реляционное пространство необходимо руководствоваться следующими правилами:

- каждый первичный концепт преобразуется в реляционную сущность;
- операторы наследования преобразуются во вторичные ключи, связывающие отношения по первичным ключам (в результате образуются связи 1-к-0..1);
- роли интерпретируются как особый вид концептов, осуществляющих их связь (так как сами роли также являются концептами, то становится возможным определять роли для ролей, т. е. организовывать не только иерархические роли, но и составные аксиомы вложенности ролей);
- каждое место роли транслируется в атрибут реляционной сущности. Так как в предлагаемой онтологической модели все роли имеют ограничения в виде концептов, то эти ограничения могут быть заданы явно в виде вторичных ключей, направленных с мест роли на соответствующие ограничивающим концептам сущности, в том случае, если ограничивающие концепт сущности являются первичными концептами;
- в случае, если концепты являются вторичными, необходимо наложить ограничение на уровне наименьшего потомка из первичных концептов — такое ограничение будет нестрогим и не гарантирует полного выполнения ограничения, в связи с чем необходимо воспользоваться процедурным расширением реляционных СУБД, определив на данном отношении-роли и на отношении



Реляционная структура иерархической онтологии

концепта-ограничителя триггеры, оперирующие данными в представлениях и позволяющие обеспечить полное выполнение ограничения.

Общий вид такой реляционной структуры, соответствующей выражениям (2)—(6), (8), представлены на рисунке.

Для синтеза полного концепта по иерархии реляционных сущностей необходимо руководствоваться рекурсивными правилами по работе с иерархически реляционными моделями [9]:

$$E^{\text{full}} = \begin{cases} \text{если } \nexists\, E_{\text{пред}}, \text{ то } E \\ E\frac{pk}{\theta} E_{\text{пред}}^{\text{full}} \end{cases}; \qquad (14)$$

$$A_{E^{\text{full}}} = \begin{cases} \text{если } \nexists\, E_{\text{пред}}, \text{ то } A_E \\ A_E \cup A_{E_{\text{пред}}^{\text{full}}} \end{cases}. \qquad (15)$$

В практике реляционного проектирования подобный алгоритм реализуется с помощью определения представлений (*view*), соединяющих все отношения в ветви рекурсии по первичному ключу. Так, для сущности "Приходный документ" согласно формулам (14), (15) можно определить следующее представление:

```
createview "Приходный документ" as
select * from "Абстракт"
innerjoin "Документ" on
"Документ".ПК="Абстракт".ПК
innerjoin "Приходный документ" on
"Приходный документ".ПК="В Документ".ПК
```

Все вторичные сущности, определенные в онтологии, транслируются в представления напрямую. Представления позволяют проводить операции по извлечению данных, но не обеспечивают целостности информации (на представлениях невозможно объявлять первичные или вторичные ключи). Таким образом, вторичные сущности в реляционной модели, как и в онтологической, не определяют нового фактологического материала, а только оперируют уже существующими данными, комбинируя и интерпретируя этот материал. Следующие реляционные выражения соответствуют выражению (6) онтологической модели:

```
createview "Открытый документ" as
select * from "Документ"
innerjoin "В состоянии" on
"Документ".ПК="В состоянии".Документ
innerjoin "Состояние" on
"Состояние".ПК="В состоянии". Состояние
where "Состояние".Знач='Открытый'
```

Современные реляционные СУБД позволяют явно реализовывать транзитивные роли на основе расширенных реляционных выражений. Для этого необходимо использовать представления, основанные на рекурсивных табличных выражениях, что позволяет обеспечить вывод онтологических выражений $Tr(R)$. Если в онтологии задана транзитивная роль "Часть-Целое", то согласно правилам трансляции в реляционной модели создается отношение с двумя атрибутами. Раскрытие транзитивности может быть выражено с помощью следующих реляционных выражений:

```
create view "Часть-Целое" as
with _CTE(Часть, Целое) as (
select Часть, Целоеfrom "Часть-Целое"
union all
select b.Часть, a.Целоеfrom "Часть-Целое" a
inner join _CTE b on b.Целое=a.Часть)
select Часть, Целое from _CTE
```

Ограничения (в том числе качественные) кардинальности ролей вида $\leqslant nR$ ($\leqslant nR.C$) могут быть реализованы с помощью триггерной логики и опе-

раций агрегации реляционной системы. Если в онтологической модели задано ограничение

$$\leqslant 2 \text{ "В состоянии". "Документ",}$$

определяющее не более двух одновременных состояний объекта класса "Документ", то такое ограничение легко может быть реализовано следующим процедурно-реляционным выражением:

```
create trigger trg1 on "Всостоянии" after
insert,
update as
if exists (
select Документ, count(*) cnt
from "В состоянии" a
where exists (
     select * from inserted b
     where a. Документ=b. Документ)
group by Документ
having count(*) > 2)
rollback
```

Таким образом, предложенные правила трансляции онтологических выражений в реляционные структуры и правила (для реляционных СУБД, обладающих поддержкой процедурных расширений), позволяет реализовать автоматический контроль *ABox*-выражений для *SHIOQ(D)* онтологий, т. е. онтологий, позволяющих задавать транзитивные роли (*S*), иерархии ролей (*H*), инверсные роли (*I*), номиналы (*O*), качественные ограничения кардинальности ролей (*Q*) и домены (*D*).

Заключение

В данной статье предложена методика трансляции онтологического описания предметной области в реляционную схему данных и ее процедурные расширения. Для осуществления данного преобразования предложено введение нового оператора в онтологическую модель, который позволяет строить явные иерархии концептов, что существенно упрощает следующее построение реляционной системы. Также предложено ввести обязательное явное ограничение ролей концептами в целях более точного соответствия онтологии реляционной системе.

Также в данной статье предложены правила преобразования онтологического описания в реляционную систему. Реляционная система, построенная по данным правилам, позволяет сохранять данные *ABox* без выполнения сложных процедур по верификации заносимой информации — эта нагрузка ложится на реляционную СУБД, обеспечивающую проверку вторичных ключей и выполнение триггеров на отношениях. Кроме того, реляционная система обеспечивает эффективное выполнение запросов на получение информации. При реализации изменений в *TBox*+*RBox* все они должны отражаться на реляционной модели и, следовательно, на хранимых в этой модели данных, что позволяет обеспечивать проверку соответствия *ABox* и *TBox*+*RBox*.

Применение предложенного комплексного подхода к описанию баз знаний позволяет существенно упростить построение систем хранения фактов для сложных и больших онтологических моделей, используя для этого эффективный базис реляционных моделей данных.

Список литературы

1. **Кузнецов И. П.** Организация семантико-ориентированных систем поиска и обработки информации // Системы и средства информатики. 2006. № 3.

2. **Roger M., Simonet A., Simonet M.** Bringing Together Description Logics and Database in an Object Oriented Model // In Proc. of DEXA'2002. C. 504—513.

3. **Бездушный А. А.** Математическая модель системы интеграции данных на основе онтологии // Вестник НГУ (Информационные технологии). 2008. № 2. С. 15—40.

4. **Hao G., Ma Sh., Sui Y., Lu J.** An Unified Dynamic Description Logic Model for Databases: Relational Data, Relational Operations and Queries // In Proc. Tutorials, posters, panels and industrial contributions at the 26th International Conference on Conceptual Modeling, Auckland, New Zealand. 2007. N 83. C. 121—126.

5. **Fisher M., Dean M., Joiner G.** Use of OWL and SWRL for Semantic Relational Database Translation // Fourth International Workshop OWL: Experiences and Directions, Washington, DC, 2008.

6. **Liu H., Lutz C., Milici M., Wolter F.** Updating Description Logic ABoxes // Proc. of Tenth International Conference on Principles of Knowledge Representation and Reasoning, Lake District of the United Kingdom, June 2—5, 2006.

7. **OWL** Web Ontology Language. URL: http://www.w3.org/TR/owl-features.

8. **Nutt W.** Inheritance hierarchies and view hierarchies // British National Conference on Databases: Advances in Databases, Rutherford Appleton Laboratory, 2001. N 3.

9. **Ильясов Б. Г., Левков А. А.** Структурная оптимизация реляционных моделей сложных иерархических систем // Информационные технологии. 2011. № 3. С. 50—54.

УДК 004.912

М. В. Бочков, д-р техн. наук, проф.,
e-mail: mrboykov@rambler.ru,
НОУ ДПО "Центр предпринимательских рисков", г. Санкт-Петербург,
П. Н. Бойков, инженер,
УССИ ФСО России в СЗФО, г. Санкт-Петербург

Способ автоматического рубрицирования неструктурированной информации сети Интернет

Предложен способ классификации гипертекстовых документов, полученных из неструктурированных источников информации, на основе автоматического выделения полезной информации и удаления служебных данных html-страниц.

Ключевые слова: *обработка html-страниц, сингулярное выражение, латентно-семантический анализ*

Разнообразие запросов, формируемых пользователем сети Интернет, связано с обращением по ссылкам, предоставляемым поисковыми системами, к значительному числу источников информации (web-сайтов), имеющим разнообразную внутреннюю структуру. Следствием этого является необходимость ручного отбора пользователем релевантной информации по большому числу ссылок, т. е. выполнение наиболее трудоемких интеллектуальных операций по отбору и интерпретации содержимого страниц.

Для преодоления описанной проблемы авторами предложен способ автоматического рубрицирования гипертекстовых документов, включающий следующие этапы:

- формирование поискового запроса и сохранение его результатов в виде массива гипертекстовых страниц;
- анализ гипертекстовых документов массива и выделение полезной информации для проведения процедуры автоматической классификации на рубрики;
- классификация документов в соответствии с предварительно формируемым множеством рубрик.

Обобщенная функциональная схема процесса рубрицирования представлена на рис. 1.

Первые два этапа сводятся к применению стандартных процедур, реализованных в современных поисковых системах.

Новизну и практическую значимость рассматриваемого способа определяет оригинальная процедура выделения полезной информации из html-страниц, в соответствии с которой анализу подвергается не вся гипертекстовая страница, а только информативные ее части, что существенно влияет на корректность результатов рубрицирования. Это связано с тем, что в заголовках или некоторых блоках html-страницы может содержаться информация, которая при дальнейшем анализе негативно повлияет на результаты рубрицирования. К такой информации, например, можно отнести ссылки на другие информационные ресурсы, в названии которых встречаются использованные в запросе ключевые слова. Предлагаемая процедура позволяет извлечь из гипертекстового документа только те данные, которые непосредственно отвечают сформированному пользователем запросу.

Основным содержанием процедуры выделения является разбиение документа на большое число частей (строк, абзацев и т. п.) и расчет для каждой из них число html-тегов. Исследования показывают, что число html-тегов в частях документа с осмысленным текстом существенно меньше, чем в частях, содержащих служебную информацию.

Массив документов, используемых при проведении эксперимента, формировался из произвольно выбираемых статей с сайтов новостных порталов (rbc.ru, news.yandex.ru, newsland.ru и др.). Текст каждой статьи анализировался построчно с подсчетом для каждой из строк числа html-тегов и общего числа символов в строке. На графике (рис. 2, см. вторую сторону обложки) красная линия отражает длину строки в символах, а синяя — число html-тегов.

Как видно из графика, с некоторой периодичностью появляются экстремумы.

Детальный анализ кода html-страниц показывает, что значительная доля html-тегов направлена на защиту от автоматического сбора (противодействия парсерам). В целях улучшения результатов анализа данную категорию следует удалить, предварительно обработав html-код. После обработки результа-

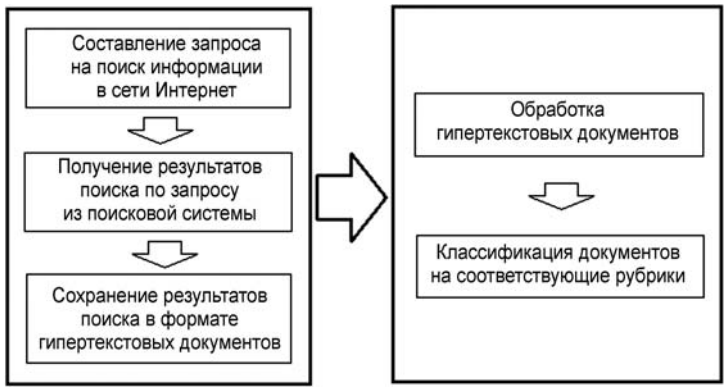


Рис. 1. Обобщенная функциональная схема сбора, обработки и анализа информации из открытых источников в сети Интернет

ты анализа структуры html-страниц становятся более очевидными и приобретают вид, представленный на рис. 3 (см. вторую сторону обложки).

Как видно из графика, существует явно выраженный экстремум, коррелированный с полезной информацией. Кроме этого на графике явно выделены группы служебной информации, способные негативно повлиять на процедуру рубрицирования и подлежащие удалению.

Результаты проведенного исследования показывают, что простейшим способом выделения полезной информации является выбор строк, в которых отношение длины html-разметки к длине строки меньше некоторого порогового значения.

Однако рассмотренный подход имеет два существенных недостатка:

- после удаления заголовков html-страницы в тексте все еще может сохраниться некоторая служебная информация;
- в процессе выделения полезной информации случайно может быть отброшена часть строк реального текста, в которых оказалось много html-тегов или сама строка оказалась слишком короткой.

Улучшение процедуры выделения может быть основано на применении методов машинного обучения, в частности нейронных сетей.

Для реализации процедуры автоматического рубрицирования документов использован латентно-семантический анализ (LSA) [3], представляющий собой метод обработки информации на естественном языке. В соответствии с LSA анализу подвергается взаимосвязь между коллекцией документов и содержащимися в них терминами, сопоставляемыми с некоторыми тематическими рубриками. Данный вид анализа используется крупнейшими компаниями, предоставляющими услуги поиска данных в сети Интернет.

Традиционные процедуры классификации документов предполагают формирование словаря терминов для каждой из рубрик, т. е. выбор слов, которые встречаются исключительно в документах данной тематики и используются для их классификации. Очевидная проблема такого подхода — необходимость включения в словарь всех возможных слов и разрешения коллизий при наличии слов, принадлежащих различным рубрикам. Дополнительную сложность представляют омонимы (слова, имеющие множество значений).

В отличие от них, LSA отображает документы и отдельные слова в так называемое "семантическое пространство", в котором и проводятся все дальнейшие сравнения.

При этом делаются следующие предположения:

- документы — это просто набор слов. Порядок слов в документах игнорируется. Важно только то, сколько раз то или иное слово встречается в документе;
- семантическое значение документа определяется набором слов, которые, как правило, идут вместе. Например, в биржевых сводках, часто встречаются слова: "фонд", "акция", "доллар";
- каждое слово имеет единственное значение. Это, безусловно, сильное упрощение, но именно оно делает проблему разрешимой.

Для реализации процедуры рубрицирования в соответствии с LSA необходимо провести трехэтапную предварительную обработку документов.

На первом этапе необходимо удалить из каждого анализируемого документа стоп-символы. Под стоп-символами понимаются части речи, которые не несут смысловой нагрузки (союзы, частицы, предлоги и др.).

На втором этапе предлагается использовать операцию стемминга — удаления окончаний и выделение основной формы каждого слова.

Одним из наиболее эффективных способов реализации стемминга является алгоритм Мартина Портера [2], в соответствии с которым выделение основы слова осуществляется путем его преобразования согласно определенным правилам.

На заключительном этапе для упрощения математических вычислений из текста удаляются слова, встречающиеся в единичном экземпляре, а все оставшиеся подвергаются индексированию.

На первом шаге процедуры рубрицирования необходимо составить частотную матрицу индексируемых слов M. В этой матрице строки соответствуют индексированным словам, а столбцы — документам. В каждой ячейке матрицы указано, какое число слов встречается в соответствующем документе (см. таблицу), где S_1, ..., S_n — индексируемые слова, D_1, ..., D_m — документы для анализа, k_{11}, ..., k_{nm} — число слов.

На следующем шаге проводится сингулярное разложение полученной матрицы. Сингулярное разложение (англ. *singular value decomposition*, *SVD*) — разложение прямоугольной вещественной или комплексной матрицы, применяющееся во многих областях прикладной математики [4].

В соответствии с процедурой сингулярного разложения исходную матрицу M представляем в виде:

$$M = UWV,$$

где U и V — ортогональные матрицы, а W — диагональная матрица.

Частотная матрица индексируемых слов

Индексируемые слова	D_1	D_2	D_m
S_1	k_{11}	k_{12}	k_{1m}
S_2	k_{12}	k_{22}	k_{2m}
...	...	...	...
S_n	k_{n1}	k_{n2}	k_{nm}

Диагональные элементы матрицы W, называемые сингулярными числами, упорядочены в порядке убывания.

Главное достоинство сингулярного разложения состоит в том, что оно позволяет выделять ключевые составляющие матрицы и игнорировать шумы. Согласно простым правилам произведения матриц, столбцы и строки, соответствующие меньшим сингулярным значениям, дают наименьший вклад в итоговое произведение. Например, мы можем отбросить последние столбцы матрицы U и последние строки матрицы V, оставив только первые два. Важно, что при этом гарантируется оптимальность полученного произведения. Разложение такого вида называют двумерным сингулярным разложением. Данные в этих столбцах будут являться координатами на декартовой системе координат. Соответственно, удаленность того или другого документа означает принадлежность его к той или иной тематике. Введя некоторый числовой порог расстояния, можно классифицировать документы по различным тематикам.

На рис. 4 (см. вторую сторону обложки) представлен пример рубрицирования документов в соответствии с предложенным способом.

Практическое применение рассмотренных в статье процедур в первую очередь связано с разработкой информационно-аналитических систем, ориентированных на анализ и визуализацию текстовых ресурсов сети Интернет.

Список литературы

1. **Чубукова И. А.** Основы информационных технологий. М.: БИНОМ, 2006. 382 с.
2. **Некрестьянов И. А.** Латентно-семантический анализ. СПб.: Наука, 2009.
3. **Kleinberg J. M.** Authoritative sources in a hyperlink environment // In Processing of ACM-SIAM Symposium on Discrete Algorithms, 1998. N 46(5). 604—632 p.
4. **Тархов Д. А.** Нейронные сети. Модели и алгоритмы: Справочник. М.: Радиотехника, 2005.
5. **Ландэ Д. В.** Основы интеграции информационных потоков. К.: Инжиниринг, 2006. 240 с.

ПАМЯТИ КОЛЛЕГИ И ТОВАРИЩА

БАТИЩЕВ
Дмитрий Иванович
(29.09.1941—10.01.2012)

Редколлегия журнала «Информационные технологии» понесла тяжелую утрату... На 71-м году ушел из жизни известный ученый, блестящий преподаватель, талантливый организатор и замечательный человек — **Дмитрий Иванович Батищев**.

Невосполнимую утрату понесла отечественная наука в лице заведующего кафедрой информатики и автоматизации научных исследований Нижегородского государственного университета им. Н.И. Лобачевского (ННГУ), Заслуженного деятеля науки РФ, доктора технических наук, одного из основателей современной науки о решении проектных задач с помощью средств вычислительной техники.

Д. И. Батищев в 1963 году закончил радиофизический факультет Горьковского государственного университета. В 1968 году защитил диссертацию на соискание степени кандидата физико-математических наук по специальности «Вычислительная математика», в 1976 году — диссертацию на соискание степени доктора технических наук по специальности «Автоматизация проектно-конструкторских работ и технологической подготовки производства». Работал в должности доцента на кафедре «Математическая логика и алгебра», профессора на кафедре «Математическое обеспечение ЭВМ», с 1986 года — заведующий кафедрой информатики и автоматизации научных исследований факультета вычислительной математики и кибернетики ННГУ. Заслуженный профессор ННГУ, заместитель председателя докторского совета ННГУ, действительный член Российской академии естественных наук, Международной академии информатизации, Международной академии открытого образования. Награжден медалями «За доблестный труд», «За трудовое отличие» и нагрудным знаком «За отличные успехи в работе». Основные направления научных исследований: математические модели и методы принятия оптимально-компромиссных решений в организационно-технических системах, новые информационные технологии для обеспечения процессов принятия решений в системах автоматизации проектирования и управления. Д. И. Батищев — основатель и руководитель ведущей научно-педагогической школы «Оптимизация в системах автоматизации проектирования и автоматизированных систем управления», научный руководитель 25 кандидатов наук и научный консультант 9 докторов наук. Опубликовано более 200 научных работ, включая учебник и 5 монографий: «Поисковые методы оптимального проектирования», «Современное состояние теории исследования операций» и др.

Профессора Д. И. Батищева отличали широкий научный кругозор, постоянное стремление двигаться вперед, забота о повышении научной квалификации сотрудников кафедры, доброжелательность, отзывчивость.

Редколлегия и редакция журнала «Информационные технологии» выражают глубокое соболезнование родным и близким покойного. Память о Дмитрии Ивановиче Батищеве навсегда сохранится в наших сердцах.

УДК 681.51; 621.391; 519.21

А. В. Меркушева, канд. техн. наук,
инж.-исследователь,
Г. Ф. Малыхина, д-р техн. наук, проф.,
В. М. Малыхин, канд. техн. наук, доц.,
Санкт-Петербургский государственный
политехнический университет,
e-mail: v_m_malykhin@mail.ru

Структуры, методы и алгоритмы адаптивной фильтрации нестационарных сигналов

Рассмотрены методы, алгоритмы и вычислительные схемы для создания системы адаптивных фильтров, ориентированных на управление линейными нестационарными системами и на их идентификацию. Даны элементы математической основы построения адаптивных фильтров с различной сложностью и показателями качества функционирования. Анализируются варианты адаптивных фильтров с ускоренной реализацией.

Ключевые слова: *системы, управление, идентификация, нестационарные сигналы, фильтрация, методы, алгоритмы, структуры вычислительных схем, вычислительная сложность*

Идентификация неизвестной системы является одной из центральных задач в различных областях приложений, таких как управление, выравнивание каналов, подавление эха в сетях связи (в том числе для телеконференций), обработка геофизических сигналов, восстановление формы сигналов при регистрации их смеси (мгновенной линейной или в форме многоканальной свертки). Идентификация является процедурой определения неизвестной модели технической системы. Основой идентификации служат экспериментальные данные: желаемое соотношение сигналов вход — выход и приемлемый выбор функции стоимости ошибки, которая оптимизируется относительно параметров неизвестной модели системы. Адаптивная идентификация относится, в частности, к процедуре, в которой при поступлении новой пары измерений сигналов вход — выход получается дополнительная информация относительно модели системы, что позволяет обновлять в реальном времени работу адаптивного алгоритма идентификации.

В качестве модели целесообразно принять линейную систему с конечной импульсной функцией (КИФ). Иначе говоря, класс моделей, для которого рассматривается оптимальная адаптивная фильтрация, предусматривает, что каждая выходная величина системы определяется взвешенной комбинацией фиксированного конечного числа (M) предшествующих значений входного сигнала системы.

Качество функционирования адаптивного алгоритма (АА) определяется рядом показателей: точностью получаемого решения относительно теоретически ожидаемой; скоростью сходимости; способностью отслеживать изменяющуюся статистику сигналов системы; вычислительной сложностью алгоритма; робастностью к накоплению ошибок округления. В том случае, когда используются многопроцессорные вычислительные системы, вопросы параллельного или конвейерного потока данных также относятся к показателям функционирования АА фильтрации.

Создано достаточно много адаптивных алгоритмов с функцией стоимости на основе наименьших средних квадратов (НСК) ошибки и рекурсивных схем оценки НСК. Причем, рекурсивные схемы оценки НСК (Р_НСК) являются предпочтительными.

В статье представлены с единой точки зрения и систематизированы достаточно широко используемые адаптивные алгоритмы НСК, Р_НСК, алгоритмы НСК и Р_НСК в области преобразования, РНСК по схеме квази-Ньютона и по ускоренной схеме Ньютона, выполняемые блочно или с блочной аппроксимацией. Все алгоритмы рассмотрены как подвиды общей рекурсии, что облегчает их анализ и одновременно обнаруживает сходство между различными схемами алгоритмов.

Фильтры Винера. Алгоритмы адаптивной фильтрации и идентификации систем реализуют оценку набора параметров модели. Параметризация линейной системы (ЛС) это немалый раздел моделирования с широкой областью приложений, причем наиболее часто ЛС имеют структуру с КИФ. Такое ограничение либо свойственно моделируемой системе, либо используется для упрощения задачи оценивания или уменьшения вычислительной нагрузки алгоритма в реальном времени (РВ).

Модель системы описывается дифференциальным уравнением:

$$y(n) = \sum_{i=1}^{M} c_i^0 x(n - i + 1) + \eta(n), \quad (1)$$

где $x(n)$ и $y(n)$ — входной и выходной сигналы; c_i^0 $(i = 1, 2, ..., M)$ — коэффициенты фильтра; $\eta(n)$ — сигнал возмущения. Уравнение (1) компактно представляется в виде:

$$y(n) = \mathbf{x}_M^T(n)\mathbf{c}_M^0 + \eta(n), \quad (2)$$

где $\mathbf{x}_M(n) = [x(n)x(n - 1)...x(n - M + 1)]^T$ — регрессор, или вектор данных размерности $M \times 1$:

$$\mathbf{c}_M^0 = [c_1^0, c_2^0, ..., c_M^0]^T, \quad (3)$$

где $\mathbf{c}_M^0$ — вектор коэффициентов фильтра размерности $M \times 1$.

Фильтр оценки для модели системы (1) определен соотношением

$$\hat{y} = \mathbf{x}_M^T(n)\mathbf{c}_M, \quad (4)$$

где $\mathbf{c}_M = [c_1, c_2, ..., c_M]^T$ — оценка параметров системы $\mathbf{c}_M^0$. В этом случае ошибка оценки выражается как разность между измеренным и предсказанным значениями выхода системы:

$$e(n) = y(n) - \hat{y}(n) = y(n) - \mathbf{x}_M^T(n)\mathbf{c}_M. \quad (5)$$

Идентификация системы с КИФ реализуется структурой, в которой по заданной последовательности входного сигнала и желаемого измеренного сигнала на выходе оцениваются оптимальные параметры системы с КИФ модели (4) путем минимизации критерия средней квадратичной ошибки (СКО):

$$V(\mathbf{c}_M) = E[e^2(n)] = E[y(n) - \hat{y}(n))^2], \quad (6)$$

где E — оператор математического ожидания. Таким образом,

$$\mathbf{c}_M = \arg\min_{\mathbf{c}_M} V(\mathbf{c}_M) =$$

$$= \arg\min_{\mathbf{c}_M} E[(y(n) - \mathbf{x}_M^T(n)\mathbf{c}_M)^2]. \quad (7)$$

Функция стоимости (6) — это квадратичный функционал, имеющий форму

$$V(\mathbf{c}_M) = E[y^2(n)] + \mathbf{c}_M^T\mathbf{R}_M\mathbf{c}_M - 2\mathbf{d}_M^T\mathbf{c}_M, \quad (8)$$

где $\mathbf{R}_M$ — автокорреляционная матрица входного сигнала $x(n)$; $\mathbf{d}_M$ — кросскорреляционный вектор между входным сигналом $x(n)$ и желаемым сигналом $y(n)$ на выходе системы:

$$\mathbf{R}_M = E[\mathbf{x}_M(n)\mathbf{x}_M^T], \ \mathbf{d}_M = E[\mathbf{x}_M(n)y(n)]. \quad (9)$$

Минимизация функции стоимости (6) относительно вектора коэффициентов фильтра осуществляется заданием градиента для $V(\mathbf{c}_M)$ равным нулю: $\nabla V(\mathbf{c}_M) = 0 \Rightarrow \mathbf{R}_M\mathbf{c}_M - \mathbf{d}_M = 0$.

Это условие дает набор линейных уравнений, так называемых нормальных уравнений:

$$\mathbf{R}_M\mathbf{c}_M = \mathbf{d}_M. \quad (10)$$

Минимизация СКО достигается выполнением соотношения

$$E_M^{\min} = E[y^2(n)] - \mathbf{d}_M^T\mathbf{c}_M. \quad (11)$$

(При этом линейное предсказание может рассматриваться как особый случай фильтрации на основе линейного фильтра.) В достаточно общем случае принимается, что сигнал $x(n)$ описывается авторегрессионной (АР) моделью:

$$\sum_{i=0}^{M} a_i^0 x(n - i) = e(n), \quad (12)$$

где a_i^0 — набор, равный единице ($a_i^0 = 1$); $e(n)$ — сигнал шума, ведущий АР модель. Предсказание вперед для системы (6) определяется соотношением:

$$\hat{x}(n) = -\mathbf{x}_M^T(\mathrm{n})\mathbf{a}_M,$$

где $\mathbf{a}_M = [a_1, a_2, ..., a_M]^T$ — вектор размерности $M \times 1$, компоненты которого — коэффициенты предсказания вперед. В этом случае оптимальное предсказание вперед (в смысле Винера) получается как решение нормальных уравнений:

$$\mathbf{R}_M\mathbf{a}_M = -\mathbf{r}_M^f, \quad (13)$$

где $\mathbf{r}_M^f = E[\mathbf{x}_M(n - 1)x(n)]$. Таким же образом для системы (12) трактуется предсказание назад.

Применение для решения (10) и (13) любого прямого или итеративного метода может быть эффективным, если заранее известна статистика входного и желаемого выходного сигналов, $x(n)$ и $y(n)$. Если известны только измерения, то необходимо использовать методы стохастической аппроксимации (СА).

