ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

2(174) 2011

ТЕОРЕТИЧЕСКИЙ И ПРИКЛАДНОЙ НАУЧНО-ТЕХНИЧЕСКИЙ ЖУРНАЛ

Издается с ноября 1995 г.

УЧРЕДИТЕЛЬ
Издательство "Новые технологии"

СОДЕРЖАНИЕ

Информация о журнале доступна по сети Internet по адресу http://www.**informika.ru/text/magaz/it/** или **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 004.9

А. Г. Трофимов, канд. техн. наук, доц.,
В. И. Скругин, аспирант,
Национальный исследовательский
ядерный университет "МИФИ"
e-mail: goodthings@ya.ru

Системы нейрокомпьютерного интерфейса. Обзор

Приводится обзор систем нейрокомпьютерного интерфейса. Описываются различные типы интерфейсов, принципы их работы, области применения, основные проблемы и наметившиеся тенденции в этой области. Перечисляются основные научные группы, занимающиеся исследованиями нейрокомпьютерных интерфейсов. Особое внимание уделяется неинвазивным интерфейсам, основанным на анализе электроэнцефалограмм.

Ключевые слова: *системы нейрокомпьютерного интерфейса, интерфейс "мозг—компьютер", количественная электроэнцефалография, электроэнцефалограмма*

Введение

Многие ученые утверждали и утверждают, что изучение мозга является одним из важнейших вопросов, стоящих перед человечеством. Согласно Ф. Крику, открывшему двойную спираль ДНК, "для человека не существует более жизненно необходимого научного исследования, чем исследования его собственного мозга. Весь наш взгляд на вселенную определяется этим".

Наиболее простым и дешевым способом регистрации активности мозга являются электроэнцефалография и томография. Эта активность может отражать как психологическое состояние испытуемого, так и характер его мыслительной деятельности [1, 2]. До конца XX века исследования электрической активности мозга человека проводились преимущественно в целях диагностики патологии мозга, прежде всего, эпилепсии. Однако в последнее десятилетие интерес к электроэнцефалограммам (ЭЭГ) и томограммам связан с еще одной причиной — попыткой использования этой информации об активности нервных клеток мозга для управления техническими устройствами. Такого рода устройства, называемые *интерфейсами "мозг—компьютер*" (ИМК), или нейрокомпьютерными (мозго-машинными) интерфейсами (Brain—Computer Interface, BCI, Brain—Machine Interface, BMI), могут применяться в медицине, военной сфере, промышленности, а также являться основой коммуникационных систем нового поколения. Некоторые исследователи ставят задачу шире — на основе информации об электрической активности нейронов мозга определить, о чем думает человек или хотя бы определить тип его мыслительной деятельности.

Основное назначение систем мозго-машинного интерфейса — это представление человеку нового канала взаимодействия с внешним миром по сравнению с обычным каналом посредством периферийных нервов и мышц. В системах мозго-машинного интерфейса управление внешними техническими устройствами осуществляется непосредственно с помощью электрической активности нейронов мозга. В основе систем нейрокомпьютерного интерфейса лежит математический анализ нейрофизиологических данных и алгоритмы их распознавания.

Мозго-машинные интерфейсы в США — сейчас одна из самых актуальных и продвигаемых тем. Интерфейсом мозг—компьютер занимается ряд университетов и коммерческих фирм, в частности, CSU (Colorado State University), Graz University of Technology, Cyberkinetics Neurotechnology Systems, IBM и др. В 2009 г. специалистам японской компании Honda удалось создать систему управления роботом Asimo "силой мысли".

Исследования в области BCI являются междисциплинарными. Трудно найти в прикладной науке еще такую область, которая объединяла бы специалистов из столь многих отраслей, среди которых — медицина, психология, нейробиология, искусственный интеллект и машинное обучение, анализ сигналов, распознавание образов. Для построения систем BCI необходимо тесное сотрудничество специалистов из этих областей.

Развитие систем мозго-машинного интерфейса

После открытия Г. Бергером в 1929 г. метод электроэнцефалографии вплоть до настоящего времени применяется в основном в трех направлениях [3]:

- диагностика неврологических расстройств в клиниках и госпиталях;
- исследование функций мозга в нейрофизиологических лабораториях;
- нейротерапия с использованием биологической обратной связи.

В течение многих лет выдвигались предположения (научные и околонаучные), что ЭЭГ можно также использовать для управления внешними механическими или электрическими устройствами. Это четвертое применение ЭЭГ и составляет основу интерфейса "мозг—компьютер". Цель разработок в области ИМК — создание устройств непосредственной связи между мозгом человека и механическими или электрическими устройствами [4].

В литературе приводится множество определений термина BCI, среди которых наибольшее признание получило определение, данное в работах Дж. Уолпау: BCI — это коммуникационная система, в которой сообщения или команды, посылаемые индивидуумом во внешний мир, не проходят через обычные нормальные выходные каналы мозга в виде периферийных нервов и мышц [4].

Согласно этому определению движения глаз не могут лежать в основе BCI-систем. В системах мозго-компьютерного интерфейса используются либо биопотенциалы мозга, зарегистрированные с поверхности кожи головы, либо биопотенциалы, зарегистрированные непосредственно с поверхности коры головного мозга или от отдельных нейронов глубинных структур мозга. В первом случае ИМК называются *неинвазивными*, во втором — *инвазивными*. Основой неизвазивных ИМК являются ЭЭГ, томограмма и пр., основой инвазивных ИМК — электрокортикограмма (ЭкоГ) либо карты электрической активности отдельных глубинных нейронов.

В литературе встречаются также и другие определения BCI:

BCI — это интерфейс между человеком и компьютером, который получает команды напрямую от мозга без совершения какого-либо физического движения [5];

BCI — это система, которая использует электрофизиологические сигналы для управления внешними устройствами [6].

В системах BCI возможна также и обратная связь от технического устройства к мозгу. Иногда такие интерфейсы выделяют отдельно и называют CBI (*computer-to-brain interface*): CBI — это система реального времени, используемая для записи сообщений или команд прямо в мозг без использования обычных входных каналов мозга [7].

Первые попытки создания ИМК начались в 1980-х гг. в Германии и США. Их цель была помочь полностью парализованным людям (с так называемым locked-in-синдромом) управлять инвалидным креслом и общаться. В [8] описана система ИМК, анализирующая сигналы ЭЭГ таких пациентов и относящая их к одному из двух классов, которые интерпретировались как ответы "да" или "нет". Примерно в то же время Питер Розенфельд (Peter Rosenfeld) из Северо-Западного университета, анализируя электроэнцефалограммы, обнаружил, что ложь вызывает определенные изменения картины ЭЭГ. С этого момента начались исследования ИМК в немедицинских целях.

В 1988 г. Фарвел (Larry Farwell) и Дончин (Emanuel Donchin) [9] впервые реализовали систему "виртуальной клавиатуры", позволившей печатать текст силой мысли. Работа системы основана на анализе ЭЭГ, а именно, компонента P300 при съеме зрительных вызванных потенциалов, возникающих как реакция на визуальный стимул. После этого было разработано много различных модификаций BCI-систем с все возрастающими возможностями, уже нашедшими свое применение как в клинике для общения с пациентами, полностью утратившими возможность движения [8], так и для дистанционного управления роботами [10].

Первый инвазивный ИМК, использующий внутрикорковые потенциалы мозга обезьяны, был разработан Ф. Кеннеди (Philip Kennedy, основатель компании Neural Signals Inc.) в 1987 г. [11]. В 1999 г. исследователи из Калифорнийского Университета Беркли, возглавляемые Я. Деном (Yang Dan), декодировали активность нейронов таламуса кошки, возникающую при предъявлении зрительных стимулов [12].

Одними из наиболее известных исследований в области инвазивных ИМК являются исследования группы М. Николелиса (Miguel Nicolelis). Успех исследований на крысах в 1990-х гг. [13] вдохновил Николелиса на разработку ИМК, который бы декодировал нейронную активность обезьян и транслировал ее в движения механической руки. Такие исследования были проведены несколько лет назад и показали, что обезьяны демонстрируют хорошие способности манипуляции механической рукой, с помощью которой они брали пищу из руки экспериментатора и подносили ко рту.

ИМК предоставляют человеку новый канал коммуникации с внешним миром (рис. 1). В отличие от существующего канала, когда электрическая активность мозга проходит через периферическую нервную систему к анализаторам или мышцам, в ИМК осуществляется передача информации непосредственно от коры больших полушарий к техническому устройству.

В общем случае архитектура ИМК включает следующие компоненты (рис. 2):

- регистрация электрической активности мозга (инвазивным или неинвазивным способом);
- предобработка данных (фильтрация от шумов и удаление компонентов ЭЭГ, не связанных с мозговой деятельностью — артефактов);
- выделение признаков ЭЭГ, характеризующих тип электрической активности мозга;
- классификация выделенных признаков (распознавание типа мыслительной деятельности);

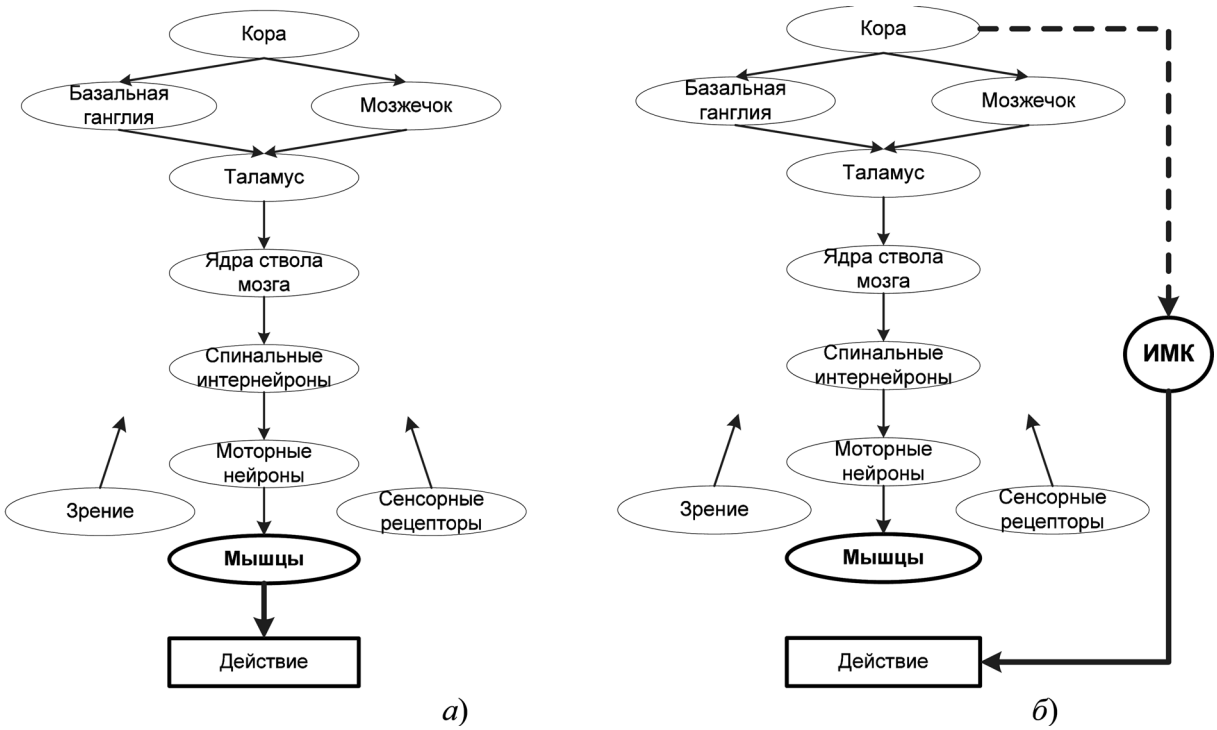


Рис. 1. Новый канал коммуникации человека с внешним миром:
а — упрощенная схема генерации моторного акта с использованием стандартных нервных путей; *б* — схема генерации действия посредством системы ИМК (из [14])

Рис. 2. Архитектура ИМК

- постобработка результатов классификации и генерация команд техническому устройству. Типы команд могут быть очень различными в зависимости от конкретного ИИМК. Например, это может быть перемещение курсора на экране компьютера, включение/выключение света, движение механической рукой и т. п.);
- обратная связь с пользователем (neurofeedback). Обратная связь используется для адаптации пользователя к ИМК. Это может быть визуальная, слуховая, тактильная или иная связь. В некоторых ИМК обратная связь отсутствует.

Классификация систем мозго-машинного интерфейса

В литературе встречается множество классификаций систем ИМК. Ранее было упомянуто разделение ИМК на инвазивные и неинвазивные. В рамках данной работы наибольшее внимание уделено неинвазивным ИМК. Согласно [4] существующие неинвазивные нейрокомпьютерные интерфейсы по принципу работы подразделяются на три типа.

1. ***ИМК, основанные на анализе медленных корковых потенциалов, SCP-BCI.*** Медленные корковые потенциалы (*slow cortical potentials*, *SCP*) воз-

никают как результат позитивной или негативной поляризации коры больших полушарий мозга, которая длится от 300 мс до нескольких секунд. Эта поляризация может быть зарегистрирована с помощью ЭЭГ. Принцип работы SCP-BCI основан на сознательной генерации и удержании испытуемым интенсивности своей корковой поляризации [15]. Использованию таких систем предшествует, как правило, длительная процедура обучения испытуемого, в ходе которой он научается управлять своей корковой поляризацией. Такие ИМК, как правило, снабжены обратной связью (например, через экран монитора), посредством которой испытуемому предоставляется информация о поляризации коры своего мозга (*нейробиологическая обратная связь, neurofeedback*). Механизм биологической обратной связи основан на парадигме *оперантного обусловливания*, которая предполагает, что испытуемые способны научиться сознательно изменять электрическую активность своего мозга, наблюдая собственную электроэнцефалограмму на экране монитора [15].

ИМК, основанные на анализе медленных корковых потенциалов по ЭЭГ, относятся к так называемым *устройствам передачи мысли* (*Thought Translation Device, TTD*). TTD — это такие ИМК, которые позволяют полностью обездвиженным пациентам общаться с внешним миром, используя только их собственные мозговые потенциалы [15]. Первые успешно работающие TTD-устройства были созданы в Германии в конце 1990-х гг. группой исследователей под руководством Н. Бирбаумера (Niels Birbaumer), известной как Тюбингенская группа (Tubingen group).

В развитие систем SCP-BCI существенный вклад внесли исследования Фетса (Fets) в области оперантного обусловливания, который в 1970-х гг. впервые продемонстрировал, что обезьяны способны быстро обучаться сознательно контролировать активность нейронов в первичной моторной коре при соответствующем поощрении [16].

Следует отметить, что не все люди могут сознательно управлять своей корковой поляризацией. До сих пор до конца непонятно, почему одни люди могут научиться использовать такие ИМК, а другие не могут. По всей видимости, это связано с тем, что ЭЭГ отражают суммарную активность сотен тысяч или даже миллионов нейронов мозга и у некоторых испытуемых этой информации становится недостаточно для успешной классификации [15].

2. ***ИМК, основанные на анализе вызванных потенциалов, ERP-BCI.*** *Вызванный потенциал, ВП* (*event-related potential, ERP*) — это электрическая реакция мозга на внешний раздражитель. В существующих ERP-BCI используются в основном два типа вызванных потенциалов: ВП P300 и ВП SSVEP.

P300 является вызванным потенциалом, который возникает как реакция на стимул, к которому привлечено внимание (например, если человек слышит свое имя в ряду других слов) [17]. Работа P300-BCI заключается в определении стимула среди некоторого множества, на который у испытуемого возникает потенциал P300 — это тот стимул, к которому привлечено внимание. Например, последовательно предъявляя испытуемому различные буквы или слова и регистрируя ЭЭГ-активность, может быть определена та буква или слово, которая вызывает потенциал P300, тем самым, последовательно меняя объект внимания, испытуемый может конструировать слова или предложения. Наиболее известные P300-BCI разрабатываются в Бекманском университете исследователями под руководством Э. Дончина (Emanuel Donchin), а также в Пекине, Сингапуре.

Вызванный потенциал SSVEP (*steady state visual evoked potential*) возникает преимущественно в затылочных частях коры как реакция на визуальные вспышки, повторяющиеся с определенной частотой. Когда сетчатка возбуждается визуальными стимулами частотой от 3,5 до 75 Гц мозг генерирует электрическую активность той же частоты. Как следствие, вызванные потенциалы SSVEP могут быть обнаружены на спектрограмме соответствующих отведений ЭЭГ. Большинство P300-BCI и SSVEP-BCI используют виртуальную клавиатуру и применяются для набора испытуемым сообщений. Исследования SSVEP-BCI описаны в работах [18—21].