Методы стохастической аппроксимации. В типовой задаче СА [1] требуется минимизировать $V(\mathbf{c}_M) = E[Q(\mathbf{c}_M, \eta)]$, где $\mathbf{c}_M \in \mathbf{R}^M$ — искомый оптимальный параметр, а $Q(\mathbf{c}_M, \eta)$ — функция, распределение которой неизвестно. Если принять эргодичность, то для решения задачи стохастической оптимизации предложено два направления СА: использование нерекурсивного и рекурсивного мето-

дов. В первом случае используются данные отсчетов случайной переменной и аппроксимация в виде минимизации

$$V_N(\mathbf{c}_M) = \frac{1}{N}\sum_{n=0}^{N}(Q(\mathbf{c}_M, \eta(n)). \qquad (14)$$

Методы рекурсивной СА являются результатом соответствующей модификации различных детерминированных итеративных алгоритмов оптимизации, так что задачи стохастической оптимизации преобразуются в детерминированные. Решение (14) используется, чтобы аппроксимировать решение первоначальной задачи минимизации $V(\mathbf{c}_M) = E[Q(\mathbf{c}_M, \eta)]$. Применение метода нерекурсивной СА для задачи идентификации системы порождает семейство уравнений оценок НСК:

$$\mathbf{c}_M(N) = \arg\min_{c_M}\frac{1}{N}\sum_{n=0}^{N}[y(n) - \mathbf{x}_M^T(n)\mathbf{c}_M]^2. \qquad (15)$$

Метод нерекурсивной СА использует алгоритм на основе обработки выборки, т. е. предварительно должно быть собрано и храниться большое количество данных. Это главный недостаток для многих приложений, для которых требуется выполнение алгоритма в реальном времени (РВ). Способ преодоления этой трудности — использование схемы рекурсивной СА, в которой оценка оптимальных параметров обновляется каждый раз при поступлении новых данных. Методы рекурсивной СА являются результатом модификации различных детерминированных итеративных алгоритмов оптимизации. Для алгоритма итеративной детерминированной оптимизации требуется знание функции стоимости $V(\mathbf{c}_M)$ (возможно, ее градиента $\nabla V(\mathbf{c}_M)$ или матрицы Гесса (Гессиана) $\nabla^2 V(\mathbf{c}_M)$[1].

Наиболее простой метод детерминированной оптимизации дает алгоритм градиентного спуска. Его рекурсивная реализация имеет вид

$$\mathbf{c}_M^i = \mathbf{c}_M^{i-1} + \mu_i \mathbf{v}_M^i. \qquad (16)$$

На i-м шаге итерации обновление оценки выполняется вдоль направления $\mathbf{v}_M^i$. Переменная μ_i регулирует эффект обновления на текущем значении оценки[2]. Наиболее общий выбор $\mathbf{v}_M^i$ — это направление, определенное отрицательным градиентом функции стоимости (ФС): $\mathbf{v}_M^i = -\nabla V(\mathbf{c}_M^{i-1})$. В результате получается так называемый градиентный алгоритм (алгоритм крутого спуска):

$$\mathbf{c}_M^i = \mathbf{c}_M^{i-1} - \mu_i \nabla V(\mathbf{c}_M^{i-1}). \qquad (17)$$

Улучшение этого алгоритма осуществляется методом Ньютона—Рафсона с хорошо подобранной весовой матрицей $\mathbf{W}_M^i$:

$$\mathbf{c}_M^i = \mathbf{c}_M^{i-1} - \mu_i \mathbf{W}_M^i \nabla V(\mathbf{c}_M^{i-1}). \qquad (18)$$

В простейшей форме алгоритма Ньютона—Рафсона в качестве весовой матрицы используется обратный Гессиан $[\nabla^2 V(\mathbf{c}_M^{i-1})]^{-1}$: $\mathbf{c}_M^i = \mathbf{c}_M^{i-1} - \mu_i[\nabla^2 V(\mathbf{c}_M^{i-1})]^{-1}\nabla V(\mathbf{c}_M^{i-1})$.

В случае, когда ФС имеет квадратичную форму (как в (8)), это заставляет корректировать направление на минимальную точку. Значение шага μ_i может быть фиксированным (обычно полагают $\mu_i = 1$) или его оптимальное значение оценивается линейным поиском. Если Гессиан не является положительно определенным или обратимым, то применяют другие виды взвешивания. Среди них метод Левенберга—Марквардта, для которого итеративный алгоритм имеет вид: $\mathbf{c}_M^i = \mathbf{c}_M^{i-1} - \mu_i[\nabla^2 V(\mathbf{c}_M^{i-1}) + \delta\mathbf{I}_M]^{-1}\nabla V(\mathbf{c}_M^{i-1})$, где $\delta > 0$ и выбрано так, чтобы весовая матрица была положительно определенной. В случае, если вычисление Гессиана представляет большие трудности, вместо этого используется аппроксимация $\mathbf{A}_M^i$, т. е. принимается $\mathbf{c}_M^i = \mathbf{c}_M^{i-1} - \mu_i[\mathbf{A}_M^i)]^{-1}\nabla V(\mathbf{c}_M^{i-1})$. Метод оптимизации в последних двух вариантах называется квази-Ньютоновским. В качестве матрицы $\mathbf{W}_M^i$ может быть взята и постоянная матрица[3].

Для всех детерминированных итеративных схем оптимизации, которые определены выше, существует несколько алгоритмов стохастической аппроксимации. Достаточно заменить слагаемое, связанное с ФС, подходящими приближенными величинами, которые вычисляются итеративно по мере получения нового набора отсчетов входа и выхода $(x(n), y(n))$. Таким образом, в схемах СА используются шаги итерации с обновлением времени.

При этом рекурсивный алгоритм СА:

$$\mathbf{c}_M(n) = \mathbf{c}_M(n-1) - \frac{1}{2}\mu(n)\mathbf{W}_M(n)\mathbf{g}_M(n), \qquad (19)$$

[1] Стохастическая оптимизация (в определенном смысле противоположная алгоритму детерминированной оптимизации) получается, если функция стоимости, градиент или Гессиан возможно заменить на несмещенные оценки, т. е. соответственно на $\hat{V}(\hat{\mathbf{c}}_M)$, $\nabla\hat{V}(\hat{\mathbf{c}}_M)$ и $\nabla^2\hat{V}(\hat{\mathbf{c}}_M)$.

[2] В некоторых случаях μ_i выбирается таким, что функция стоимости (ФС) на i-м шаге минимизируется на основе использования метода оптимизации с линейным поиском [2].

[3] Последние три метода представляются специальными случаями квази-Ньютоновского приближения и их часто называют методами предварительного создания условий (*preconditioning methods*).

где $\mathbf{c}_M(n)$ означает оценку вектора параметров неизвестной системы; $\mathbf{g}_M(n)$ — оценка градиента:

$$\mathbf{g}_M(n) \equiv \nabla \hat{V}(\mathbf{c}_M) = \nabla_{\mathbf{c}_M(n-1)} (\hat{E}_n[e^2(n)]), \qquad (20)$$

причем оператор ожидания $E[\cdot]$заменен на оценку среднего по времени $\hat{E}_n[\cdot]$; $\mathbf{W}_M(n)$ — (зависящая от времени) взвешивающая матрица, которую обычно полагают оценкой обратного Гессиана или его приближением. (Коэффициент 1/2 в (19) выбран просто для удобства.)

Алгоритмы с временной адаптацией, предназначенные для оценки параметров оптимальных КИХ-фильтров путем минимизации (6), можно интерпретировать как специальные случаи основной схемы (19) рекурсивной стохастической аппроксимации.

Алгоритмы с адаптацией градиента. Алгоритмы с адаптацией градиента получаются из основной схемы (19) на основе подстановки $\mathbf{W}_M(n) = \mathbf{I}_M$:

$$\mathbf{c}_M(n) = \mathbf{c}_M(n-1) - \frac{1}{2}\mu(n)\mathbf{g}_M(n). \qquad (21)$$

В зависимости от выбора схемы стохастической аппроксимации выводится несколько адаптивных алгоритмов.

Аппроксимация градиента без использования памяти. Аппроксимация усредняющего оператора это оценка, основанная только на точке текущих данных (без использования памяти) [3]. Так, что соотношение

$$\hat{V}(\mathbf{c}_M) = e^2(n) \qquad (22)$$

дает в результате оценку градиента в виде $\mathbf{g}_M(n) = -2\mathbf{x}_M(n)e(n)$,

$$e(n) = y(n) - \mathbf{x}_M^T(n)\mathbf{c}_M(n-1), \qquad (23)$$

и $e(n)$ известна как априорная ошибка фильтрации.

Так получается известный алгоритм НСК [3], [4] (рис. 1), который прост по структуре и хорошо обоснован теоретически. Его основные свойства (сходимость, возможность отслеживания и робастность) экспериментально верифицированы.

Инициализация : $\mathbf{c}_M(-1)=\mathbf{0}$
$e(n) = y(n) - \mathbf{x}_M^T(n)\mathbf{c}_M(n-1)$ (1) $\mathbf{c}_M(n) = \mathbf{c}_M(n-1) + \mu(n)\mathbf{x}_M(n)e(n)$ (2)
$\mu(n)$ – *константа,* МНК-алгоритм
$\mu(n)$ – *константа,* МНК-алгоритм
$\mu(n) = \dfrac{\alpha}{\beta + \mathbf{x}_M^T(n)\mathbf{x}(n)}$, $\alpha \in (0,2)$, $0 \le \beta$, Нормализованный МНК (Н_МНК) алгоритм
$\mu(n) = \alpha/\sigma_x^2(n)$, $\sigma_x^2(n) = \lambda\sigma_x^2(n-1) + e_M^2(n)$, $\lambda \in (0,1)$, $0 < \alpha < 2/M$, Нормализованный по мощности НСК (НМ_НСК) алгоритм

Рис. 1. Алгоритм НСК и основные его варианты

Алгоритм настраивается параметром μ на так называемый размер шага, и если размер шага ограничен условием $0 < \mu < 2/[\lambda_{\text{макс}}(\mathbf{R}_M)]$ (где $\lambda(\mathbf{R}_M)$ — собственное значение автокорреляционной матрицы), то алгоритм НСК сходится в среднем к оптимальному решению (10).

Если μ оптимизировано по соотношению $\mu_i = \arg\min_{\mu} V(\mathbf{c}_M^{i-1} + \mu\mathbf{v}_M^i)$, то получается нормализованный алгоритм НСК [5]. В этом случае μ изменяется со временем и определяется соотношением $\mu(n) = 1/[\mathbf{x}_M^T(n)\mathbf{x}_M(n)]$. Нормализованный алгоритм МНК допускает также детерминированную интерпретацию как фильтр, который минимизирует норму ошибки (20) и удовлетворяет ограничению, определяемому моделью (2), т. е. $y(n) = \mathbf{x}_M^T(n)\mathbf{c}_M$:

$$\mathbf{c}_M(n) = \min_{\mathbf{c}} \|\mathbf{c}_M - \mathbf{c}_M(n-1)\|^2. \qquad (24)$$

Более робастный (устойчивый к шумам) алгоритм получается заменой оптимального размера шага путем использования несколько другой формулы: $\mu(n) = \alpha/[\beta + \mathbf{x}_M^T(n)]$, $\alpha \in (0, 2)$, $0 \leqslant \beta$. (Присутствие β гарантирует, что знаменатель не обращается в нуль, а α — коэффициент, уменьшающий μ.) Другой вариант, известный как алгоритм НСК с нормализованной мощностью, получается при использовании μ в виде: $\mu(n) = \alpha/\sigma_x^2(n)$, где $\sigma_x^2(n)$ — мощность входного сигнала. Для стационарных сигналов σ_x^2, $\mu(n)$ могут быть постоянными значениями и в этом случае $0 < \alpha < 2/M$. Нормализованный алгоритм НСК (алгоритм Н_НСК) и алгоритм НСК с нормализованной мощностью (алгоритм НМ_НСК) показаны на рис. 1. Основное преимущество этих алгоритмов состоит в более быстрой сходимости (сравнительно с основным вариантом алгоритма НСК).

Скорость сходимости описанных выше алгоритмов существенно зависит от распределения собственных чисел автокорреляционной матрицы входного сигнала. Так, эти алгоритмы сходятся с неприемлемо низкой скоростью для входного сигнала в виде окрашенного шума или для речевого сигнала.

Аппроксимация со скользящим окном. Кроме оператора временного усреднения используется аппроксимация со скользящим окном [6]:

$$E_n[\cdot] = (1/L) \sum_{k=n-L+1}^{n} [\cdot]_k, \qquad (25)$$

где L определяет память, используемую аппроксимацией; $L \geqslant$ или $<$,

чем порядок фильтра (M). Использование $\hat{V}(\mathbf{c}_M) = (1/L) \sum_{k=n-L+1}^{n} [e^2(k)]$ порождает оценку градиента в следующем виде:

$$\mathbf{g}_M(n) = -\left(\frac{2}{L}\right) \sum_{k=n-L+1}^{n} \mathbf{x}_M(k)[y(k) - \mathbf{x}_M^T(k)\mathbf{c}_M(n-1)]. \quad (26)$$

Теперь, если взять матрицу входных данных размерности ($M \times L$)

$$\mathbf{X}_{M,L}(n) = [\mathbf{x}_M(n), \mathbf{x}_M(n-1), ..., \mathbf{x}_M(n-L+1)] \quad (27)$$

и вектор данных желаемого отклика размерности ($L \times 1$)

$$\mathbf{y}_L(n) = [y(n), y(n-1), ..., y(n-L+1)]^T, \quad (28)$$

то использование (27) и (28) позволяет привести вектор градиента (26) к виду

$$\mathbf{g}_M(n) = -\left(\frac{2}{L}\right) \mathbf{X}_{M,L}(n)\mathbf{e}_L(n). \quad (29)$$

Здесь $\mathbf{e}_L(n)$ определяется соотношением

$$\mathbf{e}_L(n) = \mathbf{y}_L(n) - \mathbf{X}_M^T(n)\mathbf{c}_M(n-1). \quad (30)$$

Подстановка приведенных выше соотношений в алгоритм СА с использованием метода крутого спуска[4] позволяет получить стохастический градиентный алгоритм, который называют алгоритмом НСК со скользящим окном (НСК_СО). Этот алгоритм показан на рис. 2, где для него установлено $\tilde{\mathbf{g}}_M(n) = (1/2)\mathbf{g}_M(n)$. Величина $\tilde{\mu}(n)$ на рис. 2 определяется с помощью процедуры линейного поиска. (Специальным случаем алгоритма НСК_СО является использование постоянного размера шага: $\mu(n) = \mu$).

Сравнительно с алгоритмом аппроксимации на основе временного усреднения без использования памяти алгоритм (НСК_СО) имеет более быструю сходимость. Однако вычислительная сложность его (выраженная в числе операций на одну итерацию) возрастает в L раз, составляя $4LM + 3L$. Это затрудняет использование алгоритма в приложениях РВ, если для вычисления (29) и (30) нет возможности применить эффективную схему, ориентированную на специальную структуру входящих в расчет матриц [7].

Аппроксимация с использованием экспоненциального забывания. Другой вариант алгоритма НСК получается из рассмотрения оператора временного усреднения, имеющего форму суммирования с экспоненциальным забыванием:

$$\hat{E}_n[\cdot] = \sum_{k=0}^{n} \lambda^{n-k}[\cdot] \; 0 < \lambda \leqslant 1. \quad (31)$$

Это алгоритм НСК_ЭЗО и для него $\hat{V}(\mathbf{c}_M) = \sum_{k=0}^{n} \lambda^{n-k}[e^2(n)]$. Это приближение приводит к оценке градиента: $\mathbf{g}_M(n) = -2(\mathbf{d}_M(n) - \mathbf{R}_M(n)\mathbf{c}_M(n-1)$, где $\mathbf{R}_M(n) = \sum_{k=0}^{n} \lambda^{n-k}\mathbf{x}_M(k)\mathbf{x}_M^T(k)$ и $\mathbf{d}_M(n) = \sum_{k=0}^{n} \lambda^{n-k}\mathbf{x}_M(k)y(k)$.

Способы ускорения адаптивно-градиентных методов. Увеличение скорости сходимости ("ускорение") адаптивно-градиентных методов в значительной степени зависит от относительного диапазона рассеяния собственных чисел ($\lambda_{max}/\lambda_{min}$) входной автокорреляционной матрицы и в то же время эти методы сохраняют низкую вычислительную сложность. Для увеличения скорости сходимости предложены методы предобработки входных данных (предварительное отбеливание и декорреляция входных сигналов [3, 8, 9]).

Декорреляция во временной области. По аналогии с алгоритмом градиентного ("крутого") спуска (16) стохастическая аппроксимация этой схемы может быть получена путем выбора

$$\mathbf{v}_M(n) = \mathbf{x}_M(n) - a(n)\mathbf{x}_M(n-1), \quad (32)$$

где $a(n) = \dfrac{\mathbf{x}_M^T(n)\mathbf{x}_M(n-1)}{\mathbf{x}_M^T(n-1)\mathbf{x}_M(n-1)}$ — коэффициент декорреляции $\mathbf{x}_M(n)$ и $\mathbf{x}_M(n-1)$. Рекурсия обновления фильтра принимает следующий вид:

$$\mathbf{c}_M(n) = \mathbf{c}_M(n-1) + \mu(n)\mathbf{v}_M(n). \quad (33)$$

Инициализация: $\mathbf{c}_M(-1)=\mathbf{0}$	
$\mathbf{X}_{M,L}(n) = [\mathbf{x}_M(n)\mathbf{x}_M(n-1)...\mathbf{x}_M(n-L+1)$	(1)
$\mathbf{y}_L(n) = [\mathbf{y}(n)\mathbf{y}(n-1)...\mathbf{y}(n-L+1)]^T$	(2)
$\mathbf{c}_L(n) = \mathbf{y}(n) - \mathbf{X}_{M,L}^T(n)\mathbf{c}_M(n-1)$	
$\tilde{\mathbf{g}}_M(n) = \mathbf{X}_{M,L}(n)\mathbf{e}_L(n)$	(3)
$\tilde{\mu}(n) = \dfrac{L\tilde{\mathbf{g}}_M^T(n)\tilde{\mathbf{g}}_M(n)}{2\tilde{\mathbf{g}}_M^T(n)\mathbf{X}_{M,L}(n)\mathbf{X}_{M,L}^T\tilde{\mathbf{g}}_M(n)}$	(4)
$\mathbf{c}_M(n) = \mathbf{c}_M(n-1) + \tilde{\mu}(n)\tilde{\mathbf{g}}_M(n)$	(5)

Рис. 2. Алгоритм НСК со скользящим окном (НСК_СО)

[4] Минимизировать $V(\mathbf{c}_M) = E[Q(\mathbf{c}_M, \eta)]$.

Инициализация: $\mathbf{c}_M(-1) = 0$
$e(n) = y(n) - \mathbf{x}_M^T(n)\mathbf{c}_M(n);\quad \mathbf{v}_M(n) = \mathbf{x}_M(n) - \alpha(n)\mathbf{x}_M(n-1)$
$\alpha(n) = \dfrac{\mathbf{x}_M^T(n)\mathbf{x}_M(n-1)}{\mathbf{x}_M^T(n-1)\mathbf{x}_M(n-1)};\quad \mu(n) = \dfrac{\rho e(n)}{\mathbf{x}_M^T(n)\mathbf{v}_M(n)}$ (ρ – выравнивающий коэффициент)
$\mathbf{c}_M(n) = \mathbf{c}_M(n=1) + \mu(n)\mathbf{v}_M(n)$

Рис. 3. Алгоритм НСК с предварительной декорреляцией

Инициализация: $\mathbf{c}_M(-1) = 0$	
$e(n) = y(n) - \mathbf{x}_M^T(n)\mathbf{c}_M(n-1);$	$\tilde{e}(n) = e(n) + \mathbf{a}_M^T(n)\mathbf{e}_M(n)$
$\mathbf{e}_M = [e(n)e(n-1)...e(n-M+1)]^T;$	$\mathbf{c}_M(n) = \mathbf{c}_M(n-1) + \mu\mathbf{e}_M^f(n)\tilde{e}_M(n)$
$e^f(n) = x(n) + \mathbf{a}_M^T(n)\mathbf{x}_M(n-1);$	$\mathbf{e}_M^f(n) = [e^f(n)e^f(n-1)...e^f(n-M+1)]^T$

Рис. 4. Алгоритм НСК с фильтрацией

Оптимальный размер шага μ получается путем минимизации $\mu(n) = e(n)/(\mathbf{x}_M^T\mathbf{v}_M)$. Метод можно рассматривать как метод с адаптивной "приборной переменной" [10, 11]. Алгоритм представлен на рис. 3. Подробное рассмотрение свойства повышенной скорости сходимости этого алгоритма содержится в работе [12].

Мбоупом, Бонетом и Бершадом (Mboup, Bonnet, Bershad [8]) построен улучшенный вариант алгоритма. Ошибка предсказания (вперед) порядка M применена для создания приборной переменной в виде:

$$\mathbf{v}_N(n) \equiv \mathbf{e}_M^T(n) = \\ = [e^f(n)e^f(n-1)\ ...\ e^f(n-M+1)]^T; \qquad (34)$$

$$e^f(n) = x(n) + \mathbf{a}_M^T(n)\mathbf{x}_M(n-1). \qquad (35)$$

Соотношение (35) выражает ошибку предсказания (вперед) данных входного сигнала, а $\mathbf{a}_M(n)$ — вектор, определяющий ошибку предсказания на момент времени n (так называемый "предсказатель"). Кроме того, мгновенная ошибка $e(n)$ фильтруется предсказателем (вперед) в момент n:

$$\tilde{e}(n) = e(n) + \mathbf{a}_M^T(n)\mathbf{e}_M(n),$$

где $\mathbf{e}_M(n) = [e(n)e(n-1)\ ...\ e(n-M+1)]^T$. (36)

Предсказатель вперед $\mathbf{a}_M(n)$ адаптивно оценивается посредством алгоритма НСК. Обсуждаемый алгоритм, условно называемый алгоритмом "НМК с фильтрованным x", показан на рис. 4. Вычислительная сложность алгоритма типа НСК с предварительной декорреляцией (см. рис. 3.) пропорциональна M и проанализирована в работах [13, 14].

Декорреляция в области преобразования. Первоначально для улучшения качества алгоритмов НСК применялось унитарное преобразование входного вектора сигналов $\mathbf{x}_M(n)$. Полученный таким образом алгоритм имел большую скорость сходимости только для отдельных классов входных сигналов, но вычислительная сложность его меньше, чем у основного варианта алгоритма НСК. Семейство этих алгоритмов известно как "алгоритмы адаптивной фильтрации в области преобразования" [13, 14]. Эти алгоритмы интерпретируются как специальный случай соотношения (19). Выбранная подходящим образом фиксированная взвешивающая матрица используется для декорреляции. Для улучшения сходимости алгоритма НСК используется дискретное преобразование Фурье (ДПФ) или дискретное косинус-преобразование с последующей нормализацией. Такая предобработка, за которой следует нормализация, понижает (относительный) диапазон собственных чисел у матрицы автокорреляции вектора входных сигналов, что и способствует увеличению скорости сходимости модифицированного алгоритма НСК. Выполнение этих методов зависит от степени ортогонализации, достигаемой после применения преобразования.

Другой подход к ускорению алгоритмов НСК основан на субполосном разложении сигналов и адаптивной фильтрации в каждой из полос [15—18]. Это обобщение адаптивной фильтрации в области преобразования, где применение ДПФ (в качестве предобработки) можно интерпретировать как разложение на субполосы.

Адаптивные алгоритмы Гаусса—Ньютона. Поиск оптимального фильтра может быть ускорен, если градиент достаточно быстро изменяется в окрестности минимума функции стоимости. Адаптивный алгоритм Гаусса — Ньютона получается, когда весовая матрица $\mathbf{W}_M(n)$ выбирается в виде обратной матрицы относительно оценки Гессиана функции стоимости. При этом весовая матрица определяется выражением: $\mathbf{W}_M(n) \equiv [\nabla^2 \hat{V}(\hat{\mathbf{c}}_M)]^{-1} = [(\nabla_{\mathbf{c}_M(n-1)})^2 \times$ $\times\ (\hat{E}_n[e^2(n)])]^{-1}$, где оператор ожидания $E[\cdot]$ заменен на оценку среднего по времени $\hat{E}[\cdot]$. Для $\hat{E}[\cdot]$ используется суммирование в окне с экспоненциальным забыванием "старых" отсчетов, поэтому оценка градиента определяется соотношением (32), а оценка Гессиана — соотношением $\nabla^2 \hat{V}(\mathbf{c}_M) = \mathbf{R}_M(n)$. Оптимальный размер шага оценивается при $\mu(n) = 1$ и получающееся соотношение $\mathbf{c}_M(n) = \mathbf{c}_M(n-1) + \mathbf{R}_M^{-1}(n)(\mathbf{d}_M(n) - \mathbf{R}_M(n)\mathbf{c}_M(n-1))$

приводит к известному алгоритму НСК (36). Решение его с помощью временной рекурсии отражает схему P_НСК [4] и основано на свойстве обновления, которым обладают автокорреляционная матрица и кросскорреляционный вектор отсчетов сигнала: $\mathbf{R}_M(n) = \lambda\mathbf{R}_M(n-1) + \mathbf{x}_M(n)\mathbf{x}_M^T(n)$ и $\mathbf{d}_M(n) = \lambda\mathbf{d}_M(n-1) + \mathbf{x}_M(n)y(n)$.

Заключение. Адаптивная идентификация реализует процедуру, в которой при поступлении новой пары измерений сигналов вход—выход получается дополнительная информация относительно модели системы. Это позволяет обновлять в реальном времени работу адаптивного алгоритма идентификации.

В качестве класса моделей, для которых рассматривается оптимальная адаптивная фильтрация и идентификация, использованы линейные системы с КИФ. Каждая выходная величина такой системы определяется взвешенной комбинацией фиксированного конечного числа (M) предшествующих значений входного сигнала системы.

Качество функционирования адаптивного алгоритма определяется рядом показателей: точностью получаемого решения относительно теоретически ожидаемой; скоростью сходимости; способностью отслеживать изменяющуюся статистику сигналов системы; вычислительной сложностью алгоритма; робастностью к накоплению ошибок округления. В том случае, когда используются многопроцессорные вычислительные системы, вопросы параллельного или конвейерного потока данных также относятся к показателям функционирования АА фильтрации.

Рассмотрена система алгоритмов, которые служат для создания адаптивных фильтров и идентификации систем. Основой фильтров служат принципы Винеровской фильтрации и стохастической аппроксимации. Проанализирован алгоритм наименьших средних квадратов и его варианты (алгоритм с адаптацией градиента, алгоритм с аппроксимацией оператора ожидания скользящим окном и окном с экспоненциальным забыванием). Показана интерпретация этих алгоритмов как СА и оптимизация методом градиентного спуска. Рекурсивный и квазирекурсивный алгоритмы НСК выведены как стохастические аппроксимации итеративного детерминированного ньютоновского метода. Показаны способы ускорения адаптивных градиентных методов.

Концепции, методы и формы тестирования алгоритмов, а также отдельные приложения разработаны Куснером и Юном, Хайкиным, Вангом, Шанчезом и Гарсиа ([1], [4]—[6], [13]).

Список литературы

1. **Kusner H., Yiin G.** Stochastic approximation algorithms and applications. Berlin: Springer, 1997. 174 p.
2. **Pflug G.** Optimization of stochastic models. Kluwer Academic, 1996. 166 p.
3. **Уидроу Б., Стирнз С.** Адаптивная обработка сигналов: пер. с англ. М.: Радио и связь, 1989. 439 с.
4. **Haykin S.** Adaptive Filter Theory. Prentice Hall Publishing, 1996. 458 p.
5. **Haykin S., Widrow B.** (eds.) Least Mean-Square Filters: New Insight and Development Insight. Wiley—Interscience, 2002. 310 p.
6. **Wang T., Wang S.** On the optimum design of the block adaptive FIR filter // IEEE Transactions on Signal Processing. 1994. Vol. 41. N 6. P. 2131—2140.
7. **Mikhael W., Wu F.** Fast algorithms for block FIR adaptive digital filtering // IEEE Transactions on Circuits and Systems. 1987. Vol. CAS-34. N 10. P. 1152—1160.
8. **Mboup M., Bonnet M., Bershad N.** LMS coupled adaptive prediction and system identification: statistical model and transient mean analysis // IEEE Transactions on Acoustic, Speech and Signal Processing. 1994. ASSR-42. N 10. P. 2607—2615.
9. **Меркушева А. В., Малыхина Г. Ф.** Методы и алгоритмы разделения смеси сигналов: Применение декорреляции и статистик второго порядка // Научное приборостроение. 2009. Т. 19. № 2. Р. 90—103.
10. **Гроп Д.** Методы идентификации систем. М.: Мир, 1979. 302 с.
11. **Льюнг Л.** Идентификация систем. Теория для пользователя. М.: Наука, 1991. 427 с.
12. **Doherry J., Porayath R.** Robust echo canceler for acoustic environments // IEEE Transactions on Circuits and Systems II. 1997. Vol. 44. N 5. P. 389—398.
13. **Sanchez V., Garsia P.** Diagonalizing properties of the DCT // IEEE Transactions on Acoustic, Speech and Signal Processing. 1995. Vol. ASSR-432. N 11. P. 2680—2687.
14. **Beufays F.** Transform-domain adaptive filters: An analysis approach // IEEE Transactions on Acoustic, Speech and Signal Processing. 1995. Vol. ASSR-42. N 2. P. 422—431.
15. **Shink J.** Frequency-domain and multirate adaptive filtering // IEEE Signal Processing Magazine. 1992. Vol. 9. N 1. P. 14—39.
16. **Resende F., Diniz P. S. R., Tokuda K., Raneko M., Nishihara A.** New adaptive algorithm based on multi-band decomposition of the error signal // IEEE Transactions on Circuits and Systems II. 1998. V. 45, N. 5. P. 392—599.
17. **Clark G., Mitra S., Parker S.** Block implementation of adaptive digital filters // IEEE Transactions on Circuits and Systems. 1981. Vol. CAS-26. N 6. P. 584—592.
18. **Меркушева А. В., Малыхина Г. Ф.** Согласованное многоканальное разделение сигнала: фильтрация и мультиплексирование // Научное приборостроение. 2008. Т. 18. № 1. С. 98—109.