3. ***ИМК, основанные на анализе сенсомоторных ритмов, ти-BCI, SMR-BCI.*** Явление перестройки ритмической активности мозга в ответ на внешние события известно очень давно. В частности, изменение амплитуды альфа-ритма при открывании и закрывании глаз было описано еще Г. Бергером, основоположником метода электроэнцефалографии [1]. В связи с тем, что усиление ритмической активности означает вовлечение большего числа нейронов в совместную синхронную деятельность, увеличение амплитуды ритма называют его *синхронизацией*. Уменьшение же амплитуды означает рассогласование синхронной деятельности нейронных ансамблей, т. е. наступает *десинхронизация* ритма.

Работа рассматриваемых ИМК основана на анализе сенсомоторных ритмов (*sensorimotor rhythms, SMR*), называемых также *мю-ритмами*, которые имеют частоту 12...15 Гц и возникают в сенсомоторной коре. В состоянии покоя сенсомоторные ритмы имеют высокую амплитуду (наблюдается синхронизация ритма), в то время как при активации соответствующих сенсорных или моторных участков коры их интенсивность падает (наблюдается десинхронизация ритма).

В 1990-х гг. австрийский ученый Пфуртшеллер (Gert Pfurtscheller) начал систематически изучать феномен изменения ритмов при сенсорной стимуляции. В своих работах он показал, что уменьшение интенсивности сенсомоторных ритмов возникает не только при выполнении движения, но также и при *воображении движения* [22]. Также Пфуртшеллер показал, что механизмы *сенсомоторной синхронизации/десинхронизации* (*event-related synchronization/desynchronization, ERS/ERD*) связаны с механизмами работы мозга, отличными от механизмов вызванных потенциалов, что предоставляет ученым дополнительную информацию об исследовании реакции мозга на стимулы [23].

Явление сенсомоторной синхронизации/десинхронизации при воображении движения положено в основу mu-BCI. В таких интерфейсах для управления внешним устройством, например рукой робота, пользователь воображает то или иное движение. Преимуществом таких ИМК является отсутствие необходимости длительного предварительного обучения испытуемого. Наиболее известные исследования mu-BCI описаны в работах Пфуртшеллера [8, 24, 25].

По режиму работы ИМК разделяются на *синхронные* (*synchronous, cue-bases, computer-driven*) и *асинхронные* (*uncued, user-driven*) [22]. В синхронных интерфейсах испытуемый может генерировать управляющий сигнал только в интервалы времени, определяемые внешним устройством. В таких ИМК мозговая активность подлежит классификации лишь в заданные интервалы времени. Напротив, асинхронные ИМК всегда активны и сами детектируют момент времени, когда испытуемый желает подать управляющую команду.

По классификации, приведенной в [22], ИМК разделяются на *использующие внешний стимул* (*stimulated*), ответная активность мозга на который регистрируется (как правило, это вызванные потенциалы), и *нестимулируемые* (*unstimulated*), в которых испытуемый силой воли генерирует электрическую активность нейронов.

По способу генерации электрической активности мозга ИМК разделяются на *зависимые* (*dependent*) и *независимые* (*independent*). Здесь подразумевается зависимость от нервных путей, обеспечивающих изменение этой активности [22]. Зависимые ИМК основаны на том, что изменение ЭЭГ-активности безусловно связано с раздражением каких-либо рецепторов, например сетчатки глаза. В независимых ИМК изменение ЭЭГ-активности зависит только от *намерения* испытуемого. Независимые ИМК, в отличие от зависимых, могут применяться для коммуникации полностью обездвиженных пациентов с locked-in-синдромом. Примером зависимого ИМК является SSVEP-BCI. К независимым ИМК относится, например, mu-BCI.

Согласно классификации К. В. Анохина ИМК могут быть разделены на два типа: *поверхностные* и *глубокие*. К поверхностным ИМК относятся устройства, которые регистрируют суммарные сигналы с поверхности черепа (электроэнцефалографы, магнитоэнцефалографы, функциональная магниторезонансная томография, диффузные оптические томографы, околоинфракрасные спектроскопы или другие устройства). Глубокие интерфейсы — это устройства, которые используют сигналы, отводимые от одиночных нейронов в мозге. Глубокие интерфейсы требуют проникновения устройств считывания в глубину мозга через поверхность черепа.

Однако К. В. Анохин также вкладывает и другой смысл в эти понятия. В случае поверхностных интерфейсов не является необходимым понимание нейрофизиологической природы мысли. В современной нейронауке до сих пор нет эффективной нейрофизиологической теории мозга, тем не менее, существующие поверхностные ИМК могут работать и без нее. В этом смысле поверхностные интерфейсы могут работать, анализируя также данные из глубинных структур мозга.

Глубокие же интерфейсы являются таковыми не только потому, что они проникают в те или иные области под черепом, но и потому, что их разработка и эффективное использование связаны с пониманием глубоких механизмов работы мозга и мышления.

Основываясь на том, как BCI транслирует биопотенциалы мозга, системы мозго-машинного интерфейса разделяются на *прямые* и *непрямые* [22]. Прямые интерфейсы улавливают слово или представление непосредственно в том виде, как оно возникает в мозгу. М. Джаст (Marcel Just) из Университета Карнеги Меллона утверждает, что добился этого с помощью функциональной ядерно-магнитной томографии (фЯМР), введя в работу небольшое число очень простых понятий — названий плотницких инструментов, например, или различных типов построек. Джасту удалось с точностью до 80...90 % определить, о какой категории понятий из 12 заданных думает испытуемый в настоящий момент [26]. Т. Митчел (Tom Mitchell), коллега Джаста, пошел дальше и описал эксперимент, который позволили бы обнаружить слова, создающие наиболее отчетливые паттерны активности мозга.

Работа непрямых интерфейсов основана на распознавании характерных паттернов электрической активности мозга и не связана с определением конкретного представления, вызвавшего эту активность. Например, с помощью таких устройств осуществляется управление курсором на экране вверх или вниз в зависимости от знака корковой поляризации.

Научные группы, занимающиеся исследованиями нейрокомпьютерных интерфейсов

Исследования в области ИМК, которые ограничивались только тремя научными группами 20 лет назад и шестью—восемью группами в 1995 г., в настоящее время демонстрируют огромный рост интереса к этой проблеме. Сейчас насчитывается более 100 групп по всему миру, занимающихся широким спектром исследований и регулярно публикующих свои достижения [14].

Начиная с 1999 г. устраиваются регулярные международные конференции и симпозиумы по BCI. По динамике числа представленных докладов и научных групп очевидно, что проблема мозго-машинных интерфейсов как инвазивных, так и неинвазивных, получает колоссальное развитие в последние годы.

Исследованием и разработкой BCI занимаются множество научных лабораторий и коммерческих организаций по всему миру. В таблице приведен далеко не полный перечень научных групп с указанием направлений исследований.

Наиболее известные разработки в области инвазивных ИМК принадлежат группам Николелиса и Чапина. Они сконструировали интерфейс, позволяющий одновременно регистрировать активность до 100 отдельных нейронов в лобных и теменных долях коры обезьяны и использовать эту активность для управления механической рукой [13].

Ниже представлены направления работ некоторых научных групп, занимающихся исследованиями ИМК.

1. Пекинская группа (Beijing group, рук. X. Gao)

Пекинский ИМК основан на анализе вызванных потенциалов SSVEP. На экране компьютера испытуемый видит матрицу размера 3 × 4, в каждой ячейке которой нарисована цифра (так называемый "виртуальный телефон" [27]). Каждая ячейка подсвечивается с определенной частотой в пределах 6...17 Гц, при этом частоты мерцания всех ячеек различны. В зависимости от того, на какой ячейке сосредоточен испытуемый, в затылочных областях его коры возникают вызванные потенциалы SSVEP соответствующей частоты. Анализируя сигналы ЭЭГ, ИМК выделяет этот вызванный потенциал и выводит соответствующую цифру на экран. В работе [19] показано, что наличие этого потенциала может быть обнаружено по ЭЭГ с помощью спектрального анализа с шириной окна в 1 с. Данный ИМК относится к зависимым, синхронным, использующим внешний стимул. Сайт группы: http://neuro.med.tsinghua.edu.cn/eng/

2. Группа Дончина (Emanuel Donchin group) и группа Аллисона (Brendan Allison group)

ИМК Дончина и Аллисона основаны на использовании вызванных потенциалов P300. На экран компьютера выводятся символы, записанные в ячейках прямоугольной решетки размера 6 × 6 или 8 × 8. В ходе эксперимента через случайные интервалы времени подсвечиваются строки или столбцы этой решетки в случайном порядке. Так, если подсвеченной оказалась строка или столбец, в котором содержится задуманная буква, то на ЭЭГ наблюдается возникновение вызванного потенциала P300. Спустя некоторое время, анализируя