УДК 621.396.6.049.75 (075)

В. В. Воловиков[1], канд. техн. наук, доц.,
А. В. Сарафанов[2, 3], д-р техн. наук, проф.,
директор по развитию бизнеса
[1]Московский государственный институт
электроники и математики,
e-mail: valbox@rambler.ru,
[2] Сибирский федеральный университет,
[3] ЗАО "Ай-Теко",
e-mail: sav@lab-init.ru

Моделирование теплового сопротивления при вынужденном конвективном теплообмене в каналах конструкций радиоэлектронной аппаратуры

Рассматриваются вопросы снижения погрешностей вычисления сопротивления конвективного теплообмена в каналах конструкций радиоэлектронной аппаратуры для математических моделей, формируемых на основе эквивалентных тепловых схем. Приведены аналитические зависимости для теплообмена при ламинарном, переходном и турбулентном течениях теплоносителя в каналах круглого и прямоугольного сечениях. Предложены способ вычисления теплового сопротивления при разбиении канала на изотермические участки и поправка на соотношение сторон при турбулентном течении в прямоугольном канале, обеспечивающие снижение погрешностей вычислений.

Ключевые слова: *моделирование тепловых процессов, радиоэлектронные устройства, электротепловая аналогия, вынужденная конвекция в каналах, снижение погрешности, тепловое сопротивление*

Для моделирования тепловых процессов в конструкциях радиоэлектронной аппаратуры (РЭА), оборудованных системами принудительного охлаждения (воздушного и жидкостного), используют формулы теории теплопередачи [4, 8–12], позволяющие определять средний коэффициент теплоотдачи от поверхностей электрорадиоизделий (ЭРИ) и конструктивных элементов (КЭ). Наиболее часто в РЭА потоки теплоносителя движутся по каналам прямоугольного (образуются между печатными узлами и стенками корпусов изделий) и круглого сечений (трубки в системах принудительного жидкостного охлаждения). При расчетах принимают допущение, что в моделируемой области пространства температуры теплоносителя и омываемого теплоносителем КЭ постоянны. На практике же в РЭА имеют место перепад температуры теплоносителя по мере его прохождения внутри конструкции; высокий градиент температуры по поверхности КЭ; неравномерность плотности теплового потока на поверхности КЭ. Для учета указанных эффектов в расчетных моделях поверхность конструктивного элемента (стенка корпуса, печатная плата) и поток теплоносителя принято разбивать на фрагменты, в пределах которых указанные величины меняются незначительно.

Средний коэффициент теплоотдачи в этом случае вычисляют по формуле [1, 3]

$$\bar{\alpha} = \frac{\lambda_f \overline{\mathrm{Nu}}}{D_\Gamma}, \qquad (1)$$

где λ_f — коэффициент теплопроводности теплоносителя; $\overline{\mathrm{Nu}}$ — среднее значение числа Нуссельта; D_Γ — гидравлический диаметр канала.

При ламинарном течении теплоносителя и стабилизированной теплоотдаче в гладком канале бесконечной длины круглого сечения и q_x = const (постоянной плотности теплового потока на поверхности стенки) значение критерия $\mathrm{Nu}_\infty = 4{,}364$, а при T_x = const (постоянной температуре стенки канала) $\mathrm{Nu}_\infty = 3{,}658$ [2, 3, 7]. При стабилизированной теплоотдаче в канале, образованном двумя бесконечными пластинами, критерий Нуссельта принимает значение $\mathrm{Nu}_\infty = 8{,}235$ при q_x = const и $\mathrm{Nu}_\infty = 7{,}54$ при T_x = const. Для каналов прямоугольного сечения с соотношением сторон a/b (a — большая сторона, b — меньшая сторона) значения критерия Нуссельта при стабилизированном течении приведены в табл. 1 [2].

Значения Nu_∞ получены для параболического распределения скоростей потока теплоносителя, которое имеет место при исчезающих малых температурных напорах и неизменных физических параметрах теплоносителя [7]. В этом случае не учитываются особенности теплообмена на начальном участке каналов, когда толщина пограничных слоев по сравнению с поперечными размерами канала мала, что способствует значительно лучшей теплоотдаче.

Таблица 1

Значения критерия Нуссельта при стабилизированном ламинарном течении теплоносителя в каналах прямоугольного сечения

b/a	Nu_∞					
	1,0	0,7	0,5	0,25	0,125	0
T_x = const	2,98	3,08	3,39	4,44	5,6	7,54
q_x = const	3,61	3,73	4,12	5,33	6,49	8,235

Теоретические и экспериментальные данные, приведенные в работах [1—3, 7, 9], показывают, что теплоотдача при постоянной температуре стенки ниже, чем при постоянной плотности теплового потока на ее поверхности. Так как в РЭА температура стенок канала и плотность теплового потока на них не постоянны по длине каналов, то для вычисления критерия Нуссельта обычно используются приводимые ниже формулы (2), (4), (5), соответствующие наихудшему из этих двух случаев — постоянной температуре стенки.

Среднее значение критерия Нуссельта определяется по формулам, приведенным в работе [3]:

а) для ламинарного режима в трубах круглого сечения при $\mathrm{Re}_f \leqslant 2200$

$$\overline{\mathrm{Nu}} = 3{,}657 + \frac{0{,}0668\mathrm{Gz}_f^{\frac{1}{3}}}{0{,}04\mathrm{Gz}_f^{-\frac{2}{3}}}; \qquad (2)$$

$$\mathrm{Gz} = \frac{\mathrm{Re}_f \mathrm{Pr}_f D_\Gamma}{l}; \qquad (3)$$

б) для каналов, образованных бесконечными параллельными пластинами, при двухстороннем нагреве $\mathrm{Re}_f \leqslant 2200$ и $0{,}1 \leqslant \mathrm{Pr}_f \leqslant 1000$ [1]

$$\overline{\mathrm{Nu}} = 7{,}55 + \frac{0{,}024\mathrm{Gz}_f^{1{,}14}}{1 + 0{,}0358\mathrm{Pr}_f^{0{,}17}\mathrm{Gz}_f^{0{,}64}}; \qquad (4)$$

в) для турбулентного режима при $2300 \leqslant \mathrm{Re}_f \leqslant 5 \cdot 10^6$; $0{,}6 < \mathrm{Pr}_f < 2000$; $0 < d/l < 1$ [1—3]:

$$\overline{\mathrm{Nu}} = \frac{(\xi/8)(\mathrm{Re}_f - 1000)\mathrm{Pr}_f}{1 + 12{,}7\sqrt{\xi/8}\left(\mathrm{Pr}_f^{\frac{2}{3}} - 1\right)} \varepsilon'_L \varepsilon'_\mu; \qquad (5)$$

$$\xi = \frac{A}{(1{,}82\log_{10}\mathrm{Re}_f - 1{,}64)^2} \varepsilon''_\mu. \qquad (6)$$

В приведенных выражениях Gz_f — число Гретца, d — диаметр трубы; l — длина трубы; Re_f, Pr_f, Gr_f — числа Рейнольдса, Прандтля и Грасгофа, отнесенные к среднеобъемной температуре теплоносителя; A — поправочный коэффициент на форму сечения канала; ε'_L — поправочный коэффициент на конечность трубы, ε'_μ, ε''_μ — поправочные коэффициенты на переменность физических свойств жидкости и газа; ξ — коэффициент сопротивления трения.

Соотношение (2) справедливо для каналов круглого сечения и дает ошибку вычисления $\overline{\mathrm{Nu}}$ менее 14 % при $\mathrm{Gz} > 1000$ и менее чем 7 % при $\mathrm{Gz} < 1000$ [3]. В то же время соотношение (2) непригодно для расчета теплоотдачи в каналах прямоугольного сечения с использованием гидравлического диаметра [7]. При равномерном распределении скоростей течения теплоносителя на входе течение и теплообмен на начальном участке подобны течению и теплообмену теплоносителя при продольном обмывании пластины в силу малой толщины пограничных слоев по сравнению с размерами сечения канала. Однако по мере удаления от входа из-за большого влияния стеснения потока закономерности процесса меняются [7]. В связи с этим теплообмен при ламинарном течении на входе в канал прямоугольного сечения предлагается описывать с помощью уравнений для плоской пластины, а при стабилизированном течении руководствоваться данными, приведенными в табл. 1.

Выражение (5) для труб круглого сечения дает ошибку вычисления среднего значения числа Нуссельта ±10 % и может быть использовано как при постоянной температуре стенки, так и при постоянной плотности теплового потока на ее поверхности [1]. Данное соотношение может быть также применено для каналов с другими формами сечений. В этом случае в расчетах вместо диаметра канала d используют гидравлический диаметр D_Γ. Ошибка вычислений при этом составляет ±20 % [3]. Поправочный коэффициент на форму сечения канала $A = 1$ для труб круглого сечения, для каналов прямоугольного сечения A определяется по табл. 2 [6].

В соотношении (5) множитель ε'_L является поправкой на конечность трубы и вычисляется по формуле

$$\varepsilon'_L = 1 + \frac{C}{(l/D_\Gamma)^n}. \qquad (7)$$

Значения коэффициентов C и n для воздуха и газов с $\mathrm{Pr} = 0{,}7$ при различных условиях распределения скоростей течения на входе в канал и $l/D > 3$ приведены в табл. 3 [3]. Для других жидкостей и газов

Таблица 2

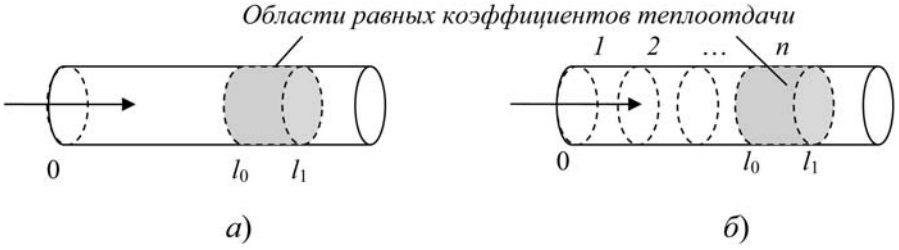
Значения коэффициента *A* для каналов прямоугольного сечения

b/a	1	0,8	0,6	0,4	0,2	0,1	0
A	1	1,01	1,02	1,04	1,06	1,08	1,1

Таблица 3

Значения коэффициентов *C* и *n* при различных условиях на входе в канал

Условие на входе в канал	*C*	*n*
Равномерное распределение скоростей	2,4254	0,676
Радиусный изгиб канала на 180°	0,9759	0,700
Радиусный изгиб канала на 90°	1,0517	0,629
Прямоугольный поворот канала	2,0152	0,614



Формализованное представление канала:
а — канал с изотермическими стенками; *б* — канал с неизотермическими стенками, разделенный на условно изотермические области

при $l/D_г > 1$, а также при неизвестных условиях на входе рекомендуется использовать значения $C = 1$, $n = 2/3$ [2].

Множитель ε'_μ представляет собой поправку на переменность физических свойств теплоносителя от температуры и направления теплового потока. Для жидкостей при условии, что $0{,}025 \leqslant \mu_f/\mu_w \leqslant 12{,}5$, справедливо выражение

$$\varepsilon'_\mu = \left(\frac{\mu_f}{\mu_w}\right)^m, \qquad (8)$$

где μ_f, μ_w — динамические коэффициенты вязкости при температуре потока и стенки канала соответственно; $m = 0{,}11$ при нагревании и $m = 0{,}25$ при охлаждении жидкости [3, 7].

Вычисление поправки ε'_μ для газов проводят по формуле (9), использование которой допустимо при $0{,}27 \leqslant (T_f/T_w) \leqslant 2{,}7$:

$$\varepsilon'_\mu = \left(\frac{T_f}{T_w}\right)^m, \qquad (9)$$

где $m = 0{,}47$ при нагревании и $m = 0{,}36$ при охлаждении газа [3].

Множитель ε''_μ является поправкой коэффициента сопротивления трения на переменность физических свойств теплоносителя от температуры и направления теплового потока. Для жидкостей при $0{,}5 \leqslant (\mu_f/\mu_w) \leqslant 3$ используют соотношение [3]

$$\xi''_\mu = \begin{cases} (7 - \mu_f/\mu_w)/6 \text{ при } T_w > T_f; \\ (\mu_f/\mu_w)^{-0{,}24} \text{ при } T_w < T_f. \end{cases} \qquad (10)$$

Для газов при $0{,}27 \leqslant (T_f/T_w) \leqslant 2{,}7$ используют соотношение [3]

$$\xi''_\mu = \left(\frac{T_f}{T_w}\right)^m; \; m = \begin{cases} 0{,}52 \text{ при } T_w > T_f; \\ 0{,}38 \text{ при } T_w < T_f. \end{cases} \qquad (11)$$

Так как в РЭА стенки труб и каналов нагреты неравномерно, при моделировании их разбивают на условно изотермические области [4, 5]. Для таких областей средний коэффициент теплоотдачи рассчитывается индивидуально. В целях повышения точности моделирования при расчете среднего коэффициента теплоотдачи условно изотермической области стенки канала предлагается учитывать изменение среднего коэффициента теплоотдачи по длине канала с учетом расстояния изотермической области от начала канала или трубы.

Рассмотрим два канала с равными геометрическими размерами, первый из которых имеет изотермические стенки, а второй — неизотермические, разбитые на условно изотермические области (см. рисунок).

Средние коэффициенты конвективной теплоотдачи от областей изотермического и неизотермического каналов (выделены серым цветом на рисунке) будут равными, если выполняются следующие условия:

- размеры рассматриваемых областей одинаковы;
- температуры теплоносителя на левых границах рассматриваемых областей равны;
- температуры стенок канала в рассматриваемых областях равны;
- тепловые потоки между теплоносителем и рассматриваемой областью стенки канала равны;
- распределение скоростей течения теплоносителя на левых границах областей одинаковы.

Определим коэффициент теплоотдачи с учетом конечности трубы для условно изотермической области *n* (см. рисунок *б*), расположенной на расстоянии l_0 от начала канала, имеющей длину Δl.

Для изотермического канала, разделенного на фрагменты, выполняется следующее равенство:

$$\overline{\alpha} = \frac{\overline{\alpha}_1 S_1 + \overline{\alpha}_2 S_2 + \ldots + \overline{\alpha}_n S_n}{S_1 + S_2 + \ldots + S_n} = \frac{\sum\limits_{i=1}^{n} (\overline{\alpha}_i S_i)}{\sum\limits_{i=1}^{n} S_i}, \qquad (12)$$

где $\overline{\alpha}_i$ — средний коэффициент теплоотдачи *i*-го фрагмента канала; S_i — площадь *i*-го фрагмента канала.

Отсюда

$$\overline{\alpha}_n = \frac{\sum\limits_{i=1}^{n} (\overline{\alpha}_i S_i) - \sum\limits_{j=1}^{n-1} (\overline{\alpha}_j S_j)}{S_n}. \qquad (13)$$

Принимая во внимание выражение (1), можно также записать:

$$\overline{\mathrm{Nu}}_{l_0-l_1} = \frac{\overline{\mathrm{Nu}}_{0-l_1} S_{0-l_1} - \overline{\mathrm{Nu}}_{0-l_0} S_{0-l_0}}{S_{l_0-l_1}} = \frac{\overline{\mathrm{Nu}}_{0-l_1} l_1 - \overline{\mathrm{Nu}}_{0-l_0} l_0}{l_1 - l_0}. \quad (14)$$

Соотношение (14) может быть использовано при вычислении критерия Нуссельта на отрезке l_0—l_1 по формулам (2) и (4), а также при моделировании ламинарного течения в каналах прямоугольного сечения с помощью уравнений для плоской пластины.

Для соотношения (5) критерий Нуссельта на участке l_0—l_1 может быть получен не по формуле (14), а путем замены поправочного коэффициента ε'_L на $\varepsilon'_{l_0-l_1}$, рассчитываемый по формуле

$$\varepsilon'_{l_0-l_1} = \frac{\varepsilon'_{0-l_1} l_1 - \varepsilon'_{0-l_0} l_0}{l_1 - l_0}. \quad (15)$$

Приведенные выше соотношения дают наибольшую точность вычисления коэффициентов теплоотдачи для гладких труб. В конструкциях РЭА гладкие трубы наиболее часто встречаются в системах принудительного жидкостного охлаждения, когда теплоноситель циркулирует по трубкам радиаторов. В системах с принудительной вентиляцией РЭА теплоноситель, как правило, проходит через корпус устройства, двигаясь по каналам, образованным печатными узлами. Такие каналы не являются гладкими, так как на них расположены ЭРИ.

При ламинарном режиме течения обтекание выступов происходит плавно [6, 7], а теплоотдача может увеличиваться за счет того, что стенка с ЭРИ имеет большую поверхность теплообмена, чем гладкая.

При турбулентном режиме течения возможны два случая. В первом случае, если высота ЭРИ меньше толщины вязкого подслоя, то такие ЭРИ, как и при ламинарном течении, обтекаются безотрывно. Во втором случае, когда высота ЭРИ больше толщины вязкого подслоя, происходит отрывное вихревое обтекание, что приводит к увеличению теплоотдачи. Тем не менее, средняя теплоотдача в некоторых случаях может быть меньше, чем в гладком канале вследствие того, что за высокими ЭРИ у поверхности платы (стенки) могут образовываться застойные зоны.

На теплоотдачу влияет также неравномерность теплового поля печатных узлов (ПУ) [4, 5], причиной которой является низкий коэффициент теплопроводности материалов печатных плат. На ранних этапах проектирования (до определения параметров конструкции, ЭРИ и ПУ) точная оценка влияния высоты и размещения ЭРИ на поверхности платы на теплоотдачу затруднительна, так как существует огромное разнообразие форм и размеров ЭРИ, вариантов компоновки ПУ и т. д.

Приближенную оценку влияния высоты и размещения ЭРИ на теплоотдачу можно проводить на основе аналогов. При этом основными параметрами для сопоставления являются: гидравлический диаметр и линейные размеры канала, высота ЭРИ и расстояние между ними в направлении потока теплоносителя. Следует также учитывать формы ЭРИ, температуру теплоносителя, теплопроводность и толщину материала печатной платы, плотность ее заполнения ЭРИ и мощность тепловыделения ЭРИ.

Более точная оценка влияния высоты и размещения ЭРИ на теплоотдачу может быть получена из сопоставления результатов расчета топологических макромоделей с результатами расчета интегральных характеристик полных моделей ПУ. Данная операция по уточнению параметров макромоделей необходима с точки зрения повышения точности расчетов, так как в настоящее время при моделировании РЭА применяют иерархический подход. При этом макромодели используются при анализе конструкций верхних уровней иерархии (шкафы, крейты), а полные модели — для анализа отдельных ПУ. При этом граничные условия детальных моделей получают из результатов расчета макромоделей. Следовательно, точность расчета тепловых характеристик РЭА с применением макромоделей через граничные условия влияет на точность результатов расчета тепловых характеристик РЭА с применением детальных моделей.

В применяемых в практике моделирования топологических моделях тепловых процессов РЭА, изображаемых в виде тепловых схем [4, 5], теплообмен вынужденной конвекцией описывается тепловыми сопротивлениями, рассчитываемыми по формуле

$$R_{\mathrm{т}} = \frac{1}{\overline{\alpha} S}, \quad (16)$$

где $\overline{\alpha}$ — средний коэффициент теплоотдачи между поверхностью стенки и теплоносителем; S — площадь поверхности стенки канала.

В моделях конструкций РЭА верхних уровней иерархии, представляемых макромоделями [13], имеет место усреднение тепловых характеристик и коэффициентов теплоотдачи по крупным фрагментам внутреннего пространства. При этом учет влияния на тепловое сопротивление таких параметров, как размеры и размещение ЭРИ, неравномерность теплового поля ПУ (стенок каналов) и других, осуществляется за счет введения в формулу следующего параметра — дополнительной эффективной

площади теплообмена $S_{эф}$. В этом случае параметр S определяется соотношением

$$S = S_к + S_{эф}, \quad (17)$$

где $S_к$ — площадь стенок гладкого канала.

В общем случае параметр $S_{эф}$ является величиной, зависимой не только от геометрических размеров ЭРИ и их размещения в канале, но и от числа Re и теплового поля рассматриваемой стенки канала, т. е.

$$S_{эф} = S_{эф}(Q_{ЭРИ}, \mathrm{Re}, T), \quad (18)$$

где $Q_{ЭРИ}$ — множество параметров, определяющих размеры и размещение ЭРИ; Re — число Рейнольдса; T — температурное поле рассматриваемой стенки канала.

В эквивалентных тепловых схемах для моделирования теплоотдачи вынужденной конвекцией в канале (трубе) используется двухполюсный диссипативный элемент (сопротивление). Этот элемент следует реализовывать как направленный, т. е. один вывод должен соответствовать температуре стенки канала, а второй — температуре теплоносителя. При вычислении сопротивления это позволяет учесть эффект изменения теплоотдачи в зависимости от направления теплового потока, т. е. охлаждения или нагревания стенки канала теплоносителем.

Диссипативная ветвь теплообмена вынужденной конвекцией в канале (трубе), реализованная приведенным выше способом, обладает следующими достоинствами по сравнению с известными моделями:

- учитывается изменение теплового сопротивления в зависимости от охлаждения или нагревания стенки канала теплоносителем, условий течения теплоносителя на входе в канал, расстояния от входа до моделируемого фрагмента канала при разбиении канала на условно изотермические фрагменты;
- учитывается изменение теплового сопротивления в зависимости от формы и размеров канала при турбулентном течении теплоносителя, а также соотношения сторон канала прямоугольного сечения при ламинарном режиме течения;
- позволяет повышать точность моделирования, управляя тепловым сопротивлением с помощью параметра дополнительной эффективной площади теплообмена. Управление должно осуществляться на основе метода совместного анализа детальных моделей фрагментов и макромодели конструкции РЭА в целом.

Также следует отметить, что анализ работ по смежным областям [6] позволил внести в выражение (6) поправку на соотношение сторон прямоугольного канала. Способ вычисления теплового сопротивления с учетом указанной поправки в литературе не встречается, а ее введение в формулу расчета критерия Нуссельта не противоречит экспериментальным данным, приведенным в работах [1, 3].

Список литературы

1. **Adrian B., Kraus A. D.** Heat transfer handbook. New Jersey: John Wiley & Sons Inc. 2003. 1480 p.
2. **Frank K.** The CRC Handbook of Thermal Engineering. CRC Press, 2000.
3. **Lienhard J. H. IV., Lienhard J. H. V.** A Heat Transfer Textbook. Cambridge, Massachchusetts, USA: Phlogiston Press, 2003. 749 p.
4. **Гольдин В. В., Журавский В. В., Сарафанов А. В.** и др. Исследование тепловых характеристик РЭС методами математического моделирования / Под ред. А. В. Сарафанова. М.: Радио и связь, 2003. 456 с.
5. **Дульнев Г. Н., Тарновский Н. Н.** Тепловые режимы электронной аппаратуры: учеб. пособие. Л.: Энергия, 1971. 248 с.
6. **Идельчик Е. И.** Справочник по гидравлическим сопротивлениям / Под ред. М. О. Штейнберг. М.: Машиностроение, 1992. 672 с.
7. **Исаченко В. П., Осипова В. А., Сукомел А. С.** Теплопередача: учебник для вузов. М.: Энергия, 1975. 488 с.
8. **Кутателадзе О. С.** Теплопередача и гидравлическое сопротивление. М.: Энергоатомиздат, 1990. 367 с.
9. **Михеев М. А., Михеева И. М.** Основы теплопередачи. М.: Энергия, 1977. 343 с.
10. **Петухов Б. С.** Теплообмен и сопротивление при ламинарном течении жидкости в трубах. М.: Энергия, 1967. 184 с.
11. **Резников Г. В.** Расчет и конструирование систем охлаждения ЭВМ. М.: Радио и связь, 1988. 224 с.
12. **Харри У.** Основные формулы по теплообмену для инженеров. М.: Наука, 1997. 327 с.
13. **Воловиков В. В., Увайсов С. У.** Итерационное иерархическое моделирование физических полей радиоэлектронных устройств. Международный форум "Новые информационные технологии и менеджмент качества" (NIT & QM). М.: Фонд "Качество", 2009. С. 190—195.

УДК 004.928, 004.942

В. А. Бобков, д-р техн. наук, зав. лаб.,
e-mail: bobkov@iacp.dvo.ru,
С. В. Мельман, ст. инж.-программист,
e-mail: gruzd@dvo.ru,
В. П. Май, канд. техн. наук, вед. науч. сотр.,
e-mail: may@iacp.dvo.ru,
Институт автоматики и процессов управления ДВО РАН

Визуализация динамики циклонов[1]

Работа направлена на разработку программных графических средств визуального анализа синоптических объектов и, прежде всего, тропических циклонов. В представленной системе реализован метод пространственно-темпоральной визуализации динамики циклона с помощью вычисляемых по спутниковым данным температурных аномалий. Для увеличения вычислительной производительности обрабатываемых и визуализируемых данных применены шейдер и CUDA-технологии многопроцессорной обработки на базе графических процессоров.

Ключевые слова: *тропический циклон, визуализация скалярных полей, анимация, аномалия, шейдеры, CUDA*

Введение

Рассматриваемая в статье проблема визуализации синоптических объектов актуальна, так как средства визуального анализа необходимы для исследования, моделирования и прогноза течений, циклонов и других синоптических объектов в атмосфере и океане. Разработка таких средств ведется в мире давно. Вначале это были программы с преимущественной 2D-визуализацией, например, системы визуализации спецэффектов для нужд телевидения — система WeatherEye от европейского центра изучения и прогнозов погоды METOFFICE или Terra3D [1]. Недостаток этих систем — отсутствие интерактивности. В других программных продуктах, таких как Cat5Data, F5Data, TornadoTarget, реализована интерактивная 2D-визуализация спутниковых и аэроснимков, погодных карт. Существуют и системы численного представления данных и данных на графиках: Lightning/2000, LNM View [2], Weather Display [3].

Данные программы предназначены для интерпретации прогнозов погоды и отражения текущих показаний датчиков. Для задач моделирования и предсказания сложных погодных ситуаций, таких как тропические циклоны (циклоны, ураганы), требуются более совершенные методы 3D-визуализации. В некоторых системах для этих целей используются пакеты 3D-визуализации общего назначения, такие как VolPack [4]. Недостатком подобного подхода является отсутствие специализированных программных инструментов предварительной обработки и анализа визуализируемых данных и ограничения на форматы обрабатываемых данных. Поэтому сохраняется актуальность создания специализированных систем 3D-визуализации синоптических данных.

Из немногочисленных существующих систем такого ранга следует выделить систему CAMVis от NASA [5, 6] и "Систему визуализации ураганов в 3D-окружении" с использованием системы стереовидения VERTEX при поддержке проекта NOAA [7]. Данные системы отличаются высокой степенью информативности и качеством визуализации с применением суперкомпьютерных вычислительных мощностей. Однако они являются уникальными и не тиражируемыми, поскольку характеризуются достаточно сложной программной архитектурой с привязкой к специализированному вычислительному оборудованию и учетом потребностей, диктуемых спецификой форматов данных и решаемых задач в конкретном центре мониторинга. В качестве вычислительного оборудования используются дорогостоящие суперкомпьютеры и кластеры ЭВМ. Как видно, подобные системы имеют узкую направленность и "заточенность" под нужды центров мониторинга.

В настоящей статье представлена система визуализации синоптических объектов, которая является развитием работы авторов, представленной в работах [8, 9]. Система ориентирована прежде всего на визуальный анализ тропических циклонов и эксплуатируется на обычной вычислительной технике. Необходимость в создании системы и требования к ней были продиктованы потребностями центра спутникового мониторинга в ИАПУ ДВО РАН.

Основными требованиями к системе были:

- режим пространственно-временной анимации;
- высокая наглядность и возможность визуального анализа динамики циклонов;
- работа с большими объемами данных с высокой производительностью;

[1] Работа выполнена при финансовой поддержке РФФИ (проект 11-07-98504-р_восток_а) и Президиума РАН (Программа 2).

- интерактивные расчеты вспомогательных параметров;
- возможность объединения данных с разных спутников;
- возможность работы с различными специализированными форматами входных данных;
- устойчивость к зашумленным данным.

Основной вклад предлагаемой работы в решение проблемы визуализации синоптических объектов, на взгляд авторов, состоит в следующем:

- разработка способа и алгоритмических средств анимационной визуализации циклонов с помощью вычисляемых аномалий физических полей;
- использование шейдеров и технологии CUDA для обеспечения режима анимации и ускорения прикладных вычислений;
- реализация рабочей версии системы визуализации синоптических объектов, ориентированной на визуальный анализ динамики циклонов, с учетом требований специалистов спутникового центра ДВО РАН.

Анимация циклонов

Входные данные, восполнение. Для визуализации циклонов используются данные спутниковых измерений пространственных физических полей в атмосфере (температура, давление, влажность) в районах прохождения циклона. Чтобы отследить динамику развития циклона, необходимо, чтобы циклон находился в зоне видимости спутников от момента зарождения и до его распада с учетом того, что циклон движется. После первичной обработки спутниковые данные представляются в виде профилей (регулярная решетка) по уровням высоты над землей. При работе с данными возникают две трудности. Первая трудность традиционная — отсутствие значений в некоторых точках и наличие ошибочных измерений. Вторая трудность связана с тем, что орбитальные спутники не способны обеспечить требуемую непрерывную "видимость" циклона, поскольку спутники двигаются по фиксированным орбитам, а траектория движения циклона непредсказуема. Поэтому необходимо использовать совместные/объединенные измерения двух и более спутников, например спутников NOAA 15, 16, 18, 19.

Однако два соседних спутниковых измерения, как правило, разделены не только во времени, но могут относиться и к различным регионам. Временные интервалы между измерениями могут изменяться от 30 мин до 12 ч. Если предположить, что циклон за такие интервалы времени меняется незначительно, то можно применить приемы пространственно-темпорального восполнения данных.

Рассматривались различные существующие методы экстраполяции и интерполяции данных. Эксперименты показали, что в данном случае можно ограничиться простыми методами — восполнение "по соседу" и линейная интерполяция. При этом применялись два способа совместного использования данных: объединение измерений и пересечение измерений. Каждый из них имеет свои очевидные преимущества и недостатки.

Вычисление аномалий, анимация в пространстве и времени. Для наглядного представления циклона в динамике предлагается использовать предварительно вычисленные карты температурных аномалий тропического циклона. Под аномалией понимается отклонение от среднего [10]. Анализ аномалий в пространстве и времени позволяет метеорологам судить о физике процесса.

Если известно, что профили в районе содержат циклон и положение центра циклона также известно, то на каждом уровне высоты можно построить среднее значение параметра (нормальное состояние). Для вычисления среднего берутся значения из области периферии циклона, так как известно, что на периферии циклона состояние параметров близко к нормальному. Формально на каждом уровне высоты для вычисления среднего используются данные внутри кольца с внутренним радиусом 250 км от центра, и внешним радиусом 500 км. Кольцо ограничивается радиусом 500 км, так как далее могут происходить процессы, не относящиеся к исследуемому циклону (рис. 1, см. третью сторону обложки). Аномалия на уровне вычисляется как разность измеренного значения и среднего значения. Затем для каждого уровня строится одномерная функция зависимости величины аномалии от расстояния до центра циклона. Таким образом, мы получаем распределение аномалий по высоте и удаленности от центра циклона (рис. 2).

Визуализация ядра циклона по карте аномалий. При визуализации аномалий четко выделяется область, которая является ядром циклона. В ядре цик-

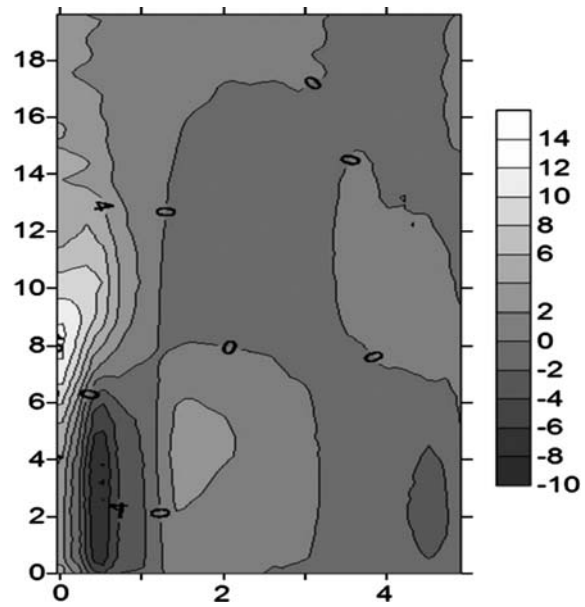


Рис. 2. Распределение аномалий в зависимости от радиуса (×100 км) и высоты (км)

лона сконцентрированы максимальные аномалии. Для визуализации ядра циклона используется пространственная регулярная решетка, центр которой помещается в центр циклона. Каждая ячейка решетки заполняется значением из карты аномалий в зависимости от удаленности от центра циклона. Размер решетки определяет компромисс между качеством визуализации и трудоемкостью вычислений. В приведенном ниже примере оптимальным выбором была решетка 64 × 64 × 64.

Алгоритмы визуализации синоптических данных. В настоящей версии системы применяются следующие алгоритмы анимационной визуализации скалярных полей:

- алгоритм объемной текстурной визуализации;
- алгоритм объемной многочастичной визуализации;
- алгоритм визуализации изоповерхностей;
- алгоритм трассировки на шейдерах.

Три первые из перечисленных были разработаны авторами в предыдущей версии системы. Ниже дается их краткая характеристика, а более подробное их описание можно найти в работе [9].

Алгоритм объемной текстурной визуализации базируется на использовании библиотеки OpenGL и аппаратно поддерживаемых 3D-текстур. Поле скалярных данных по определенной схеме интерпретируется как 3D-текстура, и графический акселератор берет на себя дальнейшую интерполяцию данных между ячейками сетки.

Идея алгоритма объемной многочастичной визуализации заключается в отображении на экран частиц в соответствии со значением скаляра в рассматриваемых точках объема. Для реализации наглядности отображения скалярного поля варьируются следующие параметры: цвет частиц, освещенность, вероятность появления. Реализуется анимация с постоянной сменой положений частиц (мельтешение), в результате чего пользователь может судить о структуре скалярного поля по характеру распределения частиц на последовательности кадров.

Для конвертирования воксельного представления изоповерхностей в полигональное представление используется метод "марширующих кубиков" [11]. Это обеспечило аппаратную поддержку визуализации изоповерхностей скалярных полей.

Алгоритм трассировки объемов на шейдерах. Алгоритм основывается на применении 3D-текстур, текстуры ColourLookUpTable (CLUT) и шейдер-технологии. На первом этапе выполняется загрузка данных в видеопамять. Данные преобразуются к формату текстур с учетом того, что загружаемая в видеопамять текстура должна содержать значения в интервале [0, 1] по каждой компоненте. Для этого выполняется нормировка вычисляемых градиентов на скалярном поле и смещение значений на 0,5. Поле скаляров также приводится к интервалу [0, 1]. Таким образом, в каждой точке воксельной решетки имеются три значения вектора градиента и одно значение скалярного поля. После этого данные загружаются в четырехкомпонентную 3D-текстуру, где первые три компоненты определяют нормаль в точке, а четвертая — нормированное значение скалярного поля. На втором этапе в видеопамять загружается текстура CLUT. Создание текстур CLUT происходит непосредственно в приложении с помощью CLUT-редактора (рис. 3, см. третью сторону обложки).

Далее выполняется загрузка в графический ускоритель вершинного и пиксельного шейдера. Поскольку известны физические размеры скалярного поля, есть привязка к системе координат меркаторской проекции. Операциями масштабирования и сдвига в буфер кадра вводится куб со стороной ребра, равной единице, при этом конвейер OpenGL для закраски куба использует ранее загруженные шейдеры. OpenGL настраивается таким образом, что видны только те грани куба, которые находятся ближе к наблюдателю. В результате, шейдерная программа получает точку входа, преобразованную в систему координат текстуры. Теперь следует в вершинном шейдере вычислить вектор наблюдения, т. е. вектор от точки наблюдателя до точки входа. Источник освещения располагается в позиции наблюдателя, при этом вектор наблюдения и вектор освещения совпадают, что существенно сокращает время расчетов. Во фрагментном шейдере происходит накопление цвета по принципу "от ближнего к дальнему" до тех пор, пока луч не выйдет за пределы куба или накопленная прозрачность не достигнет значения "пиксель непрозрачный". На каждом шаге накопления цвета из текстуры данных считывается значение преобразованного градиента и скаляра в точке.

Рекуррентная формула расчета интенсивности цвета в пикселе:

$$I_{k+1} = I_k + (1 - \alpha_k)\alpha_{CLUT}\frac{|\mathbf{L}|}{N}\left(C_{CLUT}\left|\left(\mathbf{n}\frac{\mathbf{L}}{|\mathbf{L}|}\right)\right| + C_D\left(\mathbf{n}\frac{\mathbf{L}}{|\mathbf{L}|}\right)^4\right)$$ — интенсивность цвета в пикселе на шаге $k + 1$ для $k \in [0, N]$;

$\alpha_{k+1} = a_k + (1 - \alpha_k)\alpha_{CLUT}\frac{|\mathbf{L}|}{N}$ — прозрачность в пикселе на шаге $k + 1$ для $k \in [0, N]$;

$I_0 = 0$ и $\alpha_0 = 0$ — на первом шаге пиксель полностью прозрачный и интенсивность цвета в пикселе равна нулю;

N — число шагов вдоль отрезка $\mathbf{L}$ (выбирается пользователем);

$\mathbf{L}$ — вектор от точки входа луча зрения в бокс сцены до его выхода (в СК текстуры);

C_{CLUT} — цвет вокселя из CLUT;

$\mathbf{n}$ — нормаль в вокселе;

C_D — цвет зеркального блика задается пользователем.

Для повышения информативности визуализации C_D можно задавать двумя различными значениями в зависимости от знака выражения $(\mathbf{n} \cdot \mathbf{L})$. Таким образом, блики для вокселей с нормалью, направленной на наблюдателя, будут окрашиваться одним цветом, а с нормалью, направленной от наблюдателя, будут окрашиваться другим цветом.

Особенности совместной работы нескольких алгоритмов визуализации. При одновременной работе нескольких алгоритмов визуализации с использованием Z-буфера могут возникать ситуации некорректного отображения результатов рендеринга. Чтобы избежать такой некорректности при совместной работе метода трассировки лучей с другими алгоритмами, необходимо обеспечивать правильную последовательность выполнения: алгоритм трассировки должен отрабатывать после работы всех других алгоритмов. Это необходимо, чтобы до трассировки лучей был сформирован Z-буфер. Непосредственно в алгоритме трассировки при накоплении цвета на каждой итерации движения по лучу выполняется проверка: находится ли очередная точка к наблюдателю ближе, чем фрагмент, занесенный в буфер кадра другими алгоритмами.

При реализации анимации во времени расчет параметров для визуализации (вычисление текстур, градиентов, minmax значений скалярных полей и построение полигональных моделей) выполняется "на лету". Анимация в пространстве поддерживается стандартным образом за счет вышеупомянутых алгоритмов.

Следует отметить, что модульная структура системы позволяет легко расширять базу алгоритмов визуализации.

Многопроцессорная обработка

Одним из требований к системе была возможность ее эксплуатации не только на многопроцессорной вычислительной системе, но и на персональных ЭВМ со штатным аппаратным обеспечением. Поэтому при разработке системы учитывались следующие направления ускорения вычислений и поддержки режима реального времени: использование многоядерных центральных процессоров (ЦП), использование графических ускорителей с шейдери CUDA технологиями.

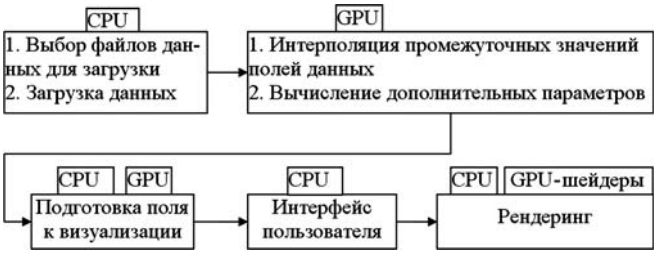


Рис. 4. Распределение вычислений между центральным процессором и графическим ускорителем

Поскольку физические поля заданы в узлах регулярных 3D-решеток, операции над ними носят сеточный характер, что удобно для организации параллельных вычислений. Анализ показал, что этапы обработки синоптических данных в зависимости от их трудоемкости и подверженности распараллеливанию могут быть распределены между центральным процессором CPU и графическим ускорителем GPU следующим образом (рис. 4). Этап загрузки, восполнения данных и вычисления аномалий выполняется на CPU в многопоточном режиме. Интерполяция и вычисление дополнительных параметров выполняется на CUDA-ядре GPU. Подготовка данных к визуализации распределяется между CPU и GPU. Визуализация эффективно осуществляется средствами шейдер-технологии.

К дополнительным параметрам, которые вычисляются на регулярной сетке, относятся давление насыщенного водяного пара, относительная влажность, абсолютная влажность, массовая доля водяного пара, связь парциального давления и точки росы, массовая доля водяного пара.

Поскольку в каждой ячейке сетки рассчитываются несколько параметров (температура, влажность, давление и др.), все вычисления в ячейке можно разделить на независимые нити. Скорость вычислений на CUDA в данном случае будет сильно зависеть от эффективности использования операций чтения/записи в глобальную память GPU, так как это одна из самых затратных операций. Важной особенностью GPU является возможность объединения нескольких обращений к глобальной памяти в одну операцию над блоком (транзакцию). Для этого нужно обеспечить выполнение четырех условий:

1) нити должны обращаться к 32-битовым словам, давая при этом в результате один 64-байтовый блок;

2) общий блок обязательно должен быть выровнен в памяти;

3) все 16 слов должны лежать в пределах данного блока;

4) нити должны обращаться к словам последовательно, т. е. k-я нить должна обращаться к k-му слову.

Словом в нашем случае является 32-битовое число с плавающей запятой. Условие 2 выполняется еще при загрузке данных в GPU. Сетка блоков для CUDA-ядра формируется таким образом, чтобы обработке подверглись все данные 3D-сетки. А сетка нитей в блоке имеет размерность $16 \times X$. Организация блока, кратного 16 потокам, обеспечивает выполнение условия 1 и 3. X выбирается таким образом, чтобы максимально эффективно использовать регистры мультипроцессора при вычислении конкретных параметров, так как каждый параметр имеет свою формулу и, таким образом, свою сложность. Последнее условие также обеспечивается

тем, что последовательно расположенные нити могут обрабатывать последовательные ячейки сетки.

Эксперименты показали, что указанное распределение вычислительной нагрузки между CPU и GPU носит оптимальный характер.

Структура системы

В системе обеспечиваются следующие графические функции:

- темпоральные и пространственные срезы;
- гибкое управление интерфейсом;
- интерактивное управление модулями загрузки и визуализации;
- интерактивная обработка данных.

К инструментарию относятся следующие режимы:

- управление камерой ("Arcball", свободный полет, плоский режим, автоматическое слежение за динамическими объектами);
- управление данными (отсечение скаляров, фильтрация);
- управление визуализацией;
- управление динамическими полями.

Структура системы представлена на рис. 5.

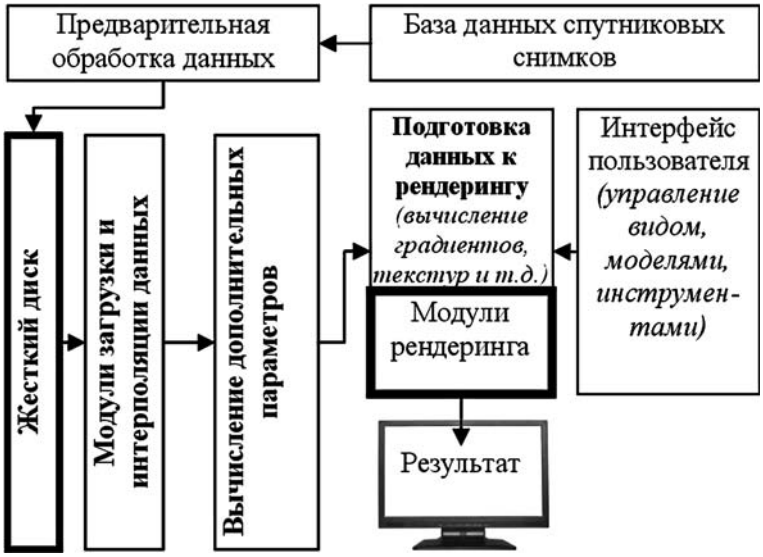


Рис. 5. Структура системы

Работа системы на реальных данных

Работа системы проиллюстрирована на примере визуализации реальных данных тропического циклона Melor со временем жизни с 29.09.2009 по 10.10.2009. Профили циклона получены по данным AMSU спутников NOAA 15, 16, 18, 19. Траектория тайфуна взята из базы данных KITAMOTO Asanobu @ National Institute of Informatics [12]. Примененная для визуализации температурных аномалий ядра циклона палитра цветов CLUT показана на рис. 6 (см. третью сторону обложки).

Желтым цветом показаны аномалии выше 4,5 K, красным цветом — аномалии от 1 до 4,5 K, синим цветом — аномалии от −4,5 до −1 K, зеленым цветом — аномалии ниже −4,5 K. Шкала серого цвета определяет прозрачность. Траектория циклона показана на рис. 7 (см. третью сторону обложки) [12].

На рис. 8—10 (см. четвертую сторону обложки) представлены кадры анимации, полученной в результате работы системы. Анимация показывает динамику циклона Melor от его зарождения и развития до распада. При анимационной визуализации динамики циклона хорошо видно суточное "дыхание" циклона. При прохождении над японскими островами отчетливо видно формирование глаза циклона (зона спокойствия). Ярко выраженное теплое ядро (красного цвета) находится на 20-м уровне, что соответствует высоте 10 км. Также в тропосфере отчетливо видно формирование температурных аномальных зон над ядром циклона.

Заключение

Представлена действующая версия системы визуализации физических полей синоптических объектов, ориентированная на визуальный анализ тропических циклонов. Эксперименты показали эффективность выбранного способа отображения динамики циклонов с помощью визуализации аномалий полей. Реализованные возможности многопроцессорной обработки прикладных данных по технологии CUDA и рендеринга на шейдерах обеспечили приемлемую скорость обработки больших объемов спутниковых данных и режим анимационной визуализации. В настоящее время система используется в центре спутникового мониторинга в ИАПУ ДВО РАН в режиме опытной эксплуатации для визуального контроля результатов первичной обработки оперативных спутниковых измерений и для исследования тропических циклонов на ретроспективных данных. Развитие системы планируется в нескольких направлениях:

- расширение базы алгоритмов визуализации;
- реализация системы на гибридной многопроцессорной архитектуре с использованием множества графических плат;
- адаптация системы к требованиям специалистов Росгидромета.

Список литературы

1. URL: http://www.met.fu-berlin.de/terra3d/en/index.htm
2. URL: http://www.thiesclima.com/soft_e.htm
3. URL: http://www.weather-display.com/index.php
4. URL: http://graphics.stanford.edu/software/volpack
5. **Green B., Henze C., Shen B.-W.** Development of a scalable concurrent visualization approach for high temporal- and spatial-resolution models // AGU 2010 Western Pacific Geophysics Meeting, Taipei, Taiwan, June 22—25, 2010 (submitted).
6. **Shen B.-W., Bryan G., Tao W.-K., Henze C., Cheung S., Li J.-L. F., and Mehrotra P.** Coupling NASA Advanced Multi-Scale Modeling and Concurrent Visualization Systems for Improving Predictions of Tropical High-Impact Weather (CAMVis) // Earth Science Technology Forum (ESTF) 2010. Arlington, Virginia, June 22—24, 2010.

7. **Wang Shao Rong, Rui You, Sheng Li, Yi Song Chen, Guo Ping Wang.** Realtime Visualization of Hurricane in Distributed Environment // Applied Mechanics and Materials. 2010. Vol. 40—41. P. 948—954.
8. **Мельман С. В., Бобков В. А.** Система визуализации пространственных полей синоптических объектов // Материалы конференции "Графикон-2007". 2007. С. 264—268.
9. **Мельман С. В.** Визуализация объемов в задачах анализа физических полей синоптических объектов // Информационные технологии. 2008. № 1. С. 62—66.
10. **Kidder S. Q., Goldberg M. D., Zehr R. M., DeMaria M., Purdom J. F. W., Velden C. S., Grody N. C., and Kusselson S. J.** Satellite analysis of tropical cyclones using the Advanced Microwave Sounding Unit (AMSU) // Bulletin of the American Meteorological Sociey. 2000. N 81. P. 1241—1259.
11. **Lorensen W. E., Cline H. E.** Marching Cubes: a high resolution 3D surface reconstruction algorithm // Conference Proceedings "SIGGRAPH 87". 1987. P. 163—169.
12. URL: http://agora.ex.nii.ac.jp/digital-typhoon/news/2009/TC0918/index.html.en

УДК 519.862.8, 519.863.3, 519.8:33, 519.853.32, 519.854.6

Ю. А. Мезенцев, канд. экон. наук, доц.,
П. С. Павлов, аспирант,
Новосибирский государственный технический университет,
e-mail: mesyan@yandex.ru

К программной реализации декомпозиционного алгоритма решения одного класса задач дискретной оптимизации с полуопределенной релаксацией

Рассмотрена универсальная экономико-математическая модель, предназначенная для определения оптимальных стратегий управления подсистемами (компонентами подсистем) логистики предприятий. Разработан приближенный эффективный алгоритм решения оптимизационных задач реальной размерности, имеющих высокую вычислительную сложность.

Ключевые слова: *декомпозиция, задача полуопределенного программирования, дискретная оптимизация, релаксация, эффективный алгоритм*

Введение

В работе [1] приведены содержательная и формальная постановки дискретной динамической задачи оптимального управления поставками и сбытом торгового предприятия. Разработан и специальный алгоритм решения данной оптимизационной нелинейной целочисленной задачи.

Настоящая работа посвящена развитию данной темы, уточнению формальной постановки, разработке декомпозиционного алгоритма ее решения, доведения его до программной реализации и анализу эффективности применения для реальных объектов.

Задачи подобного класса возникают ежедневно на многих предприятиях, что предопределяет востребованность инструментальных средств их решения. Приведем краткую содержательную постановку задачи.

1. Содержательная постановка задачи

Объектом управления является предприятие, в качестве основной деятельности которого рассматривается оптово-розничная торговля. Собственное производство также может иметь место, однако производственные подразделения являются подчиненными по отношению к сбытовым. Управляющие воздействия: выбор поставщиков товаров, определение объемов закупок по всему ассортиментному списку, определение цен и объемов сбыта.

Транспортные издержки по перечисленным причинам можно считать несущественными. От отдаленности поставщиков, таким образом, зависит только время доставки груза, которое компенсируется необходимым уровнем запасов на складе. Существенными можно считать условия поставок, характеризующиеся наличием оптовых скидок, зависимость которых от объемов поставок в стоимостном выражении имеет ступенчатый характер [1, 2].

С учетом перечисленных обстоятельств задачу управления можно сформулировать следующим образом.

Необходимо обеспечить такую стратегию закупок (выбора поставщиков, объемов поставок с учетом скидок), а также ценовую политику продаж по группам потребителей, которая максимизирует критериальный показатель (например чистый доход на конец планового периода). При этом необходимо учитывать ограничения на оборотные средства на начало периода и емкость склада. Под термином "стратегия закупок", таким образом, будем понимать совокупность планируемых объемов закупок товаров по всему ассортименту, выбираемым ценам и скидкам от всех потенциальных поставщиков, определяемую для каждого выделяемого временного интервала планового периода. Аналогично под термином "стратегия продаж" будем понимать совокупность планируемых объемов продаж и цен на товары по всему ассортименту для всех групп потребителей, определяемую на основе их спроса для каждого выделяемого временного интервала планового периода.

Ограничениями задачи также являются логические условия, учитывающие возможные скидки при закупках и продаже товаров, спрос, емкость склада и финансовые возможности фирмы в динамике по интервалам планового периода.

2. Формальная постановка задачи определения оптимальных объемов закупок, объемов и цен продаж при управлении закупками и сбытом продукции и алгоритм решения

Введем следующие обозначения:

t — номер временного интервала, с дискретностью до которого определено модельное время (далее месяца);

j — номер поставщика ($j = \overline{1, J}$); i — номер продукта в ассортименте поставок ($i = \overline{1, I}$); l — индекс типа потребителя ($l = \overline{1, L}$); k — номер интервала шкал объемов скидок ($k = \overline{1, K}$) и спроса ($k = \overline{1, K'}$);

$y_{i,j}(t)$ — объем закупок в натуральном выражении продукта i у поставщика j в месяце t;

$O_i(t)$ — остаток на складе продукта i в месяце t;

$C_{i,j}(t)$ — базовая оптовая цена продукта i у поставщика j в месяце t;

$d_j(t)$ — объем закупок в стоимостном выражении у поставщика j в месяце t по базовой цене (без учета скидок);

$h_{j,k}(t)$ — значение правой границы интервала k шкалы объемов скидок у поставщика j в месяце t;

$g_{j,k}(t)$ — соответствующая интервалу k шкалы объемов скидка (в процентах) у поставщика j в месяце t;

$w_{j,k}(t)$ — индикатор попадания объема закупок в интервал k шкалы скидок у поставщика j в месяце t;

$x_{i,l,k}(t)$ — объем продаж в натуральном выражении продукта i потребителю типа l в месяце t на интервале k шкалы функции спроса;

$p_{i,l,k}(t)$ — цена продажи единицы продукта i для потребителя типа l в месяце t на интервале k функции спроса;

$Q(t)$, $\Delta Q(t)$ — размер и остаток оборотного капитала в месяце t;

$N(t)$ — заработная плата и накладные расходы в месяце t;

$s_{i,l,k}(t)$ — значение правой границы интервала k шкалы функции спроса на товар i у потребителя типа l в месяце t.

В работе [1] сформулирована дискретная динамическая задача оптимального управления поставками и сбытом торгового предприятия и разработан специальный декомпозиционный алгоритм ее приближенного решения.

В формальной постановке данной задачи не учитывается следующее обстоятельство. Определение цены продаж — прерогатива объекта управления. Поэтому функция спроса зависимости объемов продаж от цен может существенно отличаться от ступенчатой [1, 2] (пример на рисунке). Единственно, что можно предполагать с большой долей уверенности — попадание реальной цены в некоторый интервал. Обычно такие интервалы могут быть заданы для каждой товарной позиции, для каждого временного интервала и каждой категории покупателей.

Зависимость цен и доходов от объемов продаж в традиционном понимании можно отобразить графически (см. рисунок).

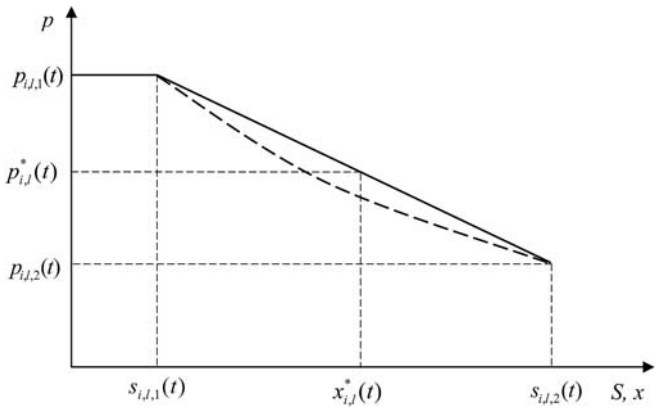
В данном примере предполагается, что при значении верхней границы дохода $p_{i,l,1}(t) = 3$ ед. за единицу продукта, объем продаж составит $s_{i,l,1}(t) = 100$, а при доходе $p_{i,l,2}(t) = 2$ ед. за единицу продукта (нижняя граница) $s_{i,l,2}(t) = 200$ единиц. В первом случае общий доход в 300 ед. оказывается меньше, чем во втором (400 ед.). Любое другое (отличное от граничных) значение объема продаж может быть получено при соответствующей корректировке цен (и доходов за единицу продукта).

Например, если положить $s_{i,l,1}(t)$ минимальный спрос при максимальном значении дохода $p_{i,l,1}(t)$ за единицу, а $s_{i,l,2}(t)$ — максимальный спрос при минимальном доходе $p_{i,l,2}(t)$ за единицу продукта, то любое значение дохода можно вычислить, как линейную комбинацию

$$p_{i,l}(t) = \alpha_{i,l} p_{i,l,1}(t) + (1 - \alpha_{i,l}) p_{i,l,2}(t),\ 0 \leqslant \alpha_{i,l} \leqslant 1,$$
$$i = \overline{1, I},\ l = \overline{1, L}. \qquad (1)$$

Аналогично вычисляется и объем продаж.

Линейная функция спроса (см. рисунок) не лучшим образом отображает фактическую зависимость объема продаж от цены. Однако более качественные зависимости объема продаж продукта от дохода за единицу чаще всего получить невозможно, имея в виду немалые наборы ассортиментных позиций, характерные для большинства реализаций рассматриваемой задачи управления. Такие зависимости теоретически нелинейны (пример изображен штри-



Зависимости цены от объема продаж продукта

ховой кривой на рисунке). К тому же, даже линейные зависимости, интегрированные в рассматриваемые задачи управления с учетом размерностей, делают эти задачи фактически неразрешимыми точными методами.