Список научных групп, занимающихся исследованиями BCI

Руководитель	Страна, институт	Направления исследований
Schwartz	Pittsburg, USA	Инвазивные BCI
John Chapin	Rochester, USA	Инвазивные BCI
Miguele Nicolelis	Duke, USA	Инвазивные BCI
Kennedy	Atlanta, USA	Инвазивные BCI
Levine	Michigan, USA	Инвазивные BCI
Jonatan Wolpaw	Albany, NY, USA, Wadsworth Center	BCI2000, работа с пациентами
Emanuel Donchini	Beckman Institute for Advanced Science and Technology, University of Illinois, USA	P300-BCI, "виртуальная клавиатура" (spelling)
Brendan Allison	Bremen University, Germany	P300-BCI, "виртуальная клавиатура" (spelling)
Anderson	University of California, USA	Инвазивные BCI
Sadja and Parra	NY, USA	Быстрая визуальная стимуляция
Niels Birbaumer	Tubingen, Germany	SCP-BCI, TTD-устройства, работа с пациентами
Gert Pfurtscheller	Graz, Austria	ERS/ERD, работа с пациентами
Penny, Roberts, Sycacek	Oxford	Байесовский подход к классификации ЭЭГ
Birch and Mason	UBC, Canada	EEG-BCI
Millan	EPFL, Switzerland	Мысленное управление роботом
Donoghue	Brown University, USA	Инвазивные BCI
Cuntai	Singapore	P300-BCI
Gao	Beijing	SSVEP-BCI
Müller, Blankertz, Gurio	Berlin Institute of Technology, Germany	BBCI

вызванные потенциалы P300, с высокой степенью достоверности можно определить символ, вызывающий этот потенциал [28]. Работа ИМК Дончина и Аллисона продемонстрирована в [И1, И2].

3. Тюбингенская группа (Tubingen group, рук. Niels Birbaumer)

Тюбингенский ИМК основан на использовании медленных корковых потенциалов (SCP-BCI) и относится к TTD-устройствам. Испытуемому предоставляется возможность управления курсором в двух направлениях: вверх или вниз, в то время как курсор движется от левого края экрана к правому. Цель состоит в том, чтобы, мысленно управляя курсором, привести его в одну из областей на правой границе экрана. Движение курсора вверх происходит, если корковые потенциалы являются положительными, и вниз — если отрицательными. Использованию данного ИМК предшествует процедура обучения испытуемого посредством биологической обратной связи, в ходе которой он учится управлять своей корковой поляризацией. Обратная связь осуществляется через экран монитора, на котором испытуемый видит траекторию движения курсора. В работе [29] продемонстрировано, как полностью парализованный пациент с locked-in-синдромом пишет на экране компьютера письмо с помощью данного ИМК. Тюбингенский ИМК относится к независимым, синхронным, нестимулируемым. Сайт группы: http://www.mp.uni-tuebingen.de/

4. Группа университета Грац (Graz group, рук. Gert Pfurtscheller)

Graz-BCI основан на анализе сенсомоторных ритмов (mu-BCI). Для управления курсором или внешним устройством пользователь использует мысленное представление о движении правой рукой, левой рукой или ногой. При таких представлениях наблюдается изменение активности мю-ритма соответственно в левой, правой или центральной областях коры, что фиксируется на ЭЭГ. В работе [22] описано применение данного ИМК для управления искусственной конечностью. Другим применением является управление ракеткой при игре в компьютерный теннис. Сайт группы: http://bci.tugraz.at/staff.html/. Работа Graz-BCI продемонстрирована в [И3].

5. Вадсвортская группа (Wadsworth group, рук. Jonathan Wolpaw)

Вадсвортский BCI также основан на анализе сенсомоторных ритмов (mu-BCI). В данном ИМК анализ интенсивности мю-ритмов в различных участках коры используется для управления курсором на плоскости. В ходе эксперимента на экране компьютера в случайном месте на периферии (или в одном из определенных мест) возникает изображение цели. Управляя курсором из центра экрана, испытуемому ставится задача привести его к этой цели. Это первый неинвазивный ИМК, позволяющий управлять курсором в двумерном пространстве [30]. Сайт группы: http://www.bcire-search.org/

6. Берлинский ИМК (Berlin BCI, BBCI)

Берлинский ИМК является неинвазивным ИМК, основанным на анализе сенсомоторных ритмов (mu-BCI). Для регистрации электрической активности используется ЭЭГ со 128 участками коры. BBCI обнаруживает изменения сенсомоторных ритмов ЭЭГ, возникающих при воображении движения правой или левой рукой, путем анализа активности областей левого и правого полушарий. Основной интерес берлинской группы лежит в разработке мультимедиа-приложений наподобие известных игр Пакман, пинбол или пинг-понг. Как утверждают авторы, для обучения мысленному управлению в игре требуется не более 20 мин.

Исследования BBCI проводятся при сотрудничестве Лаборатории машинного обучения Берлинского технологического института Отдела интеллектуального анализа данных Института Фраунгофера FIRST (IDA group) и факультета нейрофизиологии Берлинского медицинского университета. Возглавляют проект Мюллер (Klaus-Robert Muller), Курио (Gabriel Curio) и Бланкерц (Benjamin Blankertz). Сайт группы: http://www.bbci.de/. Работа BBCI продемонстрирована в [И4, И5, И6].

7. Проект BC12000

BC12000 представляет собой многоцелевую систему, используемую для исследований в области ИМК. Разработка BC12000 была начата в 2000 г. в Вадсвортском центре (Элбани, Нью-Йорк) под руководством Г. Шалка (Gerwin Schalk). BC12000 задумывался в целях сведения воедино и развития существовавших достижений в области ИМК и фактически представляет собой программно-инструментальный комплекс для проведения исследований ИМК. BC12000 состоит из множества подсистем, среди которых — модули регистрации сигналов, модули обработки сигналов, модули исполнения действий и др. В BC12000 реализованы подходы, основанные на анализе вызванных потенциалов P300, медленных корковых потенциалов, сенсомоторных ритмов. Примерами задач, решаемых с помощью BC12000, являются управление курсором в одномерном или двумерном пространстве, печатание букв на экране компьютера, игры в пинг-понг и др. BC12000 свободно доступен академическим и образовательным институтам для использования в исследовательских целях. Сайт

проекта BC12000: http://www.bci2000.org. Работа BBCI продемонстрирована в [И7, И8].

В России группы, занимающиеся исследованиями ИМК, представлены в меньшей степени. Исследования в области инвазивных ИМК практически отсутствуют, о чем говорит отсутствие публикаций в этой области. Среди российских групп, занимающихся проблемами неинвазивных ИМК, можно выделить группу МГУ под руководством проф. Каплана А. Я. (биологический факультет, кафедра физиологии человека и животных). Направления работ: разработка ИМК, основанных на анализе ЭЭГ, исследование способности мозга изменять собственную электрическую активность посредством обратной связи в ИМК, исследования явления неосознаваемого оперантного обусловливания и др. Одним из достижений группы А. Я. Каплана является ИМК, позволяющий управлять игрушечной машинкой на основе анализа мю-ритмов ЭЭГ. Сайт группы: http://brain.bio.msu.ru/

Применения нейрокомпьютерных интерфейсов

Мозго-машинные интерфейсы применяются преимущественно в трех направлениях.

1. Медицина

К ИМК, используемым в медицинских целях, относятся нейропротезы — устройства, которые могут стимулировать или замещать функции нервной системы. Ниже приводятся разновидности ИМК, используемых в медицинских целях, и названия ведущих организаций-производителей:

- слуховые нейропротезы (Advanced Bionics Corp., Medtronic Inc.);
- сетчаточные имплантанты для людей с потерей зрения (Optobionics Inc. — проект Artificial Silicon Retina);
- нейропротезы для сенсомоторного контроля у парализованных людей (Cyberkinetics Inc. — проект Braingate^{TM});
- ИМК для мысленного управления собственной конечностью (группа Университета Грац, Австрия);
- имплантируемые микрочипы для детекции и купирования судорожной активности мозга (группа Университета Дьюка, США);
- нейропротезы для восстановления нарушенных высших нервных функций и памяти у человека.

2. Военная отрасль

Исследования по применению ИМК в военных целях впервые начаты в США и ведутся в настоящее время при финансировании проектов федеральным агентством DARPA (Defence Advanced Research Projects Agency). Ниже приведены некоторые направления военного использования ИМК:

- мозговые нейропротезы для управления беспилотными летательными устройствами;
- создание на поле боя сети дистанционно контролируемых вооруженных и невооруженных устройств;
- контроль вооружений с помощью мыслей;
- беспроводные коммуникации между людьми.