Рассмотрим группу ограничений на динамику чистого дохода [1]:

$$\Delta Q(t+1) = \Delta Q(t) + \sum_{i=1}^{I}\sum_{l=1}^{L}\sum_{k=1}^{K} p_{i,l,k}(t)x_{i,l,k}(t) - N(t) - \sum_{i=1}^{I}\sum_{j=1}^{J}[1-\tilde{g}_j(t+1)]C_{i,j}(t+1)y_{i,j}(t+1). \quad (2)$$

Если теперь вместо набора $p_{i,l,k}(t)$ ввести в рассмотрение зависимости (1), то получим квадратичные функции многих переменных:

$$\Delta Q(t+1) = \Delta Q(t) + \sum_{i=1}^{I}\sum_{l=1}^{L}[\alpha_{i,l}p_{i,l,1}(t) + (1-\alpha_{i,l})p_{i,l,2}(t)]x_{i,l}(t) - N(t) - \sum_{i=1}^{I}\sum_{j=1}^{J}[1-\tilde{g}_j(t+1)]C_{i,j}(t+1)y_{i,j}(t+1),\ t=\overline{1,T},$$

где $0 \leqslant \alpha_{i,l} \leqslant 1$ — дополнительные переменные, подлежащие определению. Ситуация несколько упрощается, если зависимости $p_{i,l}(t)$ от $x_{i,l}(t)$ выразить в явном виде. Предположив линейную зависимость, при $s_{i,l,1}(t) \leqslant x_{i,l}(t) \leqslant s_{i,l,2}(t)$ получим

$$p_{i,l}(t) = a^0_{i,l}(t) + a^1_{i,l}(t)x_{i,l}(t),$$
$$i=\overline{1,I},\ l=\overline{1,L},\ t=\overline{1,T}. \quad (3)$$

Коэффициенты $a^0_{i,l}(t)$, $a^1_{i,l}(t)$ определяются непосредственно из исходных данных, например с помощью решения систем уравнений:

$$\tilde{a}^1_{i,l}(t)x_{i,l}(t) + \tilde{a}^2_{i,l}(t)p_{i,l}(t) = b;$$

$$\tilde{a}^1_{i,l}(t)s_{i,l,1}(t) + \tilde{a}^2_{i,l}(t)p_{i,l,1}(t) = b;$$

$$\tilde{a}^1_{i,l}(t)s_{i,l,2}(t) + \tilde{a}^2_{i,l}(t)p_{i,l,2}(t) = b.$$

Откуда $p_{i,l}(t) = \dfrac{b}{\tilde{a}^2_{i,l}(t)} - \dfrac{\tilde{a}^1_{i,l}(t)}{\tilde{a}^2_{i,l}(t)}x_{i,l}(t)$,

$$a^0_{i,l}(t) = \frac{b}{\tilde{a}^2_{i,l}(t)},\ a^1_{i,l}(t) = -\frac{\tilde{a}^1_{i,l}(t)}{\tilde{a}^2_{i,l}(t)},$$
$$i=\overline{1,I},\ l=\overline{1,L},\ t=\overline{1,T}.$$

В общем виде искомые коэффициенты вычисляются следующим образом:

$$\tilde{a}^1_{i,l}(t) = \frac{p_{i,l,2}(t)-p_{i,l,1}(t)}{p_{i,l,2}(t)s_{i,l,1}(t)-p_{i,l,1}(t)s_{i,l,2}(t)};$$

$$\tilde{a}^2_{i,l}(t) = \frac{s_{i,l,1}(t)-s_{i,l,2}(t)}{p_{i,l,2}(t)s_{i,l,1}(t)-p_{i,l,1}(t)s_{i,l,2}(t)};$$

$$a^0_{i,l}(t) = \frac{1}{\tilde{a}^2_{i,l}(t)} = \frac{p_{i,l,2}(t)s_{i,l,1}(t)-p_{i,l,1}(t)s_{i,l,2}(t)}{s_{i,l,1}(t)-s_{i,l,2}(t)}; \quad (4)$$

$$a^1_{i,l}(t) = \frac{\tilde{a}^1_{i,l}(t)}{\tilde{a}^2_{i,l}(t)} = -\frac{p_{i,l,2}(t)-p_{i,l,1}(t)}{s_{i,l,1}(t)-s_{i,l,2}(t)}. \quad (5)$$

Тогда зависимость цены от спроса представляется в соответствии с выражением

$$p_{i,l}(x_{i,l}(t)) = \begin{cases} p_{i,l,1}(t), \text{ если } x_{i,l}(t) \leqslant s_{i,l,1}(t), \\ a^0_{i,l}(t) + a^1_{i,l}(t)\,x_{i,l}(t), \text{ если } x_{i,l}(t) > s_{i,l,1}(t), \end{cases} \quad (6)$$
$$i=\overline{1,I},\ l=\overline{1,L},\ t=\overline{1,T}.$$

В частности, в рассматриваемом примере (см. рисунок) получается следующая зависимость:

$$p_{i,l}(t) = \begin{cases} 3, \text{ если } 0 < x_{i,l}(t) \leqslant 100, \\ 4 - 0{,}01\,x_{i,l}(t), \text{ если } 100 < x_{i,l}(t) \leqslant 200. \end{cases}$$

С учетом названных обстоятельств уравнения (2) примут такой вид:

$$-\Delta Q(t+1) + \Delta Q(t) + \sum_{i=1}^{I}\sum_{l=1}^{L} x_{i,l}(t)p_{i,l}(x_{i,l}(t)) - \sum_{i=1}^{I}\sum_{j=1}^{J}[1-\tilde{g}_j(t+1)]C_{i,j}(t+1)y_{i,j}(t+1) = N(t),$$
$$t=\overline{1,T}. \quad (7)$$

Третье слагаемое в (7) в соответствии с (6) принимает следующие значения:

$$\sum_{i=1}^{I}\sum_{l=1}^{L} x_{i,l}(t)p_{i,l}(x_{i,l}(t)) = \begin{cases} \sum\limits_{i=1}^{I}\sum\limits_{l=1}^{L}[a^0_{i,l}(t)x_{i,l}(t) + a^1_{i,l}(t)(x_{i,l}(t))^2], \\ \text{если } x_{i,l}(t) > s_{i,l,1}(t), \\ \sum\limits_{i=1}^{I}\sum\limits_{l=1}^{L} p_{i,l,1}(t)x_{i,l}(t), \text{ если } x_{i,l}(t) \leqslant s_{i,l,1}(t). \end{cases} \quad (8)$$

Заменим уравнения (7) неравенствами:

$$-\Delta Q(t+1) + \Delta Q(t) + \sum_{i=1}^{I}\sum_{l=1}^{L} x_{i,l}(t)p_{i,l}(x_{i,l}(t)) -$$

$$- \sum_{i=1}^{I}\sum_{j=1}^{J} [1 - \tilde{g}_j(t+1)]C_{i,j}(t+1)y_{i,j}(t+1) \geqslant N(t),$$

$$t = \overline{1, T}\,. \qquad (9)$$

Легко убедиться, что это не приводит к изменению смысла ограничений (7). Вместе с тем, если положить в (6) $x_{i,l}(t) > s_{i,l,1}(t)$, $\forall i = \overline{1, I}$, $l = \overline{1, L}$, $t = \overline{1, T}$, то неравенства (9), в отличие от (7), удовлетворяют условиям задач полуопределенного программирования [3, 4]. Основным условием, которому должны удовлетворять ограничения задач полуопределенного программирования в данном случае являются свойства положительной полуопределенности матриц квадратичных форм, определяемых в (9) выражениями $\sum_{i=1}^{I}\sum_{l=1}^{L}[a^0_{i,l}(t)x_{i,l}(t) + a^1_{i,l}(t)(x_{i,l}(t))^2]$. Кроме этого, ограничения (9) должны образовывать выпуклый конус второго порядка с непустой внутренностью, чему и служит замена уравнений на неравенства.

Известны эффективные алгоритмы решения подобных задач большой размерности [3—5]. Поэтому наличие квадратичных ограничений в задачах оптимизации данного класса само по себе не приводит к существенному повышению трудоемкости решения.

Значительное увеличение трудоемкости влечет за собой наличие альтернативных условий (6) или (8). Более того, не существует точного эффективного алгоритма решения рассматриваемой задачи управления, дополненной условиями (6) и (8).

Тем не менее, можно предложить приближенный эффективный алгоритм. Для его изложения запишем в явном виде рассматриваемую задачу оптимизации.

$$p_{i,l}(x_{i,l}(t)) =$$

$$= \begin{cases} p_{i,l,1}(t), \text{ если } x_{i,l}(t) \leqslant s_{i,l,1}(t), \\ a^0_{i,l}(t) + a^1_{i,l}(t)x_{i,l}(t), \text{ если } x_{i,l}(t) > s_{i,l,1}(t), \end{cases} \qquad (10)$$

$$i = \overline{1, I}\,,\ l = \overline{1, L}\,,\ t = \overline{1, T}\,;$$

$$\sum_{j=1}^{J}\sum_{i=1}^{I} [1 - \tilde{g}_j(t)]C_{i,j}(t)y_{i,j}(t) + \Delta Q(t) - Q(t) = 0,$$

$$t = \overline{1, T}\,; \qquad (11)$$

$$Q(t) \geqslant 0,\ \Delta Q(t) \geqslant 0,\ y_{i,j}(t) \geqslant 0,$$

$$i = \overline{1, I}\,,\ j = \overline{1, J}\,,\ t = \overline{1, T}\,; \qquad (12)$$

$$0 \leqslant x_{i,l}(t) \leqslant s_{i,l,2}(t),\ i = \overline{1, I}\,,\ l = \overline{1, L}\,,\ t = \overline{1, T}\,; \qquad (13)$$

$$\sum_{j=1}^{J} y_{i,j}(t) + O_i(t-1) - \sum_{l=1}^{L} x_{i,j}(t) - O_i(t) = 0,$$

$$i = \overline{1, I}\,,\ t = \overline{1, T}\,; \qquad (14)$$

$$\Delta Q(t+1) - \Delta Q(t) - \sum_{i=1}^{I}\sum_{l=1}^{L} x_{i,j}(t)\, p_{i,j}(t)(x_{i,l}(t)) +$$

$$+ \sum_{i=1}^{I}\sum_{j=1}^{J} [1 - \tilde{g}_j(t+1)]C_{i,j}(t+1)y_{i,j}(t+1) \leqslant -N(t),$$

$$t = \overline{1, T}\,; \qquad (15)$$

$$z = \Delta Q(T+1) \to \max. \qquad (16)$$

Описание приближенного алгоритма оптимизации предварим рядом замечаний. Очевидно, что при принятых предположениях о функции спроса, если $s_{i,l,1}(t)p_{i,l,1}(t) \geqslant s_{i,l,2}(t)p_{i,l,2}(t)$, то продажи с доходом ниже $p_{i,l,1}(t)$ не имеют смысла, и полученные значения $x_{i,l,2}(t)$ следует обнулить.

Если же $s_{i,l,1}(t)p_{i,l,1}(t) < s_{i,l,2}(t)p_{i,l,2}(t)$ и $x_{i,l,2}(t) > 0$, то продажи $x_{i,l,1}(t) = s_{i,l,1}(t)$ с доходом $p_{i,l,1}(t)$ невозможны, и единое значение дохода $p_{i,l}(t)$ следует пересчитать в соответствии с (3). В предельном случае, когда $x_{i,l,2}(t) = s_{i,l,2}(t)$, $p_{i,l}(t) = p_{i,l,2}(t)$. При любом подобном пересчете возникают новые обстоятельства, которые влекут уменьшение оценок доходов по всем цепям купли-продажи и всем временным интервалам.

Рассмотрим алгоритм решения задачи (10)—(16), учитывающий сделанные поправки относительно цен и потребительского спроса.

Алгоритм $A_{2.1}$

1. Задаем начальное значение номера шага алгоритма $q := 0$. Используем исходные данные $O_i(1)$, $Q(1)$, $N(t)$, $C_{i,j}(t)$, $g_{j,k}(t)$, $h_{j,k}(t)$, $p_{i,l,k'}(t)$, $s_{i,l,k'}(t)$, $i = \overline{1, I}$, $j = \overline{1, J}$, $t = \overline{1, T}$, $k' = \overline{1, 2}$, $k = \overline{1, K}$, $l = \overline{1, L}$.

2. Положим $p_{i,l}(x_{i,l}(t)) := a^0_{i,l}(t) + a^1_{i,l}(t)x_{i,l}(t)$, $i = \overline{1, I}$, $l = \overline{1, L}$, $t = \overline{1, T}$ (заменяем все условия (10) на (3)), формируя исходную ослабленную задачу.

3. Находим решение задачи полуопределенного программирования (11)—(16) с дискретными переменными G^q, используя приближенный алгоритм [1, 2] и метод следования центральному пути. Оптимальное решение обозначим Y^q, X^q, z^q ($Y^q = \|y^q_{i,j}(t)\|$, $X^q = \|x^q_{i,j}(t)\|$, $z^q = z(X^q, Y^q)$).

4. Увеличиваем номер шага алгоритма $q := q + 1$.

5. Проверка условий оптимальности. Проверяем выполнение условий (10). Если выполняются все условия

$$p_{i,l}(x_{i,l}^{q-1}(t)) = \begin{cases} p_{i,l,1}(t), \text{ при } x_{i,l}^{q-1}(t) \leqslant s_{i,l,1}(t), \\ a_{i,l}^{0}(t) + a_{i,l}^{1}(t)\, x_{i,l}^{q-1}(t), \text{ при } x_{i,l}^{q-1}(t) > s_{i,l,1}(t), \end{cases}$$

$\forall i = \overline{1, I}$, $l = \overline{1, L}$, $t = \overline{1, T}$, переход к п. 8, иначе, следующий пункт.

6. Если $x_{i,l}^{q-1}(t) \leqslant s_{i,l,1}(t)$ и $p_{i,l}(x_{i,l}^{q-1}(t)) > p_{i,l,1}(t)$, заменяем стоимостную оценку переменной $x_{i,l}(t)$ фиксированным значением: $p_{i,l}(x_{i,l}(t)) := p_{i,l,1}(t)$.

Если $x_{i,l}^{q-1}(t) > s_{i,l,1}(t)$ и $p_{i,l}(x_{i,l}^{q-1}(t)) = p_{i,l,1}(t)$, заменяем фиксированную стоимостную оценку переменной $x_{i,l}(t)$ линейной функцией (3) $p_{i,l}(x_{i,l}(t)) :=$ $:= a_{i,l}^{0}(t) + a_{i,l}^{1}(t)x_{i,l}(t)$ и дополняем систему ограничений (11)—(15) условием $x_{i,l}(t) \geqslant s_{i,l,1}(t)$. Проверки и необходимые замены оценок проводятся для всех переменных $x_{i,l}(t)$ $\forall i = \overline{1, I}$, $l = \overline{1, L}$, $t = \overline{1, T}$.

7. Находим решение текущей задачи Y^q, X^q ($Y^q = \|y_{i,j}^q(t)\|$, $X^q = \|x_{i,j}^q(t)\|$) и переходим к п. 4.

8. На текущем шаге q получено оптимальное решение задачи (10)—(16) Y^*, X^*, z^*. Останов алгоритма.

Оценим трудоемкость представленного алгоритма.

Поскольку п. 6 не допускает возврата оценок переменных $p_{i,l}(x_{i,l}(t))$ к предшествующим состояниям и таких состояний для каждой переменной всего три, включая исходное, вполне очевидно, что верхняя оценка числа его шагов $\overline{q}_{2.1}$ не превышает удвоенного числа переменных $x_{i,l}(t)$, $i = \overline{1, I}$, $l = \overline{1, L}$, $t = \overline{1, T}$, т. е. $\overline{q}_{3.2} \leqslant 2ILT$. Поэтому трудоемкость $A_{2.1}$ пропорциональна $\overline{q}_{2.1}$ — кратной трудоемкости полиномиального алгоритма решения упрощенной задачи [1]. Из этого непосредственно следует полиномиальная трудоемкость (эффективность) алгоритма $A_{2.1.}$

Апостериорная же оценка $q_{2.1}$ трудоемкости алгоритма $A_{2.1}$ много меньше $\overline{q}_{2.1}$. Относительно близости решений, получаемых посредством $A_{2.1}$, к оптимальным, можно также судить по результатам численных экспериментов.

3. Программная реализация и результаты тестирования декомпозиционного алгоритма

В качестве основного вычислителя для решения последовательности релаксированных подзадач полуопределенного программирования, порождаемых алгоритмом $A_{2.1}$, использован пакет IBM ILOG CPLEX [6].

Для реализации алгоритма $A_{2.1}$ и взаимодействия с инструментами CPLEX выбран распространенный язык программирования C# и среда разработки Microsoft Visual C# 2010 Express.

Основной класс, необходимый для работы с CPLEX, называется Cplex и входит в пространство имен Cplex. Конструктор класса не принимает параметров. Доступны следующие основные методы:

- Sum — возвращает сумму переменных в переменной класса INumExpr;
- AddGe — добавляет линейное ограничение типа "больше либо равно";
- AddMaximize — задает функцию цели, ориентированную на максимум;
- Solve — старт решателя, возвращает булево значение (успешности решения).

Помимо этих методов класс предоставляет такие методы как AddLe, AddMinimize, Prod, AddEq и т. д. Для работы с классом Cplex необходимо использовать классы INumExpr, INumVar. INumExpr инкапсулирует выражения, используется для хранения линейных ограничений задачи, а также как временная переменная в составе сложных линейных выражений. INumVar содержит переменные решения модели; многомерные переменные должны быть заданы как массивы массивов типа INumVar.

Приведем примеры использования перечисленных выше классов. Создадим переменную решения $y_{i,j}(t)$ (по условиям задачи — объем закупок у поставщиков имеет три измерения: товар—поставщик—время):

```
var y = new INumVar[_data.I][][];
for (int i = 0; i < _data.I; i++){
    INumVar[][] yJ = new INumVar
    [_data.J][];
    for (int j = 0; j < _data.J; j++){
      yJ[j] = cplex.NumVarArray(_data.T,
      LowerBound,
        UpperBound, NumVarType.Float);}
    y[i] = yJ;}
```

Класс INumExpr в прикладных задачах наиболее часто используется для хранения промежуточных значений. В качестве примера рассмотрим реализацию ограничения:

$$x_{i,l,k}(t) \leqslant s_{i,l,k}(t) - \sum_{k'=1}^{k-1} x_{i,l,k'}(t),$$

$$i = \overline{1, I},\ l = \overline{1, L},\ k = \overline{1, K},\ t = \overline{1, T},$$

Номер группы тестовых задач	Число непрерывных переменных	Число булевых переменных	Число линейных ограничений	Число нелинейных ограничений	Среднее число итераций алгоритма	Среднее время решения, с
1	648	18	546	81	5	9,9
2	675 000	30	345 006	22 560	10	2605,1
3	405 000	30	207 006	13 560	8	1205, 4
4	189 000	30	96 606	6360	8	421, 7
5	54 000	30	27 606	1860	7	51,4
6	6750	30	3456	285	8	21,7

```
for (int t = 0; t < _data.T; t++){
for (int l = 0; l < _data.L; l++){
for (int i = 0; i < _data.I; i++){
for (int k = 0; k < _data.K_; k++){
    INumExpr sumXByK = cplex.NumExpr();
for (int k_ = 0; k_ <= k – 1; k_++){
    sumXByK = cplex.Sum(sumXByK,
    x[i][l][k_][t]);}
    cplex.AddLe(x[i][l][k][t],cplex.Sum(
    _data.s[i][l][k][t], cplex.Negative
    (sumXByK)));}}}}
```

Зададим целевую функцию:

```
cplex.AddMaximize(cplex.ScalProd(_data.alpha,
deltaQ));
```

В данном случае задается целевая функция, максимизирующая скалярное произведение массива значений alpha и переменной решения deltaQ.

Для запуска решателя необходимо вызвать метод класса Cplex Solve(). Значение целевой функции хранится в свойстве ObjValue.

Заметим, что для сопряжения общей среды исполнения (описание алгоритма на языке C#) и CPLEX при реализации $A_{2.1}$ необходимо получать промежуточные значения целевой функции и отдельных переменных решения. Для этого существует метод GetValue, принимающий в качестве параметра либо переменную типа INumExpr, либо переменную типа INumVar:

```
var xValue = cplex.GetValue(xILT[i][l][t]);
```

Программа реализована как приложение WPF, вызывающее библиотеку класса, содержащую указанные алгоритмы в отдельном потоке. В качестве входных данных программа может принимать как файлы, содержащие точные условия задачи, так и файлы, описывающие рамки генерации переменных (данный режим создан для тестирования алгоритма).

Результатами работы программы является набор многомерных массивов значений переменных и полученное значение целевой функции. Структура результатов такова, что их можно представить в виде OLAP куба. Использование платформы. NET позволяет достаточно легко реализовать полноценный OLAP с помощью сервисов MS SQL Server. В настоящее же время в режиме тестирования вывод осуществляется в CSV-файл.

В таблице сведены результаты расчетов шести групп сгенерированных тестовых задач различных размерностей (по пять задач в группе). Размерности задач внутри группы одинаковы. Размерность задач из группы 2 соответствует реальному объекту.

Конфигурация системы: процессор Intel Pentium 4 2,4 ГГц, оперативная память 512 Мбайт, операционная система Microsoft Windows XP Professional Service Pack 3, .NET Framework 4. В таблице приведены результаты экспериментов.

Заключение

Вычислительные эксперименты с программной реализацией алгоритма $A_{2.1}$ решения формально неразрешимой точными методами задачи (10)—(16) показали высокую эффективность алгоритма. На малых размерностях (тесты группы 1) удалось получить апостериорную относительную оценку максимального отклонения от оптимума, равную 5 %. Оценка быстродействия алгоритма также вполне обнадеживает. Так, задача с 675 000 переменными при использовании современного шестиядерного процессора считается около 3,5 мин. Это обусловливает блестящие перспективы внедрения разработки, поскольку, как отмечалось выше, поиск оптимальных решений рассматриваемой задачи управления актуален для большей части хозяйствующих субъектов.

Список литературы

1. **Мезенцев Ю. А.** Математические модели управления подсистемами логистики на предприятиях // Автоматизация и современные технологии. 2008. № 8. С. 46—55.
2. **Мезенцев Ю. А.** Об одной задаче математической логистики // Сб. трудов XV международной открытой конференции "Современные проблемы информатизации в экономике и обеспечении безопасности". Воронеж: Научная книга, 2010. Вып. 15. С. 22—29.
3. **Vandenberghe L., Byond S.** Semidefinite programming // SIAM Review. 1996. N 38. P. 49—95.
4. **Luo Z.-Q., Sturm J. F., Zhang S.** Superlinear convergence of a symmetric primal-dual path following algorithm for semidefinite programming // SIAM J. Optim. 1998. N 8. P. 59—81.
5. **Мезенцев Ю. А.** Декомпозиционный метод решения одного класса задач оптимального проектирования // Научный вестник НГТУ. 2006. № 3(24). С. 67—100.
6. **IBM ILOG CPLEX V12.1** User's Manual for CPLEX. IBM Corporation, 2009. 952 с.

УДК 65.0

В. В. Топка, канд. техн. наук, ст. науч. сотр.,
Институт проблем управления
им. В. А. Трапезникова РАН,
e-mail: topka3@mail.ru

Управление стоимостью проекта с учетом показателя его надежности

Предложена модель надежности сетевого проекта как сложной технической системы, где в качестве надежности принимается полученная ее гарантированная оценка снизу. На этой основе сформулирована задача календарного планирования не только с ресурсными ограничениями, но и с учетом показателя надежности проекта. Рассмотрена задача минимизации стоимости проекта при ограничениях на время и оценку надежности при линейных связях между переменными в детерминированной конъюнктивной модели проекта, в которой исходные данные известны неточно. Для ее решения предложено использовать регуляризованный двухшаговый метод линеаризации для некорректных задач при неточных исходных данных.

Ключевые слова: *управление проектами, сеть, надежность, оптимизация*

Введение

За последние годы значительное развитие получили методы решения задач календарного планирования с ограниченными ресурсами — *RCPSP* (*Resource-Constrained Project Scheduling Problem*). В этих задачах минимизируется продолжительность выполнения проекта при ограничениях на ресурсы или стоимость проекта либо минимизируется стоимость проекта или объем используемых ресурсов при ограничении на время выполнения проекта в условиях, как правило, детерминированной сетевой модели проекта. Рассматриваемые понятия и классификация приведены в [1], а достаточно полный обзор работ содержится в [2, 3].

В [4] показано, что *RCPSP*, как обобщение задачи о загрузке оборудования, является сильно NP-трудной задачей, в [5] показана NP-полнота проблемы существования допустимого расписания. Предложено немало алгоритмов решения этих задач, которые можно разделить на два класса: оптимизационные и эвристические. Среди оптимизационных методов можно выделить методы булева программирования, динамического программирования, потокового программирования, ветвей и границ, последний из которых затем был наиболее широко использован. Однако ввиду NP- трудности таких задач для их решения при реальной размерности проекта (более 60 работ) необходимо применение различных эвристик.

Ядром большинства эвристических методов решения *RCPSP* являются схемы генерации расписания, которые разделяются на последовательные и параллельные. Согласно последовательной схеме на каждом этапе выбирается одна работа с наиболее ранним временем выполнения, допустимым по ресурсным и сетевым ограничениям предшествования. В параллельной схеме, которая выполняет приращение времени, каждая итерация состоит из двух шагов:

- определение следующего момента времени в расписании и соответствующих множеств выполненных, активных и ожидающих выполнения работ;
- включение в расписание работ из множества ожидающих работ.

Правила приоритетов используют одну или обе схемы генерации расписаний для построения одного или более расписаний. А сами правила приоритетов используются для выбора работы из множества ожидающих выполнения работ. В зависимости от используемой при этом информации они разделяются на локальные и глобальные, статические и динамические, сетевые, временные и ресурсные правила, а также на правила нижней и верхней границы. Среди таких правил можно привести: *GRPW* — наибольший ранг веса позиции, *LFT* — позднее время окончания, *LST* — позднее время начала, *MSLK* — минимальный резерв, *MTS* — наибольшие общие последователи, *RSM* — метод составления расписания ресурса, *SPT* — наименьшее время выполнения, *WCS* — наихудший резерв. Эвристики на основе правил приоритетов комбинируют правило приоритета и схемы генерации расписания для того, чтобы построить конкретный алгоритм. Если эвристический алгоритм генерирует одно расписание, то он называется однопроходным методом, а если более одного расписания, то — многопроходным методом. Среди последних можно привести такие методы, как правило многих приоритетов (*Multi priority rule*), методы прямых — обратных расписаний (*Forward-backward scheduling*), методы выборки (*Sampling methods*). Для решения сложных оптимизационных задач были разработаны метаэвристические подходы: имитированное остывание (*Simulated Annealing*), метод поиска с запретами (*Tabu Search*), генетические алгоритмы, а также некоторые другие эвристики.

По результатам тестирования алгоритмов, проведенного в [6, 7] наиболее эффективными среди большого числа опубликованных работ признаны алгоритмы, описанные в [7—9]. Согласно этим результатам, для проектов небольшой размерности лучшим является приведенный в [8], для бóльших

размерностей с 60 или 120 работами его превосходит генетический алгоритм [9]. Но как для меньших, так и для больших размерностей проектов лучшим считается описанный в [7]. Оценки погрешности некоторых алгоритмов *RCPSP* приведены в [10]. А для задач со складируемыми ресурсами в [11] показана их полиномиальная разрешимость в случае, когда отсутствуют ресурсные ограничения возобновимого типа.

Одними из первых работ, где рассмотрена задача минимизации стоимости проекта, были [12, 13]. В первой из них при линейной непрерывной зависимости стоимости от времени предложен метод сведения двойственной задачи линейного параметрического программирования к задаче о максимальном потоке, а во второй предложен двойственный метод, основанный на структуре двойственной задачи и условиях дополняющей нежесткости задач линейного программирования. Оба эти алгоритма основаны на методе критического пути. Сетевая задача оптимизации по стоимости с непрерывными строго выпуклыми дважды дифференцируемыми функциями стоимости рассмотрена в [14, 15]. В [14] предложен итеративный алгоритм просмотра всех (за исключением единственных начальной и конечной) вершин сети, в котором время, соответствующее данной вершине, находится из условия сохранения потока, а в [15] предлагается осуществить просмотр вершин сети в порядке монотонного изменения их ранга, при этом показано, что данная модификация алгоритма сходится не медленнее геометрической прогрессии со знаменателем меньше единицы.

Задача о минимизации затрат на выполнение всех работ, сформулированная как задача параметрического программирования, приведена в [16]. Алгоритм ее решения состоит из конечного числа итераций, на каждой из которых решается задача о циркуляции минимальной стоимости. В [17] исходная модель была расширена путем введения крайних сроков для некоторых событий — "вех", и разработки модификации для используемого "в беспорядке" (*out-of-kilter*)-алгоритма [13]. В [18] рассмотрена задача календарного планирования "точно в срок", в которой путем выбора продолжительностей работ и времени осуществления событий находится минимальная общая стоимость проекта при сетевых ограничениях предшествования работ и двусторонних ограничениях на продолжительность их выполнения. Для решения данной задачи с кусочно-линейной, выпуклой (возможно, немонотонной) целевой функцией продолжительностей работ, а также времен начала/окончания работ предложена модификация "в беспорядке"-алгоритма, имеющая псевдолинейную сложность.

Приведенные алгоритмы предназначены для задач, имеющих детерминированную структуру сетевой модели, задающей технологические отношения непосредственного предшествования работ: на входе реализована конъюнктивная логика (операция AND), на выходе — детерминированный, т. е. такую структуру, как и в моделях типа PERT.

В имеющихся на рынке информационных системах управления проектами количественная оценка риска расписания (*Risk of Schedule*) и риска стоимости (*Risk of Cost*) проводится на основе моделирования по методу Монте-Карло. А для показателя риск выполнения (*Risk of Performance*) инструментарий его оценки не разработан. Предлагаемый в работе подход направлен на разработку формальных количественных методов не только оценки, но и оптимизации параметров проекта с показателем *Risk of Performance*.

При анализе процессов исследований и разработок кроме структурной неопределенности имеется неопределенность и в возможности достижения поставленных целей. Она зависит как от самой цели, так и от обеспеченности ее реализации. Этот параметр ожидаемой завершенности является случайной величиной. В системе *MS Office Project* 2003 [19], например, при отслеживании реализации календарного плана проекта или его бюджета учитываются такие показатели, как процент выполнения, фактическая длительность задачи, фактические трудозатраты или повременные трудозатраты. С помощью панели инструментов редактируются значения в специальных полях *% Complete* (по длительности), *% Work Complete* (по трудозатратам), а когда важно опираться на реальные данные, то используется поле *Physical % Complete*. Последний показатель также используется для определения приобретенной стоимости или освоенного объема (*Earned Value*) — запланированной по базовому плану стоимости фактически выполненных работ (*Budgeted Cost of Work Performed*).

На практике процессы обеспечения выполнения сходных по назначению и методам решения проблем, проектов, задач и т. д. имеют определенные общие черты и закономерности. Однако получение количественных оценок для характеристик исследований и разработок было связано с определенными трудностями: отсутствием методологических основ определения характеристик и параметров выполнения, недостатком исходной информации об этих процессах для выбора формальных методов нахождения оценок, отсутствием систематического учета и анализа отклонения ожидаемых оценок от фактических.

Для работ с высокой степенью неопределенности для получения характеристик случайных величин их выполнения — ожидаемой завершенности (вероятности достижения цели) — могут быть использованы экспертные, аналитические и вероятностно-статистические методы. В [20, с. 157] говорится, что "оценка ожидаемой завершенности — это субъективная вероятность успеха при реализа-

ции данного режима". Для построения функции распределения такой случайной величины можно использовать эмпирические данные, полученные указанными способами или в результате активного эксперимента [21], в котором в результате моделирования по методу Монте-Карло получена кумулятивная вероятность технического успеха работ в виде монотонно возрастающей с насыщением функции затраченных ресурсов.

Вероятность p_j выполнения работы к некоторому времени характеризует возможность достижения определенных технических показателей результата работы при условии, что предшествующие работы, обеспечивающие ее начало, выполнены. Эта вероятность зависит от объема тех или иных израсходованных ресурсов, которые имеют стоимостное выражение.