3. Новое поколение информационных и коммуникационных технологий

Исследования в этой области направлены на развитие новых каналов управления компьютером. Некоторые приложения таких ИМК и наиболее известные разработчики приведены ниже:

- "виртуальная клавиатура" (Тюбингенская группа, группа BBCI);
- "виртуальная мышь" (компании Neurosky, Square Enix, OCZ — устройство Neural impulse actuator, NIA);
- детектор лжи (Brain Fingerprinting Laboratories Inc.);
- мониторинг состояния пользователя (Advanced Brain Monitoring Inc.).

Интерес к системам ИМК проявляет также компания Microsoft. Руководитель проекта по разработке интерфейса между мозгом и компьютером Д. Тан говорит, что корпорация собирается создать ИМК, позволяющий мысленно управлять программами и переключать окна в Windows.

Исследования в области мозга начаты компанией IBM. Так, проект IBM Blue Brain ставит цели по моделированию на компьютере человеческого мозга и точной симуляции работы колонок неокортекса.

В августе 2008 г. начат совместный проект Ирвинского университета (Irvine University), Университета Карнеги-Меллон (CMU) и Университета Мериленда (Meryland university), финансируемый агентством DARPA (сумма $4 млн), направленный на разработку устройств синтетической телепатии (шлемы для передачи мысли). Шлем надевается на голову военного (128 электродов). Цель — достичь двойного результата: сделать возможным генерацию сообщения и направить его определенному реципиенту. По оценкам для реализации этого проекта потребуется 15 лет.

Программные системы и инструментальные средства анализа электроэнцефалограмм

Существующие программные средства анализа электроэнцефалограмм условно можно разделить на два класса: профессиональные медицинские программные пакеты и свободно распространяемые программы, разработанные энтузиастами на некоммерческой основе.

1. Профессиональные медицинские программные пакеты

К этой категории можно отнести такие программы, как *NeuroGuide EEG, WinEEG, Neocortex-Pro, Net Station EEG Software, eemagine EEG* и др. Программные пакеты подобного рода продаются как самостоятельный продукт или поставляются в комплекте с соответствующим оборудованием — аппаратурой для фиксации сигналов ЭЭГ. Все эти продукты ориентированы на использование врачами для диагностики заболеваний пациентов, поэтому их основной функционал — визуализация сигналов, их предобработка и картирование. Тем не менее, во многих из них также присутствует функционал для проведения более глубокого математического анализа сигналов ЭЭГ.

Несмотря на очень широкий спектр возможностей, применение таких систем для проведения собственных исследований сомнительно по двум причинам. В первую очередь это цена — такие программные пакеты ориентированы на клиническое, а не частное использование. Второй очень важный фактор — невозможность доработки и подстройки системы под собственные нужды.

2. Открытые разработки

В первую очередь это огромное множество программных продуктов, разработанных любителями. Они нацелены как на изучение и исследование ЭЭГ, так и на разработку систем BCI. Среди них: *Wave++*, *Waili, BWView, BrainBay, SPM, Brainstorm, EEGLab* и др. Также к этой категории относится множество разработок различных университетов и исследовательских групп, работающих в области анализа ЭЭГ. Как правило, это самые функциональные и совершенно открытые продукты, чаще всего это пакеты расширения для MatLab. Это является большим плюсом, так как их программный код доступен и является легко расширяемым. Можно без особых проблем результаты работы одной программы затем обрабатывать другими модулями MatLab.

Заключение

В работе представлен обзор существующих систем нейрокомпьютерного интерфейса — коммуникационных систем, в которых сообщения или команды, посылаемые индивидуумом во внешний мир, не проходят через обычные нормальные выходные каналы мозга в виде периферийных нервов и мышц, а формируются на основе результатов анализа электрической активности нейронов мозга.

Проведена классификация, описана общая архитектура BCI, указаны наиболее значимые зарубежные разработки в области систем мозго-машинного интерфейса, приведены краткие описания направлений работ основных научных групп, занимающихся проблемами BCI, и созданных ими реальных BCI-систем. Систематизированы области применения систем нейрокомпьютерного интерфейса и проведен анализ существующих систем количественного анализа электроэнцефалограмм.

В результате проведенного обзора установлено, что системы нейрокомпьютерного интерфейса в настоящее время имеют активное развитие, число научных групп, занимающихся этой проблематикой, за последнее десятилетие увеличилось на порядок. Причиной этому стали как достижения в изучении наук о мозге, так и стремительное развитие вычислительной техники. На сегодняшний день возможность управлять техническими устройствами посредством мысли — это уже реальность. Тем не менее, при построении систем BCI существует масса проблем, решение которых может быть найдено только в результате тесного сотрудничества специалистов в области медицины, технологии, нейробиологии, математики и информатики.

Настоящая работа выполнена в рамках ФЦП "Научные и научно-педагогические кадры инновационной России" на 2009—2013 годы, а также при финансовой поддержке Совета по Грантам Президента РФ для поддержки молодых российских ученых (Грант МК-4245.2009.8).

Список литературы

1. **Гусельников В. И.** Электрофизиология головного мозга. М.: Высшая школа. 1976.

2. **Жадин М. Н.** Биофизические механизмы формирования электроэнцефалограммы. М.: Наука, 1984.

3. **Зенков Л. Р.** Клиническая электроэнцефалография (с элементами эпилептологии). М.: Медпресс-информ, 2004.

4. **Wolpaw J. R., Birbaumer N., McFarland D. J., Pfurtscheller G., Vaughan T. M.** Brain-computer interfaces for communication and control // Clinical Neurophysiology. 2002. V. 113. P. 767—791.

5. **Levine S. P., Huggins J. E., Fessler J. A.** etc. University of Michigan Direct Brain Interface 2002 Update // Brain-Computer Interfaces for Communication and Control. 2002. 54 p.

6. **Bayliss J. D.** A Flexible Brain-Computer Interface // PhD thesis. Department of Computer Science University of Rochester. 2001.

7. **Danilov Y., Tyler M.** BrainPort: an alternative input to the brain // Journal of Integrated Neuroscience. 2005. V. 4. N 4. P. 1—14.

8. **Neuper C., Muller G., Kubler A., Birbaumer N., Pfurtscheller G.** Clinical application of an EEG-based brain-computer interface: a case study in a patient with severe motor impairment // Clinical Neurophysiology. 2003. V. 114. P. 399—409.

9. **Farwell L. A., Donchin E.** Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials // Electroenceph. Clin. Neurophy. 1988. V. 70. P. 510—523.

10. **Millan J., Renkens F., Mourino J., Gerstner W.** Non-Invasive Brain-Actuated Control of a Mobile Robot by Human EEG // IEEE Trans. on Biomedical Engineering. 2004. N 51. P. 1026—1033.

11. **Kennedy P. R., Bakay R. A.** Restoration of neural output from a paralyzed patient by a direct brain connection // Neuroreport. 1998. V. 9. N 8. P. 1707—1711.

12. **Chen G., Dan Y., Li C.** Stimulation of non-classical receptive field enhances orientation selectivity in the cat // J. Physiol. 2005. V. 564. P. 233—243.

13. **Nicolelis M. A.** Brain-machine interfaces to restore motor function and probe neural circuits // Nat. Rev. Neurosci. 2003. V. 4. P. 417—422.

14. **Wolpaw J. R.** Brain-computer interfaces as new brain output pathways // The Journal of Physiology. 2007. V. 579. P. 613—619.

15. **Hinterberger T., Mellinger J., Birbaumer N.** The Thought Translation Device: Structure of a multimodal brain-computer communication system // Neural Engineering, Conference Proceedings. First International IEEE EMBS. 2003. P. 603—606.

16. **Fetz E. E.** Operant conditioning of cortical unit activity // Science. 1969. V. 163. P. 955—958.

17. **Rastjoo A., Arabalibeik H.** Evaluation of Hidden Markov Model for p300 detection in EEG signal // Studies in health technology and informatics. 2009. V. 142. P. 265—267.

18. **Muller-Putz G., Scherer R., Brauneis C., Pfurtscheller G.** Steady-state visual evoked potential (SSVEP)-based communication: impact of harmonic frequency components // J. Neural Eng. 205. N 2. P. 123—130.

19. **Lin Z., Zhang C., Wu W., Gao X.** Frequency Recognition Based on Canonical Correlation Analysis for SSVEP-Based BCIs // Transactions IEEE on Biomedical Engineering. 2006. V. 53. N 12. Part 2. P. 2610—2614.

20. **Martinez P., Bakardjian H., Cichocki A.** Fully-Online, Multi-Command Brain Computer Interface with Visual Neurofeedback Using SSVEP Paradigm // Computational Intelligence and Neuroscience. 2007. V. 2007. P. 13—41.

21. **Beverina F., Palmas G., Silvoni S., Piccione F., Giove S.** User adaptive BCIs: SSVEP and P300 based interfaces // PsychNology Journal. 2003. V. 1. N 4. P. 331—354.

22. **Pfurtscheller G., Neuper C., Birbaumer N.** Human brain-computer interface // Motor cortex in voluntary movements. 2005. P. 367—401.

23. **Pfurtscheller G., Lopes da Silva F.** Event-related EEG/MEG synchronization and desynchronization: basic principles // Clin Neurophysiol. 1999. V. 110. N 11. P. 1842—1857.

24. **Pfurtscheller G., Neuper C.** EEG-based brain-computer interfaces // Electroencephalography: basic principles, clinical applications and related fields. 2005. P. 1265—1271.

25. **Muller-Putz G., Scherer R., Neuper C., Pfurtscheller G.** Steady-state somatosensory evoked potentials: suitable brain signals for brain-computer interfaces // IEEE Transactions on Rehabilitation Engineering. 2006. V. 14. P. 30—37.

26. **Shinkareva S., Mason R., Mitchell T., Just M.** Classifying cognitive states associated with reading single words and two-word sentences // 12th Conference on Human Brain Mapping. 2006.

27. **Cheng M., Gao X., Gao S., Xu D.** Design and Implementation of a Brain-Computer Interface With High Transfer Rates // IEEE Transactions on Biomedical Engineering. 2002. V. 49. N 10.

28. **Allison B., Pineda J.** ERPs evoked by different matrix sizes: Implications for a brain computer interface (BCI) system // IEEE Transactions on neural systems and rehabilitation engineering. 2003. V. 11. N 2. P. 110—113.

29. **Iversen I., Ghanayim N., Kubler A., Neumann N., Birbaumer N., Kaiser J.** A brain-computer interface tool to assess cognitive functions in completely paralyzed patients with amyotrophic lateral sclerosis // Clin Neurophysio. 2008. V. 119. N 10. P. 2214—2223.

30. **Wolpaw J. R.,McFarland D. J.** Control of a two-dimensional movement signal by a noninvasive brain-computer interface in humans // Proceedings of the National Academy of Sciences. 2004. V. 101. N 51. P. 17849—17854.

Список использованных интернет-ресурсов

И1.http://www.crunchgear.com/2008/03/05/university-of-bremens-brain-computer-interface-the-future-world-is-here/

И2.http://www.youtube.com/watch?v=hs5L6EmOB2M&hl=ru

И3.http://bci.tugraz.at/downloads.html

И4.http://www.youtube.com/watch?v=qCSSBEXBCbY

И5.http://www.youtube.com/watch?v=yhR076duc8M

И6.http://www.youtube.com/watch?v=BT4Sqg56XQ0

И7.http://www.youtube.com/watch?v=zozOjmpD5Cw

И8.http://www.youtube.com/watch?v=fQgC6H0qyf8

УДК 004.89

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

Документальные базы знаний на основе онтологий

Применение онтологий в интеллектуальных системах расширяет возможности и круг решаемых задач. Рассматривается использование ролевой кластеризации онтологии, порождающих отношений и контекстный анализ для выполнения семантического поиска информации, автоматического аннотирования документов, поддержки принятия решений, синтеза учебных программ и электронных образовательных ресурсов.

Ключевые слова: *документальные базы знаний, онтология, кластеризация, автоматическое аннотирование, синтез электронных образовательных ресурсов*

Введение

Создание текстовых документов требуется в разнообразных сферах человеческой деятельности. Наиболее формализованную структуру имеют формуляры, под которыми понимаются различного рода справки, договоры, постановления, приказы и т. п. Наименее формализованы художественные произведения. Между этими двумя полюсами находится обширный слой проектных и образовательных документов, в который входят различного рода пояснительные записки и другие спецификации проектов, учебники и учебные пособия. Для документов этого слоя, который будем называть слоем проектных описаний и образовательных ресурсов (ПООР), отсутствуют заранее заданные структуры, главное требование к ним — донесение нужной информации для определенного контингента пользователей в форме, наиболее удобной для правильного восприятия.

Помимо удобного восприятия к документам ПООР предъявляется требование малых затрат на их создание. Для удовлетворения этого требования разработаны технология интерактивных электронных технических руководств [1, 2] в сфере проектирования промышленной продукции и технологии SCORM [3] и разделяемых единиц контента (ТРЕК) [4] в сфере образования. В основу этих технологий положен принцип сборки документов из отдельных заранее разработанных частей, называемых разделяемыми единицами контента (РЕК), или модулями. Отличием ТРЕК от других подходов является использование предметных онтологий для управления сборкой документов из модулей. Благодаря онтологиям создаются гипертекстовые учебные пособия, соответствующие оптимальным обучающим траекториям [5].

Онтологический подход может быть использован не только для синтеза отдельных электронных учебных пособий (ЭУП). Родственной для синтеза ПООР задачей является сборка программ из модулей в объектно-ориентированных технологиях программирования [6].

Существует большое число автоматизированных инструментальных средств и обучающих систем [7—11], однако ни в одной из них, за исключением системы БиГОР [12], не применен онтологический подход и, как следствие, отсутствуют возможности оперативного создания индивидуализированных ЭУП.

Благодаря онтологиям в БиГОР обеспечиваются следующие возможности:

- малые затраты времени и средств на создание новых пособий из имеющихся модулей;
- формирование новых ЭУП путем навигации по семантической сети понятий предметной области;
- упорядочение отобранных модулей в составе пособия с соблюдением отношений предшествования "определяющее понятие — определяемое понятие";
- допустимость межмодульных гиперссылок;
- легкость корректировок содержания уже созданных ЭУП и, следовательно, поддержания ЭУП в актуальном состоянии;
- семантический поиск информации;
- синтез оптимальных траекторий обучения и поддерживающих их индивидуализированных учебных пособий и др.

Однако для существующей реализации ТРЕК характерен ряд ограничений и недостатков:

- цели и задания на формирование ЭУП непосредственно не связаны с компетенциями, подлежащими формированию у обучаемых в соответствии с федеральными государственными образовательными стандартами, недостаточно развиты средства формирования умений, поскольку в автоматически формируемые траектории обучения не входят задания и упражнения;
- объектами синтеза являются только ЭУП, не рассматриваются возможности применения ТРЕК для синтеза образовательных программ, создания технических описаний, пояснительных записок и т. п.;
- база учебных материалов состоит из модулей, специально разработанных для системы БиГОР, отсутствуют средства адаптации фрагментов научных публикаций при их включении в базу;
- в онтологиях, как правило, используются понятия, выражаемые одно- или двухсловными терминами, отношения таких понятий не рассматриваются как новые понятия, хотя именно они зачастую отображают семантику умений, проблем и задач.

В статье обсуждаются задачи, решаемые на основе онтологических баз знаний (БЗ), предлагается развитие ТРЕК в направлении снятия перечисленных ограничений. Излагается общий подход к синтезу отдельных документов слоя ПООР, образовательных программ и ЭУП, реализующих эти программы, для различных категорий обучаемых.

Концепты, отношения и интерпретаторы

Компетенции, как правило, выражаются фразами, состоящими из нескольких слов, являющихся значениями некоторых концептов. Будем называть такие фразы-словосочетания, представляющие собой отношения нескольких концептов, сложными концептами, а компоненты сложного концепта будем считать простыми концептами.

Пусть **ARB** — отношение **R** между простыми концептами **A** и **B**. Отношения могут быть порождающими и семантическими. Порождающее отношение **ARB** есть сложный концепт, представляемый в виде словосочетания. Например, "моделирование электронной схемы" — концепт, порожденный концептами "моделирование" и "электронная схема".

В порождающих отношениях **ARB**, соответствующих проблемам, задачам и компетенциям, в качестве **R** преимущественно фигурируют отношения "объект/свойство", "объект/действие", "действие/средство".