Модель и постановка задачи

Пусть задан ациклический ориентированный граф $G(E, \Gamma)$, где E — множество вершин, соответствующих работам проекта (представление *Activity-on-Nod*), а $\Gamma \subseteq E \times E$ — множество его дуг, задающих отношения частичного порядка непосредственного технологического предшествования работ.

Введем индикаторную функцию работ $j = 1, ..., n$ сетевого проекта $G(E, \Gamma)$

$$x(j) = \begin{cases} 1, \text{ если работа } j \text{ выполнена;} \\ 0, \text{ если работа } j \text{ не выполнена,} \end{cases}$$

и индикаторную функцию технологической цепочки (пути) $\mu^j = \{i, ..., j\}$ из начальных вершин i таких, что множество вершин, предшествующих вершине i, $\Gamma_i^{-1} = \emptyset$, в конечную вершину j цепочки, соответствующую выполнению μ^j:

$$x(\mu^j) = \begin{cases} 1, \text{если последовательность} \\ \text{работ } \mu^j \text{ выполнена;} \\ 0, \text{если последовательность} \\ \text{работ } \mu^j \text{ не выполнена.} \end{cases}$$

Для детерминированной сетевой модели с конъюнктивной логикой (AND) по формуле полной вероятности вероятность выполнения i-й работы, когда $j, ..., k$ — ее технологические предшественники, $j \in \Gamma_i^{-1}, ..., k \in \Gamma_i^{-1}$, определяется выражением

$$P_i = p(x(i) = 1 | x(\mu^j) \wedge ... \wedge x(\mu^k) = 1) P(x(\mu^j) \wedge ... \wedge x(\mu^k) = 1) + p(x(i) = 1 | x(\mu^j) \wedge ... \wedge x(\mu^k) = 0) P(x(\mu^j) \wedge ... \wedge x(\mu^k) = 0).$$

Поскольку в данном случае для выполнения работы необходимо выполнение всех предшествующих ей работ, то второе слагаемое равно нулю. В случае, если пути $\mu^j, ..., \mu^k$ независимы, то

$$P(x(\mu^j) \wedge ... \wedge x(\mu^k) = 1) = P(x(\mu^j) = 1) ... P(x(\mu^k) = 1)$$

и тогда

$$P_i = p_i(\mu^j) ... P(\mu^k),$$

где p_i — условная вероятность

$$p_i = p(x(i) = 1 | x(\mu^j) \wedge ... \wedge x(\mu^k) = 1), \quad j \in \Gamma_i^{-1}, ..., k \in \Gamma_i^{-1}. \qquad (1)$$

В телекоммуникационных, транспортных, энергетических и некоторых других технических системах вида двухполюсника с сетевой структурой их надежность (вероятность связности) обеспечивается за счет работоспособности составляющих минимальных путей. Для анализа их надежности применяется метод минимальных путей и минимальных сечений, различные граничные оценки надежности [22]. В сетевой модели детерминированного проекта с конъюнктивной логикой (AND) для его надежного выполнения (безотказной работы) необходимо выполнение всех входящих элементов — работ.

Если технологические цепочки, скажем, μ^j и μ^k, не являются независимыми, значит $\exists\, s \in E$: $s \in \mu^j \wedge \wedge \mu^k$, а в этом случае отказ работы s ($x(s) = 0$) приводит к тому, что ухудшатся показатели как $x(\mu^j)$, так и $x(\mu^k)$ одновременно, а это означает, что данные величины положительно коррелированны:

$$\text{cov}(x(\mu^j), x(\mu^k)) > 0.$$

В этом случае для произвольных $j \in \Gamma_i^{-1}$ и $k \in \Gamma_i^{-1}$, для которых технологические цепочки (μ^j) и (μ^k) не являются независимыми, выполняется условие:

$$P(x(\mu^j) \wedge x(\mu^k) = 1) > P(x(\mu^j) = 1) P(x(\mu^k) = 1).$$

В любом случае как для независимых, так и для положительно коррелированных технологических цепочек при произвольной детерминированной структуре сети $G(E, \Gamma)$ с конъюнктивной логикой выполняется:

$$P_i \geqslant p_i P(\mu^j) ... P(\mu^k), j, ..., k \in \Gamma_i^{-1}.$$

Если раскрывать это выражение дальше для получения вероятности P_{n+1} выполнения всего комплекса работ, то в произведение вероятностей войдут все вершины $j = 1, ..., n$, а отдельные условные вероятности p_s выполнения работ $s \in \mu^j \wedge \mu^k$, вхо-

дящих в различные цепочки μ^k, $k \in E$, могут встречаться более одного раза:

$$P_i \geqslant p_i P(\mu^{i_j}) \ldots P(\mu^{i_k}) \geqslant p_i\, p(x(j) = 1 | x(\mu^{j_s}) \wedge \ldots \wedge$$

$$\wedge x(\mu^{j_t}) = 1) P(\mu^{j_s}) \ldots P(\mu^{j_t}) \ldots p(x(k) = 1 | x(\mu^{k_v}) \wedge \ldots \wedge$$

$$\wedge x(\mu^{k_w}) = 1) P(\mu^{k_v}) \ldots P(\mu^{k_w}),$$

$$j, \ldots, k \in \Gamma_i^{-1};\ s, \ldots, t \in \Gamma_i^{-1},\ v, \ldots, w \in \Gamma_k^{-1}.$$

В таком разложении примут участие все дуги сети и, тем самым, будет реализовано все множество путей, проход по которым будет осуществляться от конечной вершины n до каждой отдельной вершины j и, в конечном итоге, до исходных начальных вершин. В этом разложении некоторые вершины из конечного множества вершин E могут совпадать: $j_s = k_v = r$, т. е. встречаться столько раз d_r, сколько имеется различных путей на графе $G(E, \Gamma)$ из конечной вершины n по ориентированным дугам в обратном направлении в данную вершину $r \in E$. Таким образом, в разложение для детерминированной сети с конъюнктивной логикой (AND) войдут все вершины $r = 1, \ldots, n$, каждая в количестве d_r раз.

Будем называть сеть $G(E, \Gamma)$ правильно занумерованной, если

$$j < k \Rightarrow j < k.$$

Тогда все вершины k такие, что $(j, k) \in \Gamma$ в строках матрицы путей будут находиться справа от j. Рассмотрим строки, в которых на j-м месте стоит 1. Число различных "хвостов" (подмножеств строки) от j до n, которое нетрудно сосчитать, соответствует числу путей из j-й вершины в конечную n, т. е. параметру d_j, $j = 1, \ldots, n$. Поэтому будем считать этот параметр d_j известным для данной правильно занумерованной сети $G(E, \Gamma)$. Тогда для вероятности P_{n+1} выполнения всего комплекса работ получим в случае конъюнктивных входящих дуг гарантированную ее оценку снизу

$$P_{n+1} \geqslant \prod_{j=1}^{n} p_j^{d_j} \geqslant P, \qquad (2)$$

где $P \in (0,1]$ — заданное ограничение; $d_j = 1, 2, 3, \ldots$ — некоторые известные величины.

В [23] для моделирования вероятности успеха научно-исследовательских работ были предложены, в частности, линейная, нелинейная и {0,1}-модель.

Будем моделировать зависимость вероятности (1) успеха работ проекта от затраченного однородного складируемого ресурса u трехпараметрическими степенными функциями вида

$$p_j(u_j) = b_j (u_j - u_j^0)^{\alpha_j}, \qquad (3)$$

где b_j — параметр масштаба; α_j — параметр формы; u_j^0 — параметр сдвига, причем $b_j > 0$, $0 < \alpha_j < 1$, $u_j \geqslant u_j^0 \geqslant 0$, $j = 1, \ldots, n$.

Эти функции заданы на некотором интервале положительной полуоси и достаточно наглядно описывают изучаемую ситуацию.

Модели планирования инновационных проектов с использованием функций вида (3) для вероятности успеха работ исследовались в [24], а S-образные [21] зависимости изучались в [25] и в ряде других работ автора.

Поскольку должно выполняться $p_j \in (0,1]$, то будем полагать, что

$$0 \leqslant u_j^{\min} \leqslant u_j \leqslant u_j^{\max} < \infty,$$

так что

$$b_j (u_j^{\max} - u_j^0)^{\alpha_j} = 1,$$

откуда, опуская индекс, для параметров функции получаем соотношение

$$u^{\max} = u^0 + b^{-1/\alpha}.$$

Таким образом, мы имеем функционально-взвешенный ациклический ориентированный граф $G(E, \Gamma)$, на котором задана нормированная технологическая функция (3), удовлетворяющая соотношению (2).

Заметим, что при планировании проектов по методу освоенного объема (*Earned Value*) [26] используется производный показатель освоенного объема $\alpha_x(t)$ — *QSPI* (*Quantity Schedule Performance Index* — коэффициент освоения планируемой стоимости), имеющий на конец T планового периода (под)проекта значение

$$\alpha_x(t) = \left.\frac{x_e(t)}{x_0(t)}\right|_{t=T} \in [0,1],$$

где $x_e(t)$ — освоенный объем *BQWP* (*Budgeted Quantity of Work Performed* — плановая стоимость выполненных работ), $x_0(t)$ — планируемая динамика объемов работ *BQWS* (*Budgeted Quantity of Work Scheduled* — плановая стоимость запланированных работ), который также может служить содержательной интерпретацией введенной нормированной технологической функции.

Вероятность успеха работы — это вероятность того, что в процессе осуществления проекта при выполнении данной работы не произойдет отказа, а вероятность безотказного выполнения работы это и есть ее надежность выполнения. Поэтому будем говорить о p_i из формулы (1) как о надежности работы и P_{n+1} из оценки (2) — как о надежности проекта. Рассматриваемая задача — это задача пла-

нирования ресурсов для выполнения работ проекта с учетом показателя его надежности, а не управления динамикой его реализации — когда есть фронт работ, выполненные и ожидающие выполнения работы. Эти вопросы рассматриваются в рамках задач по управлению ходом реализации проекта — задач составления расписания работ проекта при ограничениях на ресурсы.

Для зависимости затрат на работу от ее продолжительности выбирается достаточно апробированная [27] линейная зависимость

$$s_j u_j = -r_j t_j + h_j, \tag{4}$$

где r_j, h_j, s_j — параметры линейной функции.

В большинстве практических случаев зависимость затрат на выполнение сетевого комплекса от времени на его осуществление имеет вид выпуклой (вниз) функции. Затраты уменьшаются с увеличением продолжительности, достигая критического значения, после чего наблюдается рост как времени, так и затрат. То есть сокращение сроков осуществления работы требует увеличения затрат. Поскольку имеющаяся информация не обладает достаточной точностью, становится оправданным использование в рабочей области изменения аргумента линейной аппроксимации реальной зависимости. Эта задача представляется достаточно сложной и нуждается в дальнейшем исследовании, поэтому мы принимаем простейшую линейную гипотезу. В ряде случаев она достаточно хорошо описывает реальную ситуацию.

Для представления сетевых ограничений модели на продолжительность проекта будем использовать в качестве характеристической функции системы ограничений матрицу путей сети (ациклического ориентированного графа) $G(E, \Gamma)$: $A = (a_{ij})$, $a_{ij}(j \in \mu_i)$. В сетевой модели проекта $G(E, \Gamma)$, где E — множество вершин, соответствующих работам, а $\Gamma \subseteq E \times E$ — множество дуг, задающих технологические отношения частичного порядка непосредственного предшествования работ, каждый путь из начальных вершин в конечную $\mu_i = (0, ..., j, k, ..., n)$, где вершина 0 принадлежит множеству начальных вершин, однозначно представляется строкой

$$\mu_i = \{a_{i1}, a_{i2}, ..., a_{ij}, a_{ik}, ..., a_{in}\},$$
$$i = 1, ..., m;\ j, k = 1, ..., n;\ (j, k) \in \Gamma,$$

специально построенной матрицы путей (по вершинам) $A = (a_{ij})$, $a_{ij}(j \in \mu_i)$, в которой ее элементы 1 стоят в i-й строке на j-м месте только в том случае, когда j-я вершина принадлежит i-му пути: $a_{ij} = 1 \Leftrightarrow j \in \mu_i$. Всем остальным случаям, когда $j \notin \mu_i$, соответствуют пустые клетки.

Согласно формулам (2)—(4) задача минимизации стоимости проекта при ограничении снизу на надежность его выполнения, сетевых ограничениях на его продолжительность и линейных связях между переменными задачи в конъюнктивной сетевой модели имеет вид

$$\sum_{j=1}^{n} c_j u_j \to \inf_{(u,t) \in D}; \tag{5}$$

$$\prod_{j=1}^{n} (b_j (u_j - u_j^0)^{\alpha_j})^{d_j} \leqslant P; \tag{6}$$

$$\sum_{j \in \mu_i} a_{ij} (j \in \mu_i) t_j \leqslant T,\ i = 1, ..., m, \tag{7}$$

$$s_j u_j = -r_j t_j + h_j,$$

где u_j — объем однородного складируемого ресурса, выделяемого на выполнение j-й работы, $u_j > u_j^0 \geqslant 0$, $j = 1, ..., n$, $t_j > 0$ — продолжительность j-й работы, $c_j > 0$ — цена единицы ресурса при выполнении работы j, параметры b_j, c_j, r_j, $s_j > 0$, $0 < \alpha_j < 1$, P, $T > 0$, d_j — натуральное число. Причем параметры α_j, b_j, c_j, h_j, r_j, s_j, u_j^0, P, T задачи, полученные в результате аппроксимации реальных данных, известны неточно.

Выразив все через переменную u_j, получим выпуклую задачу:

$$\sum_{j=1}^{n} c_j u_j \to \inf_{u \in D}; \tag{8}$$

$$\ln P - \sum_{j=1}^{n} d_j \ln (b_j (u_j - u_j^0)^{\alpha_j}) \leqslant 0; \tag{9}$$

$$\sum_{j \in \mu_i} a_{ij} (j \in \mu_i) \left(-\frac{s_j}{r_j} u_j + \frac{h_j}{r_j} \right) \leqslant T,\ i = 1, ..., m. \tag{10}$$

Метод решения

Представим выпуклую задачу минимизации (8)—(10) в общем виде:

$$J(u) = \inf,$$
$$u \in U = \{u \in U_0 : g_i(u) \leqslant 0,\ i = 1, 2, ..., m\}, \tag{11}$$

где U_0 — заданное выпуклое множество из некоторого гильбертова пространства H, функции $J(u)$, $g_1(u)$, ..., $g_m(u)$ определены и дифференцируемы по Фреше на U_0. Задача (11) неустойчива к возмущениям исходных данных $J(u)$, $g_i(u)$ и является, вообще говоря, некорректной не только по аргументу, но и по функции, и для ее решения нужно применять методы регуляризации [28, 29]. Будем применять метод регуляризации [30], основанный на двухшаговом методе линеаризации из работы [31].

Предположим, что вместо точных значений функции $g_i(u)$ и градиентов $J'(u)$, $g_i'(u)$ известны их приближения

$$g_{ik}(u),\ J_k'(u),\ g_{ik}'(u),\ u \in U_0,$$
$$k = 1, 2, \ldots, \text{соответственно.}$$

Рассмотрим следующий итерационный процесс:

$$u_{k+1} = P_{U_k}[u_k - \gamma_k t_k'(u_k) - \beta_k(u_{k-1} - u_k)],$$
$$k \geqslant 1, \qquad (12)$$

где

$$t_k'(u) = J_k'(u) + \alpha_k(u) \qquad (13)$$

есть приближенное значение градиента функции Тихонова

$T_k(u) = J(u) + 0{,}5\alpha_k\|u\|^2$, $u \in U_0$, P_{U_k} — проекция точки z на множество U_k,

$$U_k = \{z \in U_0 : g_{ik}(u_k) + \langle g_{ik}'(u_k), z - u_k\rangle \leqslant$$
$$\leqslant \theta_k(1 + \|u_k\|^2),\ i = 1, \ldots, m\}, \qquad (14)$$

α_k, β_k, γ_k, θ_k — заданные числа, представляющие собой параметры метода.

Описанный метод (12)—(14) является двухшаговым, так как для вычисления очередного приближения u_{k+1} используются два предыдущих приближения u_{k-1}, u_k; начальные точки u_0, $u_1 \in U_0$ предполагаются известными. Так как точка u_{k+1} из (12) по определению проекции точки является решением задачи минимизации

$$0{,}5\|z - [u_k - \gamma_k t_k'(u_k) - \beta_k(u_{k-1} - u_k)]\|^2 \to \inf,$$
$$z \in U_k \qquad (15)$$

или эквивалентной ей задачи

$$0{,}5\|z - u_k\|^2 + \langle \gamma_k t_k'(u_k) + \beta_k(u_{k-1} - u_k), z - u_k\rangle \to \inf,$$
$$z \in U_k, \qquad (16)$$

то при описании метода (12)—(14) условие (14) можно заменить задачей (15) или (16). Таким образом, на каждом шаге метода (12)—(14) вычисление точки u_{k+1} сводится к задаче минимизации квадратичной функции (15) или (16) на множестве U_k. В данном случае, если $H = E^n$ есть n-мерное евклидово пространство, U_0 — многогранное множество $U_0 = E_+^n$, то задача (15) или (16) превращается в стандартную задачу квадратичного программирования, для решения которой существуют конечные методы [32].

Приведем достаточные условия, при выполнении которых последовательность $\{u_k\}$, полученная методом (12)—(14), сильно сходится в H к нормальному решению задачи (11).

Теорема [30]. Пусть выполнены следующие условия:

1) U_0 — выпуклое замкнутое множество из H, функции $J(u)$, $g_i(u)$, $i = 1, 2, \ldots, m$, дифференцируемы по Фреше и выпуклы на U_0, и их градиенты $J'(u)$, $g_i'(u)$ удовлетворяют условию Липшица

$$\max\{\|J'(u) - J'(v)\|;\ \max_{1 \leqslant i \leqslant m}\|g_i'(u) - g_i'(v)\|\} \leqslant$$
$$\leqslant L\|u - v\|,\ u, v \in U_0,\ L = \text{const} > 0;$$

$$J_* = \inf_{u \in U} J(u) > -\infty,\ U_* = \{u \in U: J(u) = J_*\} \neq \emptyset$$

и существует точка $u_C \in U$, для которой

$$g_i(u_C) < 0;\ i = 1, 2, \ldots, m;$$

2) приближения $g_{ik}(u)$, $J_i'(u)$, $g_{ik}'(u)$, $k = 1, 2, \ldots$ для $g_i(u)$, $J'(u)$, $g_i'(u)$ таковы, что

$$\max_{1 \leqslant i \leqslant m}|g_{ik}(u) - g_i(u)| \leqslant \delta_k(1 + \|u\|^2),\ u \in U_0,$$

$\max\{\|J_i'(u) - J'(u)\|;\ \max_{1 \leqslant i \leqslant m}\|g_{ik}'(u) - g_i'(u)\|\} \leqslant$ $\leqslant \delta_k(1 + \|u\|)$, $u \in U_0$, параметры α_k, β_k, γ_k, θ_k удовлетворяют следующим условиям:

$$\alpha_k \geqslant \alpha_{k+1} > 0,\ \beta_{k+1} \geqslant \beta_k,\ \gamma_k \geqslant \gamma_{k+1} > 0,$$
$$\delta_k \geqslant 0,\ \theta_k \geqslant 0,\ k = 1, 2, \ldots$$

$$\lim_{k \to \infty}(\alpha_k + \gamma_k + \delta_k + \theta_k) = 0,\ \lim_{k \to \infty}\beta_k = \beta_\infty > 0,$$

$$\lim_{k \to \infty}\left(\frac{\delta_k}{\theta_k} + \frac{\theta_k}{\alpha_k} + \frac{|\alpha_k - \alpha_{k+1}|}{\alpha_k^2\gamma_k} + \frac{|\gamma_k - \gamma_{k+1}|}{\alpha_k\gamma_k^2}\right) = 0,$$

$1 - 3\beta_\infty > 0$, $\delta_k\max\{5 + \|u_C\|;\ 3 + 3\|u_C\|\} \leqslant 2\theta_k$, $k \geqslant 1$.

Тогда множество U_k из (14) непусто, выпукло, замкнуто при каждом $k = 1,2, \ldots$, последовательность $\{u_k\}$, определяемая методом (12)—(14), при любом выборе начальных точек u_0, $u_1 \in U_0$ сходится по норме H к нормальному решению задачи (11), т. е. $u_* \in U_*$, $\|u_*\| = \inf_{u \in U_*}\|u\|$, причем сходимость $\{u_k\}$ к u_* равномерная относительно выбора приближений $g_{ik}(u)$, $J_i'(u)$, $g_{ik}'(u)$.

Заключение

Рассмотренная задача о минимизации стоимости проекта в детерминированной конъюнктивной модели проекта относится к классу задач календарного планирования с ограниченными ресурсами. Особенность рассмотренной модели в том, что вводится показатель надежности работы как функ-

ция складируемого ресурса при линейных связях между затратами ресурсов и временем, необходимым для выполнения работы. Такой подход позволяет построить функциональную модель оценки надежности всего проекта как сложной технической системы и сформулировать задачу календарного планирования не только с ресурсными, но и с надежностными ограничениями. Тем самым планирование проектов осуществляется не в пространстве переменных стоимость — время с последующим моделированием проектных рисков по методу Монте-Карло, как в имеющихся сейчас на рынке программных пакетах *Project Management*, а путем решения соответствующих непрерывных оптимизационных задач по управлению проектом в пространстве трех переменных: стоимость — время — надежность (вероятность успеха).

Рассмотренная в статье задача минимизации стоимости проекта, когда функция надежности работы аппроксимируется (вогнутой) степенной функцией с показателем степени меньше единицы, является задачей выпуклого программирования, поэтому для ее решения можно применить регуляризованные методы решения некорректных задач при неточных исходных данных, к которым относится указанный регуляризованный двухшаговый метод линеаризации [30]. Предложенная постановка и метод ее решения имеют практическое значение для разрабатываемого функционального наполнения автоматизированной информационной системы планирования инновационных проектов — системы ДИСУПИР 3.1 [33]. В статье приводится также обзор известных подходов к решению задач календарного планирования с ограниченными ресурсами.

Список литературы

1. **Brucker P., Drexl A., Mqhring R.** et al. Resource-constrained project scheduling: Notation, classification, models, and methods // European J. of Operational Research. 1999. Vol. 112. N 1. P. 3—41.

2. **Herroelen W., De Reyck B., Demeulemeester E.** Resource-constrained project scheduling: a survey of recent developments // Computers & Operations Research. 1998. Vol. 25. N 4. P. 279—302.

3. **Kolisch R., Padman R.** An integrated survey of deterministic project scheduling // Omega. 2001. Vol. 29. P. 249—272.

4. **Blazewicz J., Lenstra J., Rinooy Kan A. H. G.** Scheduling subject to resource constraints: classification and complexity // Discrete Applied Mathematics. 1983. Vol. 5. P. 11—24.

5. **Михалевич В. С., Кукса Л. И.** Методы последовательной оптимизации в дискретных сетевых задачах оптимального распределения ресурсов. М.: Наука, 1983. 208 с.

6. **Hartmann S., Kolish R.** Experimental evaluation of state-of-the-art heuristics for the resource-constraint project scheduling problem // European J. of Operational Research. 2000. Vol. 127. P. 394—407.

7. **Tormos P., Lova A.** A competitive heuristic solution technique for resource-constrained project scheduling // Annals of Operations Research. 2001. Vol. 102. P. 65—81.

8. **Bouleimen K. and Lecocq H.** A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem // Sixth Intern. Workshop on Project Management and Scheduling / Eds. G. Barbarosoglu, S. Karabati, L. Ozdamar and G. Ulusoy. Istanbul: Bogazici University Printing Office, 1998. P. 19—22.

9. **Hartmann S.** A competitive genetic algorithm for resource-constrained project scheduling // Naval Research Logistics. 1998. Vol. 45. P. 733—750.

10. **Лазарев А. А., Гафаров Е. Р.** К решению задачи построения расписания выполнения проекта // А и Т. 2008. № 12. С. 86—104.

11. **Гимади Э. Х., Залюбовский В. В., Севастьянов С. В.** Полиномиальная разрешимость задач календарного планирования со складируемыми ресурсами и директивными сроками // Дискретный анализ и исследование операций. Сер. 2. 2000. Т. 7. № 1. С. 9—34.

12. **Kelley J. E., Walker M. R.** Critical path planning and scheduling: Mathematical basis // Operations Research. 1961. N 9. P. 296—320.

13. **Fulkerson D. R.** A network flow computation for project cost curves // Management Science 1961. N 7. P. 167—178.

14. **Berman E. B.** Resource Allocation in a PERT network under continuous activity time cost function // Management Science. 1964. Vol. 10. N 4.

15. **Рубинштейн М. И.** Сетевая задача оптимизации по стоимости // А и Т. 1979. № 3.

16. **Левнер Е. В.** Параметрическая модель сетевого планирования // Исследования по дискретной оптимизации. М.: Наука, 1976.

17. **Elmaghraby S. E., Pulat P. S.** Optimal project compression with due-date events // Naval Research Logistics Quarterly. 1979. N 2. P. 331—348.

18. **Levner E. V., Nemirovsky A. S.** A network flow algorithm for just-in-time project scheduling // European J. of Operational Research. 1994. Vol. 79. P. 167—175.

19. **Богданов В. В.** Управление проектами в Microsoft Project 2003: Учебный курс (+CD).СПб.: Питер, 2004. 604 с.

20. **Комков Н. И.** Модели управления научными исследованиями и разработками. М.: Наука, 1978. 343 с.

21. **Elkjaer M.** Stochastic Budget Simulation // Int. J. of Project Management. 2000. Vol. 18. № 2. P. 139—147.

22. **Ушаков И. А.** Курс теории надежности систем: учебное пособие для вузов. М.: Дрофа, 2008. 239 с.

23. **Souder W. E.** Analytical effectiveness of mathematical models for R&D project selection // Management Science. 1973. Vol. 19. N 8.

24. **Цвиркун А. Д., Акинфиев В. К., Коновалов Е. Н.** Моделирование и управление инновационными программами в крупномасштабных технических системах: Препринт. М.: ИПУ РАН. 1991.

25. **Топка В.В.** Максиминная динамическая вероятностная модель управления проектом // Изв. РАН.ТиСУ. 1996. № 4. С. 123—125.

26. **Колосова Е. В., Новиков Д. А., Цветков А. В.** Методика освоенного объема в оперативном управлении проектами. М.: ООО "НИЦ "Апостроф", 2000. 156 с.

27. **Филлипс Д., Гарсиа-Диас А.** Методы анализа сетей / Пер. с англ. М.: Мир,1984.

28. **Тихонов А. Н., Арсенин В. Я.** Методы решения некорректных задач. М.: Наука,1979.

29. **Бакушинский А. Б., Гончарский А. В.** Итеративные методы решения некорректных задач. М.: Наука, 1989.

30. **Васильев Ф. П., Недич А., Ячимович М.** Двухшаговый регуляризованный метод линеаризации для решения задач минимизации // ЖВМ и МФ. 1996, Т. 36. № 5. С. 9—19.

31. **Антипин А. С., Недич А., Ячимович М.** Двухшаговый метод линеаризации для задач минимизации // ЖВМ и МФ. 1996. Т. 36, № 4. С. 18—25.

32. **Васильев Ф. П.** Методы оптимизации. М.: Факториал, 2002.

33. **Топка В. В., Воронежцев С. А.** Автоматизированная информационная система ДИСУПИР 3.1 (концепция, описание, руководство пользователя). М.: 11-й формат, 2008. 145 с.

УДК 004.891

А. С. Боровский[1], канд. техн. наук, доц.,
А. Д. Тарасов[2], ст. преподаватель кафедры "Автоматизированные системы обработки информации и управления",
[1] Оренбургский институт путей сообщения — филиал Самарского государственного университета путей сообщения,
[2] Оренбургский государственный университет

Метод обработки экспертной информации на основе нечетких гиперграфов для проектирования систем физической защиты

Предлагается метод обработки экспертных знаний для автоматизированного принятия решения в задаче определения требуемого состава системы физической защиты охраняемого объекта. Основой метода является описание общей модели процесса функционирования системы физической защиты в виде нечетких гиперграфов.

Ключевые слова: *система физической защиты, лингвистические переменные, нечеткие гиперграфы, композиция гиперграфов.*

Введение

Проектирование систем физической защиты (СФЗ) особо важных объектов включает в себя обработку экспертных знаний. Работа с экспертами требует как можно более эффективного получения результата, так как от этого зависит защищенность объекта при проявлении различных видов угроз. Для поддержки принятия решений в процессе проектирования СФЗ необходимо автоматизировать решение задачи определения требуемого состава СФЗ на объекте. В условии нечетких и неполных экспертных знаний, высокая точность автоматизированного принятия решений возможна при наличии большого объема информации, что влечет за собой участие экспертов в большей части определения исходных и промежуточных данных задачи. Упростить работу экспертов возможно при использовании математической модели с минимумом входных данных, в которой процесс принятия решения происходит в несколько этапов с получением промежуточных результатов без дополнительных экспертных знаний.

Используемая для решения задачи общая модель процесса функционирования системы физической защиты рассматривается как объединение трех составляющих элементов:

- источники угроз — виды нарушителей;
- объект — зоны, являющиеся потенциальными целями нарушителей;
- средства защиты — инженерно-технические средства охраны.

Определение степеней взаимодействия этих составляющих элементов позволит оценить требуемый состав комплекса инженерно-технических средств охраны на объекте защиты.

Три составляющих элемента модели можно представить в виде нечетких гиперграфов и рассматривать их взаимодействие как композицию гиперграфов. Нечеткие гиперграфы являются обобщением понятия нечетких графов на случай, когда произвольные ребра могут иметь любое, в пределах данного числа вершин, число нечетко инцидентных им вершин. Следовательно, нечеткий ориентированный гиперграф можно рассматривать либо как произвольный набор нечетких подмножеств, определенных в одном множестве, либо как совокупность нечетких отношений [1]. Например, "источники угроз" можно представить как множество видов нарушителей, объединяемых в подмножества по признаку наличия каких-либо свойств или характеристик.