Семантическое отношение отражает семантическую связь. Семантические отношения могут быть симметричными (например, партнерство) и антисимметричными. Условимся считать антисимметричные отношения целое/часть, род/вид, причина/следствие, объект/свойство прямыми отношениями **R**, тогда часть/целое, вид/род, следствие/причина, свойство/объект — обратные отношения $\mathbf{R}^{-1}$. Концепт **A** в прямом отношении есть прообраз, **B** — образ. Семантику концепта

в отношении **R** будем называть ролью концепта, например, ролью концепта "узел" в отношении узел/деталь является "целое", а концепта "деталь" — "часть". Отметим, что отношение объект/свойство может быть одновременно как порождающим, так и семантическим.

Интерпретация — представление смысла концептов и отношений тем или иным способом. В ТРЕК интерпретация осуществляется с помощью кратких определений и/или более подробных текстовых пояснений, называемых модулями. Каждый модуль можно рассматривать как некоторый интерпретатор определенного концепта (или нескольких концептов). Введение интерпретации порождает представление онтологии в виде И/ИЛИ-графа, каждый интерпретатор — И-вершина, в общем случае концепт — ИЛИ-вершина, если имеется более одного варианта интерпретации (точнее, более одного варианта выражения интерпретации).

Интерпретатор можно представить как соответствие **KMY**, где **K** — множество определяющих (входных) концептов, **Y** — определяемые (выходные) концепты, **M** — интерпретатор. Другими словами, интерпретатор вводит новые отношения определяющий/определяемый, эти новые отношения совпадают с какими-либо семантическими отношениями. Если совпадение только с прямыми отношениями, то модуль (интерпретатор) **M** — прямой модуль, иначе модуль обратный $\mathbf{M}^{-1}$.

Формирование или обнаружение в текстах порождающих отношений можно осуществить на основе ролевой кластеризации предметной онтологии, а именно распределением концептов по кластерам "объект", "свойство", "действие (событие)", "средство". Тогда любые сочетания концептов из двух разных кластеров потенциально могут быть новыми концептами. Например, отношение компьютер/быстродействие (типа объект/свойство) порождает концепт "быстродействие компьютера". Примерами порождающих отношений типа объект/действие или действие/средство являются соответственно "проектирование редуктора" и "программа анализа". В порождающих отношениях роль "действие" обычно выполняют концепты, выражаемые отглагольными существительными. Внутри ролевых кластеров имеют место семантические отношения, как правило, это целое/часть, род/вид.

Онтология противоречива, если имеются интерпретаторы, придающие разный смысл одному и тому же концепту. В случаях омонимии одни и те же термины относятся к разным концептам, при этом для них должны различаться не только интерпретаторы, но и сами концепты. Один из способов различения омонимов — отождествление каждого понятия с его синсетом, в который для каждого омонима вводится отличающий его термин. Например, для понятий "класс" можно ввести синсеты "класс, тип" и "класс, аудитория".

Один и тот же концепт может иметь несколько непротиворечивых интерпретаций. Если интерпретации различаются (например, из-за транзитивности, или из-за разной степени подробности спецификации, или при замене **KMY** на $\mathbf{YM}^{-1}\mathbf{K}$), но не противоречат друг другу, то добавление или замену модуля не следует считать изменением онтологии. Изменилась форма представления интерпретации, но не сама онтология. Изменение онтологии — это изменение состава концептов или их отношений. Удобно непротиворечивую онтологию вместе с множеством модулей называть базой знаний (БЗ).

Отношения **M** и **R** близки, но в общем случае не совпадают, **R** объективно, **M** может иметь субъективную окраску, некоторые отношения из **R** могут отсутствовать в **M**.

Задачи, решаемые на основе онтологических БЗ

На рис. 1 представлена схема применения онтологической БЗ для решения ряда задач информационного поиска и синтеза ПООР.

Информационный поиск лежит в основе методов решения многих задач управления знаниями. Онтологии способствуют повышению эффективности поиска за счет предварительного выявления в документах терминов, соответствующих концептам онтологии, построения на их основе аннотаций документов и дальнейшего использования атрибутивного поиска вместо полнотекстового. В частности, если в число концептов включены понятия, характеризующие прецеденты принятия решений (ПР) в той или иной предметной области, то БЗ превращается в систему поддержки ПР в рамках метода CBR (Case Based Reasoning) [13].

Результаты атрибутивного поиска определяются составом слов, образующих поисковый образ документа (ПОД). В онтологической БЗ слова $x_i \in$ ПОД являются терминами концептов, поэтому точность поиска зависит от состава концептов в используемой онтологии. Поскольку проблемы, по которым ведется поиск, обычно выражаются словосочетаниями, соответствующими сложным концептам, такие концепты должны входить в онтологию. Как отмечено выше, сложные концепты являются порождающими отношениями **ARB**, в которых концептам **A** и **B** отведены определенные роли, поэтому для создания отношений **ARB** должна применяться ролевая кластеризация онтологии.

Для выделения в документах словосочетаний сложных концептов выполняется контекстный поиск, при котором концепты **A** и **B** образуют отношение **ARB**, только если они входят в разные

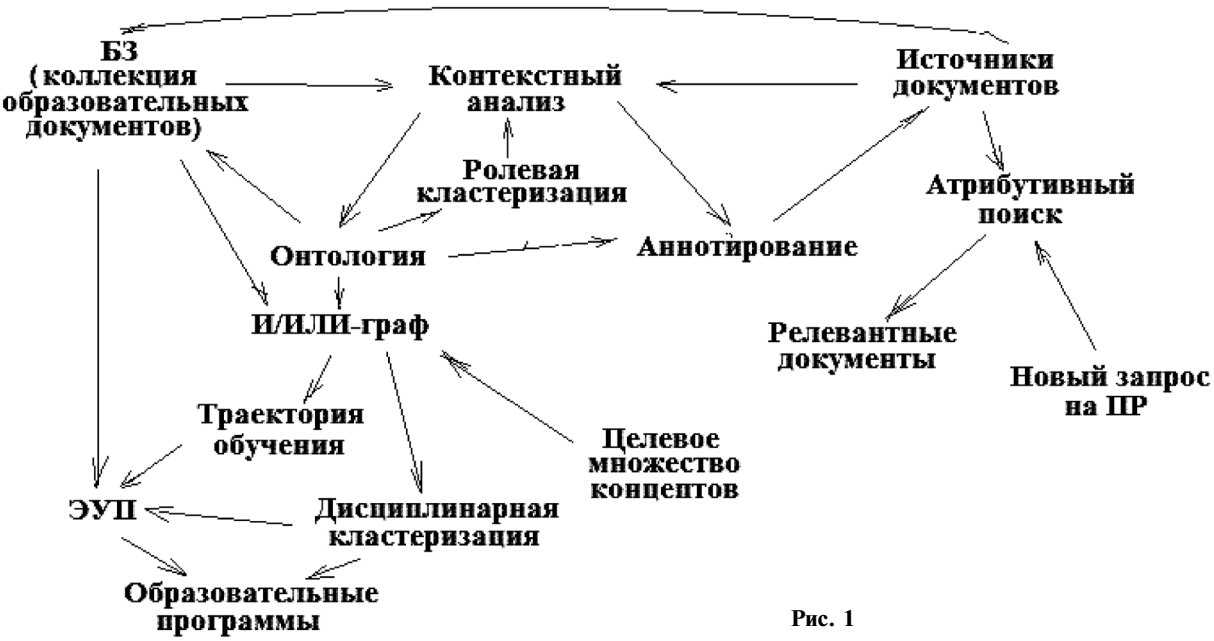


Рис. 1

кластеры и находятся в тексте документа в непосредственной близости друг от друга, причем расстояние между словами концептов **A** и **B** определяется числом разделяющих их слов. Аннотацию документа, отождествляемую с ПОД, составляют найденные отношения **ARB**, число повторений которых в тексте документа превышает заданный порог.

Таким образом, ролевая кластеризация онтологии и контекстный анализ позволяют пополнять онтологию сложными концептами, формировать аннотации (метаданные) документов словосочетаниями сложных концептов и осуществлять более точный поиск релевантных документов.

Кроме ролевой, выполняется тематическая (дисциплинарная) кластеризация концептов. Образовавшийся i-й кластер можно рассматривать как эталонное подмножество $\mathbf{E}_i$ концептов i-й дисциплины (раздела) рассматриваемой предметной области. Выделение в j-м документе терминов, характеризующих концепты из подмножества E_i, образует вектор $\mathbf{F}_{ij} = (f_{ij1}, ..., f_{ijn})$, где f_{ijk} — число повторений терминов k-го концепта (синсета) в j-м документе. Сопоставление норм векторов $\mathbf{F}_{ij}$ для разных i позволяет отнести j-й документ к определенному разделу предметной области, т. е. выполнить классификацию документов. Аналогичным образом выполняется распределение (кластеризация) документов заданной коллекции по разделам предметной области. Отметим, что мощность множества $\mathbf{E}_p \cap \mathbf{E}_q$ характеризует степень семантической близости p-го и q-го разделов предметной области.