Из возможных видов графов наиболее подходящим является гиперграф, так как все три составляющие рассматриваемой модели обладают следующим свойством: каждый элемент имеет несколько описываемых характеристик и каждая характеристика может присутствовать у нескольких элементов. Вид отношений "многие ко многим" также присутствует при взаимодействии этих трех составляющих друг с другом. Кроме того, каждая вершина гиперграфа может раскрываться в самостоятельный граф или гиперграф по мере уточнения модели процесса функционирования СФЗ.

1. Представление модели процесса функционирования СФЗ в виде гиперграфов

Дадим определение нечеткого гиперграфа. Пусть $V = \{v_i\}$, $i \in I\{1, 2, 3, ..., n\}$ — конечное множество и $E = \{e\}$, $j \in J = \{1,2, ..., m\}$ — семейство нечетких подмножеств в V. Пара $H = \{V, E\}$ называется нечетким неориентированным гиперграфом, если $e_j \neq \varnothing$,

$j \in J$ и $\bigcup_{j \in J} e_j = V$, где элементы $v_1, v_2, ..., v_n \in V$ являются вершинами гиперграфа, множество E, состоящее из $e_1, e_2, ..., e_m$, — множество нечетких ребер гиперграфа. Если все e_j различны, гиперграф называется простым, иначе, получаем мультигиперграф. Степень принадлежности вершины v_i ребру e_j называется степенью инцидентности вершины v_i и ребра e_j и обозначается $\mu_{ej}(v_i)$ [1].

Опишем используемые в данной модели гиперграфы.

Первый составляющий элемент "источники угроз" представим в виде гиперграфа "Нарушители" $H_X = (X, E, Px)$. Вершинами гиперграфа являются виды нарушителей — x_i, ребрами характеристики нарушителя — e_j, степень инцидентности $\mu_{ej}(x_i)$ — уровень j-й характеристики i-го вида нарушителя.

Второй элемент "зоны объекта защиты" представляем как гиперграф "Объект" $H_Y = (U, Y, Py)$, в котором u_i — характеристики критических элементов — вершины, y_j — критические элементы — ребра, степень инцидентности $\mu_{yj}(u_i)$ — уровень i-й характеристики j-го критического элемента.

Третий элемент "инженерно-технические средства охраны" — это гиперграф "Средства защиты" $H_Z = (Z, Q, Pz)$, в котором z_i — средства защиты — вершины, q_j — характеристики средств защиты — ребра, степень инцидентности $\mu_{qj}(z_i)$ — уровень j-й характеристики i-го средства защиты.

2. Композиции нечетких гиперграфов модели

Рассмотрим операцию композиции нечетких гиперграфов.

Пусть $H_1 = (X_1, E_1, P_1)$ и $H_2 = (X_2, E_2, P_2)$ — нечеткие гиперграфы, такие, что $E_1 \cap X_2 \neq \varnothing$. Тогда нечеткий гиперграф H называется композицией нечетких гиперграфов H_1 и H_2 и обозначается $H = H_1 \circ H_2$, если $X = X_1$, $E = E_2$, а $F(P) = F(P_1) \circ F(P_2)$, где

$$\mu_{F(P)}(x, e) = \bigcup_{y \in E_1 \cap X_2} (\mu_{F(P_1)}(x, y) \,\&\, \mu_{F(P_2)}(y, e)),$$
$$x \in X_1,\ e \in E_2.$$

В случае, когда $E_1 \cap X_2 = \varnothing$, в результате композиции получаем гиперграф, состоящий из всех изолированных вершин и ребер.

Данная композиция гиперграфов является максиминной. Также может быть использована минимальная композиция, миниминная, максимальная, а также композиция на основе других базисов операций над нечеткими переменными [1].

Опишем присутствующие в модели композиции гиперграфов.

Композиция $H_X = (X, E, Px)$ и $H_Y = (U, Y, Py)$ будет получена при условии $E \cap U \neq \varnothing$, т. е. необходимо наличие соответствия характеристик нарушителя и характеристик критических элементов.

Нечеткий гиперграф композиции $H_{XY} = (X, Y, Pxy)$, в котором x_i — виды нарушителей — вершины; y_j — критические элементы — ребра; степень инцидентности $\mu_{yj}(x_i)$ — уровень опасности нарушителя i-го вида для j-го критического элемента.

Перед определением следующей композиции проведем операцию получения двойственного гиперграфа для $H_X \to H_X^*$.

$H_X^* = (E, X, P^*x)$ двойственный гиперграф со следующими свойствами: e_i — характеристики нарушителя — вершины; x_j — виды нарушителей — ребра; степень инцидентности $\mu_{xj}(e_i)$ — уровень i-й характеристики j-го вида нарушителя.

Далее рассматриваем композицию гиперграфов $H_Z = (Z, Q, Pz)$ и $H_X^* = (E, X, P^*x)$, которая будет получена при условии $Q \cap E \neq \varnothing$, следовательно, необходимо наличие соответствия характеристик средств защиты и характеристик нарушителя.

Нечеткий гиперграф композиции $H_{ZX} = (Z, X, Pzx)$, в котором z_i — средства защиты — вершины; x_j — виды нарушителей — ребра; степень инцидентности $\mu_{xj}(z_i)$ — уровень эффективности i-го средства защиты при противодействии нарушителю j-го вида.

Определение следующей композиции $H_Z = (Z, Q, Pz)$ и $H_Y = (U, Y, Py)$ является принятием решения в данной модели — элементы композиции покажут требуемый состав комплекса инженерно-технических средств охраны. Расчет композиции должен быть проведен путем комбинации двух предыдущих: $H_{XY} = H_X \circ H_Y$ и $H_{ZX} = H_Z \circ H_X^*$.

Нечеткий гиперграф композиции $H_{ZY} = (Z, Y, Pzy)$, в котором z_i — средства защиты — вершины; y_j — критические элементы — ребра; степень инцидентности $\mu_{yj}(z_i)$ — уровень необходимости применения i-го средства для защиты j-го критического элемента.

3. Условный пример расчета модели

3.1 Описание гиперграфов модели

Рассмотрим пример составления модели процесса функционирования СФЗ на основе нечетких гиперграфов. Зададим вершины и ребра гиперграфов модели.

Вершины гиперграфа $H_X = (X, E, Px)$:

x_1 — внешний нарушитель 1-го типа: террористическая группа численностью 5—12 человек; x_2 — внешний нарушитель 2-го типа: групповой нарушитель малочисленная группа лиц (2—4 человека); x_3 — внешний нарушитель 3-го типа: одиночный подготовленный нарушитель, не имеющий санкционированного доступа на территорию объекта, имеющий целью совершение террористического

акта (ТА); x_4 — внешний нарушитель 4-го типа: одиночный нарушитель, не имеющий санкционированного доступа на территорию объекта, имеющий целью хищение материальных ценностей; x_5 — внутренний нарушитель 1-го типа: работник объекта (специалист), имеющий санкционированный доступ на территорию объекта; x_6 — внутренний нарушитель 2-го типа: работник охраны объекта.

Ребра гиперграфа $H_X = (X, E, Px)$:

e_1 — цель нарушителя; e_{2a} — осведомленность нарушителя о доступности критического элемента; e_{2b} — осведомленность о уязвимости критического элемента; e_3 — наличие оружия и техническая оснащенность; e_{4a} — возможности нарушителя к обходу средств обнаружения, т. е. к затруднению его обнаружения; e_{4b} — возможности нарушителя по преодолению рубежей защиты; e_{4c} — возможности нарушителя по прохождению системы контроля доступа (СКД); e_{4d} — возможности нарушителя по прохождению системы телевизионного наблюдения (СТН); e_5 — готовность к вступлению в вооруженный конфликт; e_6 — возможность разделения на тактические группы; e_7 — возможность пожертвовать собой; e_8 — тактика действия; e_9 — возможность к вступлению в сговор; e_{10} — модель нарушителя; e_{11} — значимость базовых угроз, т. е. какую степень последствий они могут принести.

Ребра e_2 — уровень осведомленности и e_4 — уровень подготовки разделены на несколько составляющих как пример уточнения исходных данных модели процесса функционирования СФЗ.

Определим возможные значения характеристик нарушителя, т. е. нечеткие значения инцидентности:

e_1 — цель нарушителя: e_{1a} — террористический акт, e_{1b} — хищение, e_{1c} — повреждение оборудования, e_{1d} — захват заложников;

e_2 — уровень осведомленности: высокий, средний, низкий;

e_3 — наличие оружия и техническая оснащенность: высокое, среднее, низкое;

e_4 — уровень подготовки: высокий, средний, низкий;

e_5 — готовность к вступлению в вооруженный конфликт: высокая, средняя, низкая;

e_6 — возможность разделения на тактические группы: высокая, средняя, низкая;

e_7 — возможность пожертвовать собой: высокая, средняя, низкая;

e_8 — тактика действия: e_{1a} — насильственная, e_{1b} — скрытая, e_{1c} — обманная, e_{1d} — легальная, e_{1e} — открытая;

e_9 — возможность к вступлению в сговор: высокая, средняя, низкая;

e_{10} — модель нарушителя: e_{1a} — террористическая группа: e_{1b} — групповой нарушитель: e_{1c} — одиночный нарушитель;

e_{11} — значимость базовых угроз: высокая, средняя, низкая.

Характеристики представляют собой переменные, значения которых могут быть либо измерены, либо выражены лингвистически в форме словесной экспертной оценки. Например: "высокая" — значение из числового диапазона 0,8—0,9; "средняя" — 0,4—0,5; "низкая" — 0,1—0,2.

Ребра e_1 — цель нарушителя, e_8 — тактика действия, e_{10} — модель нарушителя будет представлена в виде четких гиперграфов. Степени инцидентности вершин $x_1 ... x_6$ ребрам e_{1a}, e_{1b}, e_{1c}, e_{1d}, e_{1a}, e_{1b}, e_{1c}, e_{1d}, e_{1e}, e_{1a}, e_{1b}, e_{1c} будут равны либо 0, либо 1. Например, целями внешнего нарушителя 1-го типа могут быть террористический акт, повреждение оборудования, захват заложников. Соответственно x_1 четко принадлежит ребрам e_{1a}, e_{1c}, e_{1d}, т. е. $\mu_{e1a}(x_1) = 1 \mu_{e1b}(x_1) = 0$ $\mu_{e1c}(x_1) = 1 \mu_{e1d}(x_1) = 1$.

В данном примере рассмотрим модельный объект с пятью критическими элементами — предприятие, на котором синтезируются химические вещества.

Вершины гиперграфа $H_Y = (U, Y, Py)$:

u_1 — степень уязвимости конструкции и систем, обеспечивающих безопасность критического элемента (сложность совершения ТА, время на совершение ТА по достижении критического элемента); u_2 — приемлемость риска при совершении ТА из-за ее последствий для самих террористов; u_3 — степень экономических, политических, военных, экологических, психологических и других последствий ТА; u_4 — доступность критического элемента для совершения по отношению к нему ТА, связанная с условиями функционирования объекта, доступа персонала и других лиц, расположения и других особенностей объекта.

Ребра гиперграфа $H_Y = (U, Y, Py)$:

y_1 — материальный склад; y_2 — котельная; y_3 — административное здание; y_4 — склад готовой продукции; y_5 — склад сырья.

Вершины гиперграфа $H_Z = (Z, Q, Pz)$: z_1 — охранная сигнализация, z_2 — тревожно-вызывная сигнализация, z_3 — СКУД, z_4 — система телевизионного наблюдения, z_5 — система оперативной связи, z_6 — система электропитания, z_7 — система освещения, z_8 — система защиты информации.

Ребра гиперграфа $H_Z = (Z, Q, Pz)$: q_1 — возможности по обнаружению нарушителя; q_2 — возможности по затруднению перемещения нарушителей, q_3 — возможности по ограничению доступа в охраняемые зоны; q_4 — возможности установления оперативной связи с момента срабатывания датчика с постом охраны; q_5 — возможности системы по распознаванию объектов.

Для составления композиций гиперграфов необходимо наличие соответствий характеристик нарушителя и критических элементов, а также характеристик нарушителя и средств защиты. Покажем данные соответствия:

e_{2a} — осведомленность нарушителя о доступности критического элемента соответствует u_4 — доступности критического элемента для совершения по отношению к нему ТА;

e_{2b} — осведомленность о уязвимости критического элемента соответствует u_1 — степени уязвимости конструкции и систем, обеспечивающих безопасность критического элемента;

e_{4a} — возможности нарушителя к обходу средств обнаружения, т. е. к затруднению его обнаружения соответствует q_1 — возможности по обнаружению нарушителя;

e_{4b} — возможности нарушителя по преодолению рубежей защиты соответствует q_2 — возможности по затруднению перемещения нарушителей;

e_{4c} — возможности нарушителя по прохождению СКД соответствует q_3 — возможности по ограничению доступа в охраняемые зоны;

e_{4d} — возможности нарушителя по прохождению СТН соответствует q_5 — возможности системы по распознаванию объектов;

e_7 — возможность пожертвовать собой соответствует u_2 — приемлемости риска при совершении ТА из-за ее последствий для самих террористов;

e_{11} — значимость базовых угроз, т. е. какую степень последствий они могут принести, соответствует u_3 — степени экономических, политических, военных, экологических, психологических и других последствий ТА.

3.2 Кенигово представление гиперграфов модели

Взаимно однозначным представлением гиперграфа H является двудольный нечеткий граф $K(H) = (X \cup E, V)$, называемый нечетким кениговым представлением

$$V = \{\langle \mu_v(x, e)/(x, e)\rangle / \mu_v(x, e) = \mu_e(x) \neq 0, x \in X, e \in E\}.$$

Иначе говоря, множество вершин исходного гиперграфа принимается в качестве вершин одной доли графа $K(H)$, множество ребер исходного гиперграфа — в качестве вершин другой доли, две вершины e и x графа $K(H)$ соединяются нечетким ребром со степенью смежности $\mu_v(x, e)$ этих вершин, равной степени инцидентности вершины x и ребра e (не равной 0) исходного гиперграфа [2].

Нечеткий граф $K(H)$ позволяет естественным образом распространить многие понятия и методы теории четких и нечетких графов на нечеткие гиперграфы. Особенно удобно использовать его при исследовании связности систем, представимых нечеткими гиперграфами. Кроме того, граф $K(H)$ является отображением нечеткого соответствия между множествами вершин и ребер. Следовательно, с его помощью можно исследовать объекты, представимые гиперграфами, используя свойство нечетких соответствий [2].

Также для анализа свойств гиперграфов модели можно использовать вершинные и реберные графы. Нечеткий вершинный граф представляет нечеткое бинарное отношение смежности вершин гиперграфа. Аналогично определяется нечеткий реберный граф, представляющий нечеткое бинарное отношение смежности ребер гиперграфа [1]. Например, вершинный граф гиперграфа "Нарушители" покажет, насколько совпадают характеристики у разных видов нарушителей, присутствующих в модели.

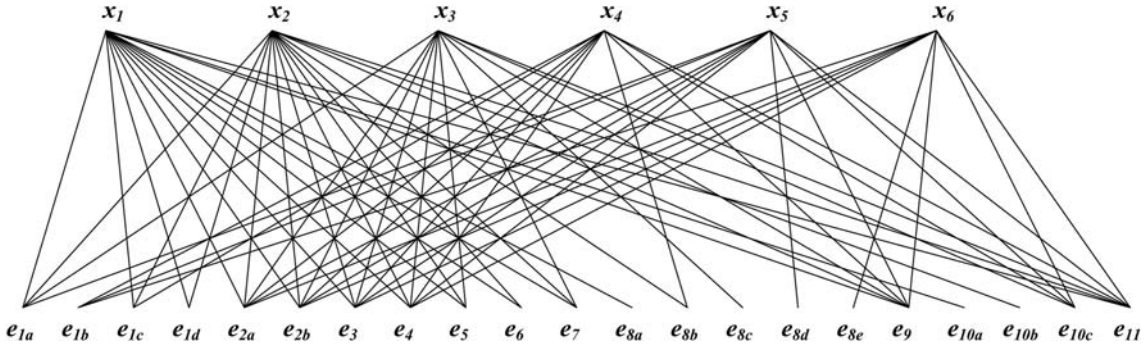


Рис. 1. Кенигово представление гиперграфа "Нарушители"

Таблица 1

Матрица инциденций гиперграфа "Нарушители"

	e_{1a}	e_{1b}	e_{1c}	e_{1d}	e_{2a}	e_{2b}	e_3	e_4	e_5	e_6	e_7	e_{8a}	e_{8b}	e_{8c}	e_{8d}	e_{8e}	e_9	e_{10a}	e_{10b}	e_{10c}	e_{11}
x_1	1	0	1	1	0,5	0,4	0,9	0,9	0,9	0,9	0,9	1	0	0	0	0	0,9	1	0	0	0,9
x_2	1	0	1	1	0,5	0,4	0,9	0,9	0,2	0,4	0,2	0	1	0	0	0	0,9	0	1	0	0,8
x_3	1	0	0	0	0,2	0,1	0,9	0,9	0,9	0	0,9	0	0	1	0	0	0,2	0	0	1	0,5
x_4	0	1	1	0	0,2	0,1	0,2	0,2	0	0	0	0	1	0	0	0	0,4	0	0	1	0,2
x_5	1	1	0	0	0,9	0,9	0,2	0,5	0	0	0	0	0	0	1	0	0,9	0	0	1	0,4
x_6	0	1	0	0	0,9	0,9	0,5	0,2	0	0	0	0	0	0	0	1	0,9	0	0	1	0,5

Построим двудольный нечеткий граф кенигова представления для гиперграфа "Нарушители" (рис. 1). Основой для построения будет однозначно представляющая гиперграф матрица инциденций (табл. 1).

3.3 Композиции гиперграфов модели

Приведем пример построения композиции гиперграфов "Нарушители" и "Объект" $H_{XY} = H_X \circ H_Y$. Используем соответствие характеристик нарушителя в гиперграфе $H_X = (X, E, Px)$ и характеристик критических элементов в гиперграфе $H_Y = (U, Y, Py)$:

$$\mu_{F(Pxy)}(x, y) =$$

$$= \bigcup_{h \in E \cap U} (\mu_{F(X)}(x, h) \,\&\, \mu_{F(Y)}(h, y)),\ x \in X,\ y \in Y,$$

где h — это соответствующие друг другу характеристики e_{2a}—u_4, e_{2b}—u_1, e_7—u_2, e_{11}—u_3.

Процесс построения композиции через кениговы представления отображен на рис. 2. Матрица инциденций гиперграфа композиции показана в табл. 2.

Аналогично построим композицию гиперграфов "Средства защиты" и "Нарушители" $H_{ZX} = H_Z \circ H^*_X$ (табл. 3).

Третью композицию определяем по двум предыдущим. Для построения композиции $H_{ZY} = H_Z \circ H_Y$ используем гиперграфы $H_{XY} = H_X \circ H_Y$ и $H_{ZX} = H_Z \circ H^*_X$ (табл. 4).

Элементы композиции $H_{ZY} = H_Z \circ H_Y$ определяют требуемый состав комплекса инженерно-технических средств охраны. Коэффициенты матрицы инциденций показывают степень необходимости обеспечения каждого критического элемента объекта различными средствами защиты.

Например, в приведенном условном примере относительно малые коэффициенты для критического элемента y_3 — "административное здание" предполагают низкие требования к его защите. Элемент y_2 — "котельная" необходимо защищать в большей степени, причем в заданных условиях лучшими средствами защиты будут z_3 — СКУД, z_4 — система телевизионного наблюдения и z_7 — система освещения.

Расчеты по данной модели проводились с помощью программы, написанной автором на языке *Visual Basic for Applications*. Результаты трех композиций гиперграфов выводятся в виде матриц инцидентности.

Таблица 2

Матрица инциденций гиперграфа композиции $H_{XY} = H_X \circ H_Y$

	y_1	y_2	y_3	y_4	y_5
x_1	0,5	0,8	0,4	0,8	0,5
x_2	0,5	0,8	0,4	0,8	0,3
x_3	0,5	0,7	0,4	0,4	0,5
x_4	0,2	0,2	0,2	0,2	0,2
x_5	0,6	0,9	0,5	0,9	0,3
x_6	0,6	0,9	0,5	0,9	0,3

Таблица 3

Матрица инциденций гиперграфа композиции $H_{ZX} = H_Z \circ H^*_X$

	z_1	z_2	z_3	z_4	z_5	z_6	z_7	z_8
x_1	0,5	0,5	0,9	0,9	0,3	0,4	0,5	0,5
x_2	0,5	0,5	0,4	0,8	0,3	0,4	0,5	0,5
x_3	0,5	0,5	0,9	0,5	0,3	0,4	0,2	0,5
x_4	0,2	0,2	0,1	0,2	0,2	0,2	0,2	0,2
x_5	0,5	0,5	0,9	0,9	0,3	0,4	0,9	0,5
x_6	0,5	0,5	0,9	0,9	0,3	0,4	0,9	0,5

Таблица 4

Матрица инциденций гиперграфа композиции $H_{ZY} = H_Z \circ H_Y$

	z_1	z_2	z_3	z_4	z_5	z_6	z_7	z_8
y_1	0,5	0,5	0,6	0,6	0,3	0,4	0,6	0,5
y_2	0,5	0,5	0,9	0,9	0,3	0,4	0,9	0,5
y_3	0,5	0,5	0,5	0,5	0,3	0,4	0,5	0,5
y_4	0,5	0,5	0,9	0,9	0,3	0,4	0,9	0,5
y_5	0,5	0,5	0,5	0,5	0,3	0,4	0,5	0,5

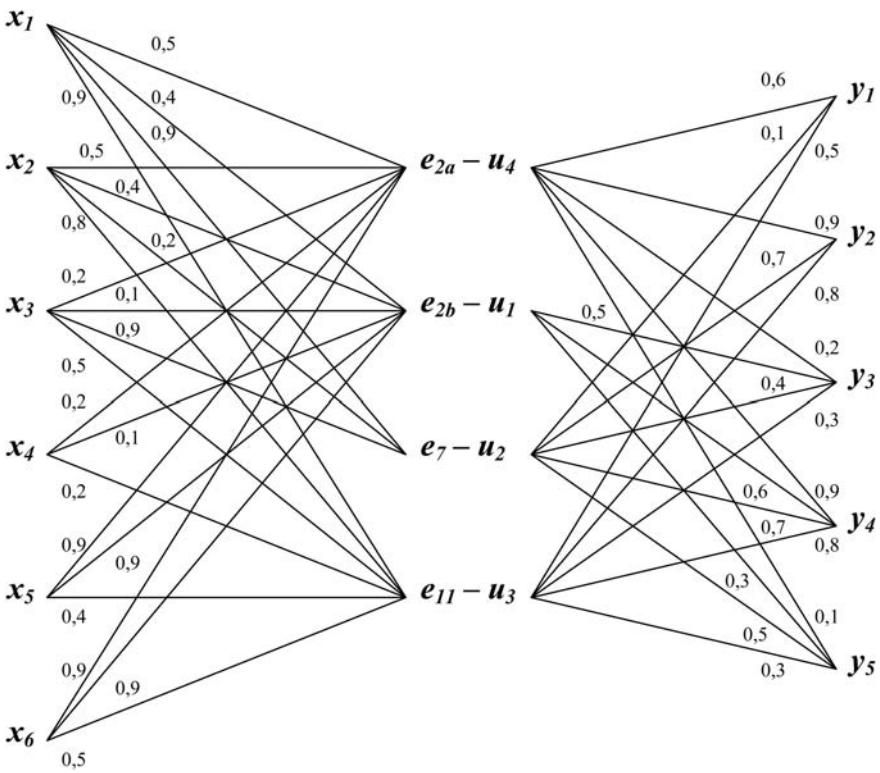


Рис. 2. Схема построения композиции $H_{XY} = H_X \circ H_Y$

Заключение

Представленный способ описания модели процесса функционирования СФЗ в виде гиперграфов и их композиции обладает следующими достоинствами:

1. Гиперграф дает возможность легко отобразить отношения вида "многие ко многим", присутствующие в предметной области, а также каждая вершина гиперграфа может раскрываться в самостоятельный граф или гиперграф по мере уточнения и усложнения модели.

2. Возможности нечетких гиперграфов позволяют отразить нечеткость и неопределенность, присутствующие в предметной области и использовать в качестве исходных данных экспертную информацию в нечеткой форме.

3. Исходные данные для гиперграфовой модели, включают в себя только определение свойств трех составляющих, представленных гиперграфами, т. е. элементов и их характеристик в виде вершин и ребер.

Так, например, в статье [3] модель представлялась в виде множеств, и эксперты должны были задавать соответствия множеств. Чтобы определить числовые значения, характеризующие соответствие множеств "средства защиты" и "источники угроз", экспертам нужно оценить уровни влияния всех рассматриваемых средств защиты на все возможные виды угроз. Для этого требуется ответить на вопросы вида: "какую степень противодействия хищению материальных ценностей имеет система телевизионного наблюдения". Такого рода информация является трудноопределяемой и часто, без возможности экспериментального подтверждения, оценивается экспертами интуитивно.

В гиперграфовой модели определение уровней влияния друг на друга элементов из различных гиперграфов проводится без участия экспертов.

К недостаткам гиперграфовой модели можно отнести следующее:

1. Для составления композиций гиперграфов необходимо наличие соответствий характеристик нарушителя и критических элементов, а также характеристик нарушителя и средств защиты. При малом количестве пар соответствий, а также вследствие возможной низкой связи между характеристиками в паре, полученное итоговое решение будет приближенным или не охватывающим все элементы модели.

2. Каждый элемент гиперграфа должен быть оценен по всем характеристикам, включаемым в данный гиперграф. При большом числе характеристик возможна ситуация, когда уровень какой-либо характеристики для конкретного элемента не может быть оценен с достаточной уверенностью. Например, сложно оценить уровень возможности по обнаружению нарушителя для системы освещения. Следовательно, в модели будет присутствовать трудноопределяемая экспертная информация, что снизит точность получения результата.

Таким образом, применение гиперграфовой модели в процессе проектирования СФЗ может повысить эффективность принятия решения при определении требуемой структуры СФЗ объекта.

Список литературы

1. **Малышев Н. Г., Берштейн Л. С., Боженюк А. В.** Нечеткие модели для экспертных систем в САПР. М.: Энергоатомиздат, 1991. 136 с.

2. **Мелихов А. Н., Берштейн Л. С., Коровин С. Я.** Ситуационные советующие системы с нечеткой логикой. М.: Наука. Гл. ред. физ.-мат. лит., 1990. 272 с.

3. **Боровский А. С., Тарасов А. Д.** Интегрированный подход к разработке общей модели функционирования систем физической защиты объектов: Труды ИСА РАН. 2011. Том 61. № 1.

УДК 004.9

Г. Ч. Набибекова, зав. отделом,
Институт информационных технологий
НАН Азербайджана, г. Баку,
e-mail: gulnara@iit.ab.az, gulnarara58@mail.ru

Применение OLAP-технологий в системах поддержки принятия решений в сфере внешней политики

Рассмотрена возможность применения OLAP-технологий в информационно-аналитических системах поддержки принятия решений в сфере внешней политики. С этой целью рассмотрены проблемы, возникающие при принятии внешнеполитических решений, сущность многомерной модели данных, функции и структура хранилища данных, многомерный OLAP-куб.

***Ключевые слова**: информационно-аналитическая система поддержки принятия решений, хранилища данных, технология OLAP, многомерная модель данных, OLAP-куб*

Введение

В процессе принятия управленческих решений аналитик выдвигает определенные гипотезы, которые прежде чем превратиться в решения, должны быть проверены. Информация, на основании которой проверяется та или иная гипотеза, представляется как зависимость между некоторыми параметрами, которые характеризуют исследуемую предметную область. Например, объем продаж зависит от таких параметров, как регион, время, категория товара. Другим примером могут служить лог-файлы Интернет-сервера. В них также имеется множество параметров, таких как время, страница, хост, ссылающийся сервер, поисковая система, посетитель и т. д. От этих параметров зависят такие величины, как число посещений, число хитов, время, проведенное на странице и другая информация, имеющаяся в логе. В этих случаях, как и во многих других, средства анализа, основанные на данных, представленных в реляционных базах данных, не могут удовлетворить нужным требованиям. Следовательно, становится очевидной необходимость рассматривать и анализировать данные с точки зрения множественности измерений.

Решением данной проблемы является использование технологии комплексной оперативной аналитической обработки многомерных данных OLAP (*On-line Analytical Processing*), основанной на многомерной логической модели данных.

Рассматривая вопрос о возможности применение OLAP, можно сказать, что OLAP применим везде, где есть задача анализа многофакторных данных. То есть при наличии некоторой таблицы с данными, в которой есть хотя бы одна описательная колонка и одна колонка с цифрами, OLAP-инструмент является эффективным средством анализа и генерации отчетов [1]. Следует отметить, что некоторые сферы, такие как маркетинг, финансовая отчетность, бухгалтерские счета и некоторые другие, являются классическими для применения OLAP-технологий. Но есть и такие сферы, в которых не сразу видны те возможности, которые дает нам их применение. В данной статье рассматриваются возможности применения OLAP в системах поддержки принятия решений в сфере внешней политики.

Проблемы принятия внешнеполитических решений

Современные международные отношения характеризуется высокой подвижностью и стремительными переменами. В таких условиях для работников внешнеполитической сферы основной проблемой является умение мгновенно реагировать на происходящие изменения, быстро адаптироваться к новым требованиям, осваивать постоянно возникающие все новые и новые "правила игры", используя экономические, политические, военные, технологические, информационные и интеллектуальные возможности.

Поскольку внешнеполитические решения могут приобретать стратегический характер и вести к долговременным последствиям, здесь совершенно не допустимы решения субъективные, продуманные лишь на полшага вперед и опирающиеся исключительно на конъюнктурные соображения. Многие из этих решений в ряде случаев трудноисправимы и могут нанести ущерб долгосрочным национальным интересам страны и непосредственно интересам граждан.

Для решения существующих проблем необходимо иметь такой высокоэффективный механизм подготовки, принятия и исполнения решений по стратегическим вопросам международной деятельности, который опирался бы на глубокие аналитические разработки, экспертизу, прогнозирование [2]. В результате действия этого механизма может быть получена научно обоснованная достоверная и качественная информация.

Рассматривая задачи, стоящие перед различными внешнеполитическими организациями, можно сделать вывод, что OLAP может сыграть ключевую роль при решении практически любой такой задачи. Использование OLAP может дать достаточно неожиданные результаты при анализе отношений между странами, показать их динамику, опреде-

лить тенденции, выявить наиболее надежных партнеров в любой сфере. Осуществляя обработку огромного числа данных, инструменты OLAP могут также значительно повысить эффективность управления как финансовыми, так и кадровыми ресурсами.

Сущность многомерной модели данных

Прежде чем приступить к исследованию вопроса практического применения OLAP для решения задач внешней политики, рассмотрим в чем заключается сущность многомерной модели данных.

Технология OLAP представляет собой интерактивную (диалоговую) аналитическую обработку, которая в отличие от плоской реляционной модели данных дает возможность на основе многомерной (гиперкубической) модели данных моделировать реальные структуры и связи, которые являются исключительно важными для аналитических систем. С помощью OLAP-технологий создаются мультипараметрические модели, цель которых более адекватно представлять реальные процессы. Технология OLAP позволяет быстро изменять взгляды на данные в зависимости от выбранных параметров и обеспечить лицу, принимающему решение, полную картину анализируемых ситуаций [3].

Многомерная логическая модель данных, которая является основой OLAP-технологий, позволяет наглядно представить процесс работы с данными.

Главной отличительной особенностью и важным преимуществом многомерного представления данных по сравнению с традиционными информационными методиками является возможность совместного анализа больших групп параметров и их взаимной связи, что, естественно, очень важно при изучении сложных явлений, включающих множество предметных областей.

Предлагаемая многомерная модель данных будет строиться на основе отчетов о зарубежных поездках сотрудников госучреждения. Эти отчеты, содержащие важную оперативную, историческую и справочную информацию, хранятся в бумажном или электронном виде в архивах различных отделов госучреждения — отделе кадров, бухгалтерии, научно-организационном отделе и т. д., т. е. хранение этой информации в подавляющем большинстве случаев носит неорганизованный, бессистемный характер. Поэтому она практически недоступна для использования в тех или иных целях в случае необходимости.

Функции и структура хранилища данных

OLAP является ключевым компонентом хранилища данных (ХД). Средства OLAP применяются для извлечения достоверной и качественной информации из данных, находящихся в ХД.

Отметим ключевые тенденции в сфере внешней политики на примере Азербайджана — они заключаются в следующем:

- *растет число стран*, с которыми Азербайджан налаживает связи, т. е. ХД пополняется по вертикали;
- *расширяются сферы сотрудничества* Азербайджана с другими странами, т. е. ХД пополняется по горизонтали.

Эти данные попадают в ХД из оперативных систем (OLTP-систем) и являются внутренними источниками данных (отчеты о зарубежных командировках).

Кроме того, ХД может пополняться за счет внешних источников [4], информация которых может являться одним из основных факторов, влияющих на принятие решений в сфере внешней политики, связанных с взаимодействием с той или иной страной. К ним относятся:

- политическая ситуация в стране;
- экономическая ситуация в стране;
- ситуация в стране, связанная с природными аномалиями.

Как известно, основными составляющими структуры хранилищ данных являются таблица фактов (*Fact Table*) и таблицы измерений (*Dimension Tables*). На основании данных отчетов о зарубежных поездках создадим таблицу фактов и таблицы измерений. Таблица фактов, как правило, содержит первичный ключ, объединяющий первичные ключи таблиц измерений [5].

На рис. 1 представлена примерная структура ХД, которая состоит из таблицы фактов (Foreignpol_Fact) и трех таблиц измерений (Person_Dim, Country_Dim, Event_Dim) [6].

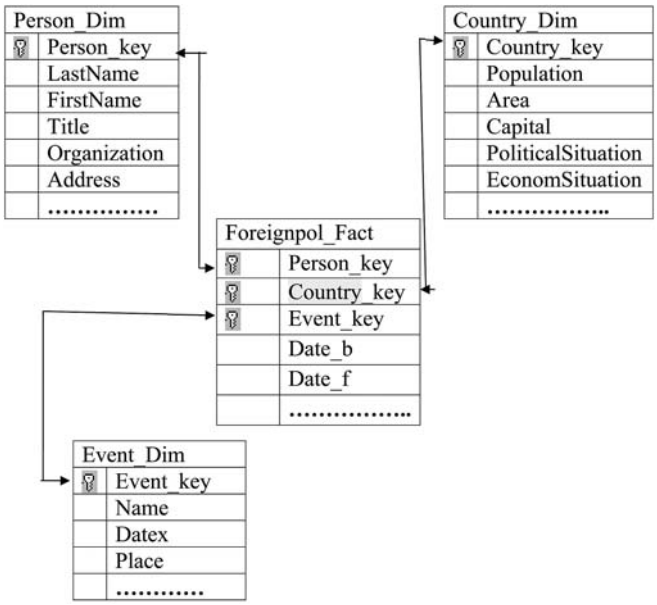


Рис. 1. Структура хранилища данных

Основные задачи, которые можно решить с помощью ХД и OLAP, следующие:

- *управление* — роль ХД заключается в подготовке отчетов и результатов анализа для информационной поддержки принятия решений;
- *управление трудовыми ресурсами* — технология ХД может существенно облегчить осуществление HR-стратегии (*Human Resources*-стратегии), можно получить обобщенную информацию о людских ресурсах и добиться повышения эффективности их деятельности. Вот некоторые направления использования этих технологий в этом направлении:
  - анализ трудовых ресурсов;
  - размещение трудовых ресурсов;
  - обучение и планирование продвижения сотрудников;
- *управление финансами* — роль финансового отчета за последнее время существенно возросла, и все чаще эта информация используется для принятия стратегических решений.

Построение многомерного OLAP-куба для реализуемой задачи

Осями многомерной системы координат служат основные атрибуты анализируемого процесса, назовем их *измерениями*. Внутри куба находятся данные, количественно характеризующие процесс, — так называемые "меры", будем называть их *показателями*.

На рис. 2 представлен OLAP-куб, построенный на основании созданного выше ХД.

Отметим, что OLAP-куб называется "кубом" чисто символически, поскольку в зависимости от запроса можно получить и двухмерный, и трехмерный, и четырехмерный, и т. д. набор данных в виде срезов. И ячейки "куба" в зависимости от запроса будут содержать агрегатные данные, соответствующие находящимся на осях "куба" значениям параметров запроса.

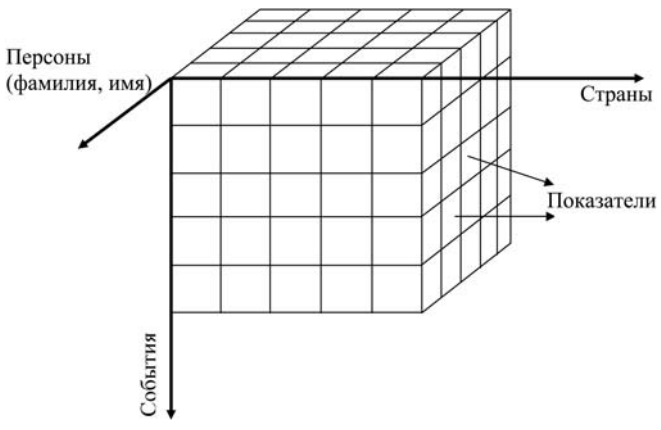


Рис. 2. OLAP-куб

В качестве примера можно привести следующие очевидные запросы для получения агрегатных данных:

1. Сколько человек посетили Турцию в 2011 г.?
2. Сколько человек посетили Турцию в 2011 г. в связи с мероприятиями в медицинской сфере?
3. Какова численность делегации, посетившей Турцию в 2011 г. в связи с Международной конференцией в аграрной сфере?
4. Каковы финансовые затраты на поездки, связанные с ИКТ?

и т. д.

Агрегатной функцией может быть SUM, AVR, MAX, MIN и т. д., т. е. все зависит от поставленной задачи и соответствующего ей запроса.

Пользователь, анализирующий информацию, может "нарезать" куб по разным направлениям, получать сводные (например, по годам, странам) или, наоборот, детальные (по отдельным сферам) данные и осуществлять другие операции, необходимые для анализа. Поскольку аналитик всегда оперирует некими суммарными данными, в базах данных OLAP практически всегда хранятся наряду с детальными данными и так называемые агрегаты, т. е. заранее вычисленные показатели. Хранение заранее вычисленных агрегатов — основной способ повышения скорости выполнения OLAP-запросов.

Заключение

Подводя итоги всему сказанному в данной статье, отметим основные преимущества использования OLAP-систем:

- интуитивно понятный пользовательский интерфейс для просмотра сложных данных;
- высочайшая производительность на любых объемах данных;
- простота создания любых отчетных форм для используемой системы;
- обеспечение точности данных [7].

Несмотря на преимущества, которые дает применение OLAP-технологий, есть причины, которые мешают их широкому распространению.

Во-первых, у специалистов отсутствует понимание существующих альтернативных подходов к решению проблем, а в IT-подразделениях не достает осознания сложностей, с которыми сталкиваются специалисты. Другими словами, один отдел не понимает проблему, а другой — ее решение.

Во-вторых, присутствуют сложности с обоснованием необходимости расходов. Покупка и внедрение новых технологий требуют финансирования, однако отдача от них ощущается не сразу и не поддается простому количественному определению. В большинстве случаев для обоснования предстоящих затрат требуется представить доказательства, что проект окупится в течение первого года. Поэтому несмотря на очевидные достоинства

OLAP — точность данных, легкость анализа и использования, а также возможность своевременного получения информации — представление обоснования его использования является непростой задачей [8].

Список литературы

1. **Некрасов В.** 30 идей применения OLAP. URL: http://www.olap.ru/basic/30ideas_olap.asp
2. **Принятие** внешнеполитических решений в России и США. URL: http://www.intertrends.ru/five/005.htm
3. **Некрасов В.** Мобильный OLAP // Открытые системы. 2003. № 5. URL: http://www.osp.ru/os/2003/05/183051/
4. **Альперович М.** Введение в OLAP и многомерные базы данных. URL: http://www.cfin.ru/itm/olap/intro.shtml
5. **Федоров А., Елманова Н.** Введение в OLAP // КомпьютерПресс. 2001. № 5. URL: http://compress.ru/article.aspx?id=10628&iid=433
6. **Набибекова Г.** Разработка информационной модели информационно-аналитической системы поддержки принятия решений // Матер. III Междунар. конф. "Проблемы кибернетики и информатики". Баку, 2010, 6—8 сентября. Т.1. С. 135.
7. **Манди Дж.** К вопросу об OLAP / Под ред. Р. Кимбалла // Intelligent Enterprise. 2003. № 22(87). URL: http://www.iemag.ru/platforms/detail.php?ID=16532&sphrase_id=163299
8. **Stoltz S.** OLAP, an alternative technology over spreadsheets. URL: http://hosteddocs.ittoolbox.com/SS110504.pdf

---

УДК 004.624

Т. А. Барбасова, канд. техн. наук, доц.,
Е. В. Вставская, канд. техн. наук, доц.,
В. И. Константинов, доц.,
ФГБОУ ВПО Южно-Уральский государственный университет, г. Челябинск,
e-mail: tatyana_barbasova@mail.ru

Кодирование информации методом пропуска полупериодов сетевого напряжения в системах освещения

Предложен метод кодирования информации пропуском полупериодов напряжения сети. Рассмотренный метод позволяет передавать требуемый объем информации, используя минимальное число пропусков полупериодов сетевого напряжения. Метод применим в системах освещения, силовым элементом которых является электронный блок с коммутацией при переходе сетевого напряжения через ноль.

Ключевые слова: *кодирование информации, управление источниками света, передача информации*

Введение

Передача информационных сообщений по сетям питающего напряжения является актуальной научно-технической задачей, поскольку позволяет создавать интеллектуальные системы индивидуального управления на базе существующих линий кабельной разводки, обеспечивающих различные устройства питающим напряжением. Наиболее эффективно управление по цепи питающего напряжения можно применять в осветительных системах. Одним из наиболее эффективных методов передачи информации является применение широтной модуляции полупериодов сетевого напряжения в диапазоне малых начальных углов, поскольку при таком способе кодирования информации энергетические свойства питающей сети не нарушаются. Однако использование такого метода передачи информации требует применения специальной схемы электронного ключа [5], особенно для потребителей с емкостным характером входного сопротивления.

Основную массу электронных коммутаторов в системах освещения составляют блоки, силовым элементом которых являются тиристоры или симисторы с коммутацией при переходе сетевого напряжения через ноль. Такие блоки представляют собой оптронные реле переменного тока, управление которыми осуществляется стандартным цифровым сигналом выхода микроконтроллера или логического элемента. Использование силовых элементов с коммутацией при переходе сетевого напряжения через ноль не позволяет формировать широтно-модулированное напряжение сети, поскольку начало полупериода всегда соответствует нулю сетевого напряжения. Поэтому способы кодирования, связанные с широтной модуляцией начального угла полупериода сетевого напряжения, для управления объектами освещения [3, 6] в таких системах непригодны. Для реализации управления в системах, имеющих стандартное сетевое коммутационное оборудование, можно использовать метод передачи информации по питающей сети, основанный на использовании пропусков целых полупериодов сетевого напряжения. Однократный пропуск полупериода сетевого напряжения является для большинства осветительных нагрузок с внутренними электронными преобразователями штатной ситуацией, поскольку электронные модули, как правило, содержат емкостные накопители энергии, рассчитанные на такой режим работы [1].

Цель работы

Для реализации информационной посылки необходимо разработать метод, который бы основывался на изменении последовательности чередования пропусков полупериодов сетевого напряжения, несущей определенный объем информации. Причем пропуски полупериодов сетевого напряжения должны быть сформированы так, чтобы изменение действующего значения сетевого напряжения составляло не более 10 %, для того чтобы питание всех устройств от сети не вызывало перебоев и все элементы системы работали нормально.

Исходя из вышеуказанного формируются следующие требования к разработке метода кодирования информации:

- минимальное число пропущенных полупериодов напряжения сети;
- уменьшение результирующего действующего значения напряжения сети не более, чем на 10 % за счет пропусков полупериодов сетевого напряжения;
- возможность передачи требуемого объема информации.

Предлагаемый метод кодирования информации предполагает изменение положения пропущенного полупериода в составе посылки.

Формирование метода кодирования информации

Суть метода заключается в том, что информацию несет количество полупериодов сетевого напряжения, расположенных между двумя пропущенными полупериодами.

Причем число не пропущенных полупериодов между пропущенными не должно быть меньше определенного, поскольку в противном случае мы можем получить уменьшение действующего значения сетевого напряжения более чем на 10%.

Определим минимальное число полупериодов сетевого напряжения, расположенных между двумя пропущенными.

Действующее значение напряжения сети вычисляется по формуле [2]

$$U = \sqrt{\frac{1}{T}\int_0^T u^2(t)dt}. \qquad (1)$$

Для синусоидального сигнала действующее значение напряжения определяется как

$$U = \frac{U_m}{\sqrt{2}}, \qquad (2)$$

где U_m — амплитуда синусоидального сигнала.

Поскольку действующее значение напряжения не зависит от знака интегрируемого напряжения,

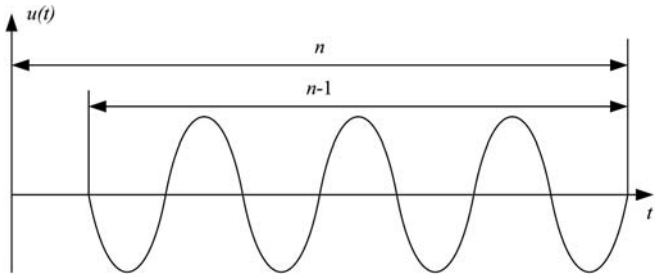


Рис. 1. Напряжение из последовательности полупериодов с одним пропущенным

действующее значение за полупериод определяется по формуле (2).

В случае, если один из полупериодов напряжения пропущен, действующее значение зависит от числа n рассматриваемых полупериодов (рис. 1).

При этом период определения действующего значения зависит от числа рассматриваемых полупериодов:

$$U = \sqrt{\frac{n-1}{n}\frac{Um^2}{2}} = \frac{U_m}{\sqrt{2}}\sqrt{\frac{n-1}{n}}.$$

Зависимость действующего значения напряжения от числа рассматриваемых полупериодов с одним пропущенным представлена на рис. 2.

Отклонение действующего значения напряжения от номинального на 10 % обеспечивается при расчете действующего значения шести полупериодов.

Кодирование информации осуществляется числом полупериодов сетевого напряжения между двумя пропущенными. При этом учитывается минимальное число полных полупериодов (5), не участвующих в кодировке. Для кодирования удобно использовать шестнадцатеричные цифры. Так, цифра 0 кодируется как 6 полупериодов между двумя пропущенными, числу 1 соответствует 7 полупериодов и т. д. Каждый пропущенный полупериод является началом отсчета для приема следующей цифры. На

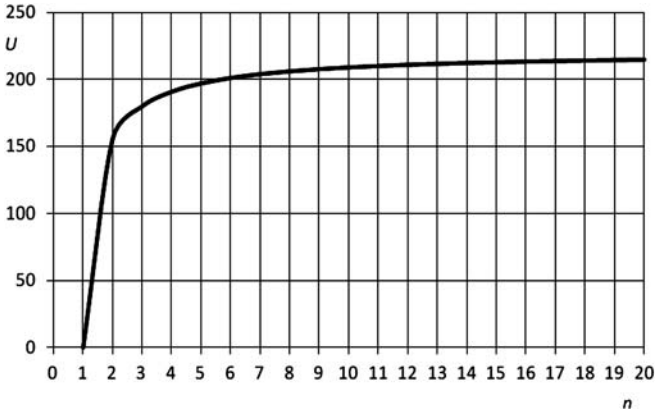


Рис. 2. Зависимость действующего значения напряжения от числа рассматриваемых полупериодов с одним пропущенным

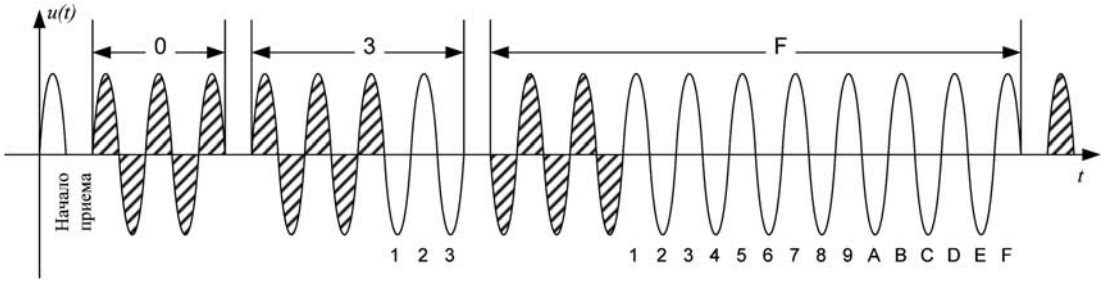


Рис. 3. Пример кодировки шестнадцатеричной посылки 03F рассматриваемым методом

рис. 3 представлен пример кодировки посылки рассматриваемым методом. Закодирована шестнадцатеричная посылка 03F.

Выбор в качестве кодируемых значений шестнадцатеричных цифр обусловлен тем, что при увеличении разрядности представления единиц посылки, например до 1 байта, резко возрастает длительность передачи посылки. При уменьшении единицы представления, например до 2 бит, увеличивается число пропущенных полупериодов, требуемых для представления посылки.

Предложенный метод позволяет кодировать информацию пропусками полупериодов сетевого напряжения и может использоваться в системах освещения, в которых силовые блоки коммутируются при переходе сетевого напряжения через ноль.

Список литературы

1. **Автоматизированные** системы управления энергоэффективным освещением / Под ред. Л. С. Казаринова / Казаринов Л. С., Шнайдер Д. А., Барбасова Т. А., Вставская Е. В. и др. Челябинск: Издательский центр ЮУрГУ, 2011. 208 с.

2. **Вставская Е. В.** Регулятор мощности нагревателя с микропроцессорным управлением // Автоматизация и управление в технических системах: Тематический сборник научных трудов. Челябинск: Изд. ЮУрГУ, 2000. С. 82—85.

3. **Вставская Е. В., Костарев Е. В.** Способ передачи информации по питающей сети и его применение в построении систем автоматизированного управления наружным освещением // Вестник Южно-Уральского государственного университета. Сер. Компьютерные технологии, управление, радиоэлектроника. 2010. Вып. 13, № 2 (219). С. 81—85.

4. **Казаринов Л. С., Вставская Е. В., Константинов В. И., Барбасова Т. А.** Проектирование светодиодных источников света по максимуму функционального резерва при ограничении на весогабаритные характеристики // Вестник Южно-Уральского государственного университета. Сер. Компьютерные технологии, управление, радиоэлектроника. 2011. Вып. 13, № 2 (219). С. 74—81.

5. **Константинов В. И., Вставская Е. В., Барбасова Т. А., Волков В. О.** Выбор оптимального режима работы светодиодных излучателей // Вестник Южно-Уральского государственного университета. Сер. Компьютерные технологии, управление, радиоэлектроника. 2010. Вып. 11, № 2 (178). С. 46—51.

6. **Патент 99913 Российской Федерации.** МПК H 04 B 3/56. Устройство для приема-передачи информации по питающей сети и управления режимами работы потребителей электрической энергии / Барбасова Т. А., Вставская Е. В., Константинов В. И., Константинова О. В., Костарев Е. В. — № 2010128856/09; заявл. 12.07.2010; опубл. 27.11.2010, Бюл. № 33 (IV ч.).

CONTENTS

Alekperov R. K. *Creation of Computing Environments Based on Cloud Computing Technology* 2

The paper investigates the development principles of distributed computing environments based on cloud computing technology. Analysis shows that the "cloud computing" comprises the following advantages: an increase in available computing power, providing users with unlimited disk space, high-speed data processing, payment of the actual use of computer resources, etc.

Keywords: distributed computing, cloud computing, computing system, services of cloud computing, computing environment, loading distribution

Salibekyan S. M., Panfilov P. B. *Object-Attribute Architecture is a New Approach to Object Systems Developing* 8

In the paper there is assertion that now most popular object-oriented programming paradigm (OOP) isn't sufficient for modern computer engineering, as reuse use is limited, encapsulation is inconvenient for distributed and parallel computing systems, no support of program and data isomorphism, etc.; and author suggest a new programming paradigm named object-attribute which is development of OOP. Suggest paradigm overcome OOP shortcomings and provide a fully fledged dataflow computing system work regime.

Keywords: programming paradigm, object-attribute architecture, object-oriented paradigm, program and data isomorphism, encapsulation, computing parallelism, machine intelligence, distribute computing system

Dokuchaev A. N. *Real-Time Scheduling of the Extra Light Tasks on Multiprocessors* 14

The real-time scheduling of systems of periodic tasks on symmetric multiprocessor platforms is considered in this paper. We propose a set of prerequisites for the new techniques of real-time scheduling on multiprocessors based on the concepts of "extra light task" and Dhall's effect. According to the simulation results, we also consider the statement of the extra light scheduling problem.

Keywords: real-time systems, symmetric multiprocessing, global priority-driven multicore scheduling, Dhall's effect, scheduling of the extra light tasks

Norenkov I. P., Uvarov M. Yu. *Knowledge Management on the Base of Ontologies Clasterization* 19

The method of role clasterization of ontologies is suggested. An ontology is subdivided into some clasters and textmining problems are solved on the base of complex concepts. generated from members of different clasters. The examples of annotating, classification, information retrieval, search of problem decision are given.

Keywords: knowledge management, clasterization of ontologies, document classification and annotation, decision support

Bolkhovityanov A. V., Chepovskiy A. M. *Method of Cognitive Semantic Analysis of Russian Sentence* 25

In this paper, we propose two mathematical models intended for analyzing the russian sentence to detect noun phrases and participial clauses. Algorithm for participial clause identification is based on the concept of syntactic relation between verb and dependent syntactic units in the russian language. Considered algorithms designed on the basis of the proposed models can improve procedure of syntactic parsing.

Keywords: natural language processing, noun phrase, syntactic relation between verb and dependent syntactic units, cognitive semantic analysis

Levkov A. A. *Data and Knowledge Integration in Large Information Systems* 29

A method of ontology organization by clearly defined typed concept hierarchy is offered. It allows direct transform ontology into stored facts structuring through relation database schema and procedural extensions without impedance mismatch.

Keywords: semantic model, relational system, description logic, model translation, hierarchical-relational database scheme

Bochkov M. V., Boykov P. N. *Way of Automatic Classification of Not Structured Information of a Network the Internet* 34

The given article suggests a type of classification of hypertext documents which is gathered from the unstructured sources of information and is based on automatic choice of useful information and elimination of html-page service data.

Keywords: processing of html-pages, singular value decomposition, LSA

Merkusheva A. V., Malychina G. F., Malykhin V. M. *Structures, Methodes and Algorithms for Adaptive Filtering of Nonsyationary Signals* 37

Methods, algorithms and schemes of computation are considered for constructing adaptive filter system being oriented on identification and control of linear system. Elements of mathematical basis are given for constructing adaptive filters of different complexity and functionality quality measure. A number of adaptive filters with fast convergence are considered and adaptive algorithms oriented on data blocks input. Computation complexity is given for several algorithms.

Keywords: systems, control, identification, non-stationary signals, filtering, methods, algorithms, computation scheme structures, computation complexity

Volovikov V. V., Sarafanov A. V. *Modeling of Thermal Resistance During Forced Convection Heat Transfer within Radio Electronic Appliances* 44

This article discusses ways to reduce errors in the computation of thermal resistance during forced convection heat transfer within radio electronic appliances. This resistance uses to build electrotermal analogy models. The article contains equations of heat transfer for laminar, mixed and turbulent flow in circular and rectangular cross-section ducts. First way to reduce errors is based on taking into account width and height ratio of a rectangular duct during turbulent flow. Also the article suggests one again way to accurate calculating of thermal resistance computation which is based on dividing ducts into uniform temperature parts.

Keywords: computation of heat transfer, radio electronic appliances, electrotermal analogy, forced convection within ducts, reduce errors, thermal resistance

Bobkov V. A., Melman S. V., May V. P. *Visualization of Cyclones Dynamics* 49

This work is devoted to developing tools for the visual analysis of synoptic objects, especially tropical cyclones based on satellite data. In the presented system the spatio-temporal visualization method of dynamics of a cyclone is realized. This method uses temperature anomalies calculated on the satellite data. CUDA technology and shaders were used to increase the performance of processing and visualization of data.

Keywords: tropical cyclones, visualization of scalar fields, animation, temperature anomalies, shaders, CUDA

Mesentsev Yu. A., Pavlov P. S. *About Single Class Discrete Optimization Problems with Semidefinite Relaxation Decomposition Algorithm Program Implementation* 54

Universal mathematical economic model has been examined. This models' purpose is to define optimal management strategy for logistic subsystems of enterprise. Approximate effective algorithm has been implemented for real dimension count problems with high complexity.

Keywords: decomposition, semidefinite programming problem, discrete optimization, relaxation, efficient algorithm

Topka V. V. *Project Cost Management under Taking into Account its Reliability Index* 60

An Activity-on-Nod network dependability model is proposed as a complex technical system whose resulted secure lower estimate is considered to be its reliability parameter. On the basis of this model, the scheduling problem is stated not only under resource constraints, but also in terms of project dependability. The problem considered is minimizing a project cost under the duration and reliability estimate constraints in a determinate conjunctive project model with the source data known inexactly. The regularized twicesteps method of linearization for ill-conditioned problems is recommended to solve the stated problem.

Keywords: project management, network, reliability, optimization

Borovski A. S., Tarasov A. D. *Method of Using Expert Information to Development Physical Defense System Based on Fuzzy Hypergraphs* 67

It is describing method of expert information using for automatic decision making in physical defense system structure definition problem. Method based on general model of physical defense system operation process which descript in fuzzy hypergraphs form.

Keywords: physical defense system, linguistic variable, fuzzy hypergraphs, hypergraphs composition

Nabibekova G. Ch. *Application of Olap-Technology in Decision Making Support Systems in the Sphere of Foreign Policy* 73

This article examines the possibility of using OLAP-technologies in the information and analytical systems of decision making support in the sphere of foreign policy. Aiming at this, the problems arising while making foreign decisions, the essence of the multidimensional data model, functions and structure of data warehouse, multidimensional OLAP-cube are studied.

Keywords: information and analytical systems for decision making support, data warehouse, OLAP-technology, multidimensional data model, OLAP-cube

Barbasova T. A., Vstavskaya E. V., Konstantinov V. I. *Encoding Information Using Method of Supply Voltage Half-Cycle Passing in Lighting Systems* 76

An encoding information method using supply voltage half-cycle passing is offered. The method allows transmitting information using minimum of passed half-cycles. The method is applicable in lighting systems with power electronic block using zero-cross voltage commutation.

Keywords: information encoding, lighting sources control, information transmit

Адрес редакции:

107076, Москва, Стромынский пер., 4

Телефон редакции журнала **(499) 269-5510**

E-mail: it@novtex.ru

Дизайнер *Т.Н. Погорелова*. Технический редактор *Е. В. Конова*.

Корректор *Е.В. Комиссарова*.

Сдано в набор 07.12.2011. Подписано в печать 24.01.2012. Формат 60×88 1/8. Бумага офсетная.

Усл. печ. л. 9,8. Заказ IT212. 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.

Оригинал-макет ООО "Адвансед солюшнз". Отпечатано в ООО "Адвансед солюшнз".

105120, г. Москва, ул. Нижняя Сыромятническая, д. 5/7, стр. 2, офис 2.