В образовательной сфере онтологии целесообразно применять для решения задач синтеза образовательных программ и поддерживающих их ЭУП. Синтез осуществляется на основе онтологической БЗ, т. е. предполагается, что задана или предварительно разработана онтология для рассматриваемой отрасли знаний вместе с необходимыми интерпретаторами.

Один из способов представления онтологии — семантическая сеть (СС) с вершинами, соответствующими концептам, и связями, соответствующими отношениям. Другой способ — И/ИЛИ-граф, в котором дополнительно к вершинам ИЛИ, отображающим концепты, появляются вершины И, отображающие модули. Дуги, ведущие от ИЛИ-вершин к И-вершинам, характеризуют отношения "концепт используется в модуле", остальные дуги — отношения "концепт определен в модуле".

Образовательная программа состоит из учебного плана и учебных программ дисциплин. Синтез учебного плана начинается с выделения в онтологии подмножества $\mathbf{T}_{\text{цел}}$ понятий, изучаемых в конкретной образовательной программе. Обычно в состав $\mathbf{T}_{\text{цел}}$ входят концепты, называемые целевыми, определяющие заданные компетенции обучаемых и подлежащие обязательному изучению.

Далее синтезируется траектория обучения, представляющая собой множества $\mathbf{T}_{\text{тр}}$ концептов и $\mathbf{M}_{\text{тр}}$ модулей, поясняющих эти концепты, причем $\mathbf{T}_{\text{цел}} \subset \mathbf{T}_{\text{тр}}$.

В ТРЕК формирование траектории обучения возможно в ручном или автоматическом режиме. Ручное формирование основано на навигации, осуществляемой пользователем по СС, начиная с ключевых слов, выражающих целевые концепты. В процессе навигации инструментальная система предоставляет пользователю, во-первых, списки

понятий, с которыми связано текущее понятие, т. е. пользователь видит семантическую окрестность текущего понятия, во-вторых, список тех модулей, в которых определено (описано и пояснено) текущее понятие. Из списка модулей пользователь выбирает те РЕК, которые он считает нужным включить в траекторию, а списки понятий используются для продолжения навигации. В процессе навигации разработчиком образовательной программы отбираются модули, содержащие как необходимый теоретический материал, интерпретирующий текущие понятия, так и практические задачи и подходы к их решению.

Автоматическое формирование траектории заключается в выборе и упорядочении подмножества модулей $u_j \subset \mathbf{U}$, где $\mathbf{U}$ — множество возможных решений задачи поиска $\mathbf{M}_{\text{тр}}$. Введем булевы переменные x_p и y_q такие, что $x_p = 1$, если концепт k_p входит в траекторию, и $y_q = 1$, если модуль m_q включен в траекторию, иначе $x_p = 0$ и $y_q = 0$. В [5] показано, что множество решений задачи поиска $\mathbf{M}_{\text{тр}}$ соответствует одному из решений логического уравнения

$$x_{\text{цел}} = \bigvee_{q \in \mathbf{Q}_{\text{цел}}} y_q \bigwedge_{r \in \mathbf{R}_q} x_r = 1, \qquad (1)$$

где V и & — знаки логических сложения и умножения; $x_{\text{цел}}$ — переменная целевого концепта; x_r и y_q — переменные r-го концепта и q-го модуля решения соответственно; $\mathbf{Q}_{\text{цел}}$ — множество номеров модулей, в которых определен целевой концепт $k_{\text{цел}}$; $\mathbf{R}_q$ — множество номеров концептов, входных для модуля m_q. Если целевых концептов несколько, то $x_{\text{цел}}$ в формуле (1) — конъюнкция переменных целевых концептов. Раскрытие рекурсии (1) приводит к дизъюнктивной нормальной форме, в которой множество элементарных конъюнкций отождествляется с $\mathbf{U}$. В этом множестве нужно выбрать конъюнкт с экстремальным значением целевой функции $F(\mathbf{U})$, в качестве которой можно использовать число модулей, суммарный объем модулей траектории, сложность освоения материала модулей и т. п.

Следующей операцией синтеза образовательной программы является тематическая кластеризация концептов и модулей траектории. Каждый кластер представляет собой единицу учебного плана, для которой должны формироваться ЭУП, выделяться ресурсы и место в расписании занятий.

Предварительно осуществляется поиск и устранение контуров в СС множества $\mathbf{T}_{\text{тр}}$. Следует отметить, что вероятность появления контуров в СС существенно снижается при использовании только прямых или только обратных интерпретаторов. Появившиеся контуры устраняются изменением интерпретаторов.

Тематическая кластеризация может быть выполнена предметными специалистами или автоматически с помощью известных формальных методов [14]. При этом показателями связности концептов являются степени вершин в СС. Полученное распределение концептов по тематическим кластерам может быть скорректировано в дальнейшем.

Следующей процедурой синтеза образовательных программ является установление последовательности изучения дисциплин. Распределение дисциплин по семестрам (или другим временным периодам) является оптимизационной задачей. Если известны число дисциплин и их трудоемкость, то возможна следующая постановка задачи:

$$\min \sum_{k \in \mathbf{K}} T_k \qquad (2)$$

и при ограничениях на последовательность дисциплин и

$$\sum_{i \in \mathbf{I}_j} T_i \leqslant d_j,$$

где T_k — трудоемкость k-й дисциплины; $\mathbf{K}$ — множество индексов дисциплин, составляющих наидлиннейший путь в графе учебного плана; $\mathbf{I}_j$ — множество индексов дисциплин j-го семестра; d_j — максимально допустимая суммарная трудоемкость дисциплин в j-м семестре. Ограничения на последовательность дисциплин определяются их взаимосвязями. Так, если связи между кластерами односторонние, например, направлены от кластера **X** к кластеру **Y**, то дисциплина **Y** должна изучаться после дисциплины **X**.

Коррекция кластеров может потребоваться, во-первых, при наличии дуг, связывающих вершины двух разных кластеров СС в обоих направлениях (рис. 2), во-вторых, при необходимости

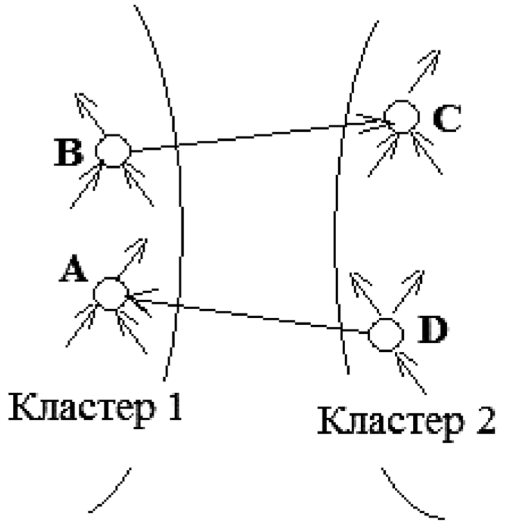


Рис. 2

параллельного расположения в учебном плане дисциплин с односторонними связями, что способствует минимизации (2). Возможно перемещение вершин между кластерами, однако оно, устраняя одни межкластерные связи, порождает другие. Альтернативой перемещению вершин является либо параллельное изучение взаимосвязанных дисциплин с их дроблением на более мелкие единицы ради учета имеющихся взаимосвязей, либо применение интерпретации концептов (в случае рис. 2 концептов **A** и **C**) с игнорированием межкластерных связей.

Синтез ЭУП возможен применительно как к заданному тематической кластеризацией составу понятий, так и к пособиям с составом концептов, выбираемым в соответствии с индивидуальными запросами. Созданное пособие представляет собой упорядоченное подмножество модулей, отобранных в процессе навигации по И/ИЛИ-графу онтологии.

Таким образом, для создания ЭУП, направленных не только на освоение обучаемыми знаний, соответствующих заданным компетенциям, но и на формирование у обучаемых нужных умений в онтологии должны присутствовать концепты, описывающие относящиеся к предметной области компетенции, проблемы и задачи. Как правило, формулировки компетенций и задач являются многословными, поэтому необходимо использовать средства пополнения онтологии сложными концептами.

Синтез ЭУП на основе онтологических БЗ имеет ряд преимуществ, охарактеризованных выше. Эти преимущества сохраняются и для других ПООР, в частности, для пояснительных записок проектов и технических описаний промышленных изделий.

Заключение

Основой создания удобных для восприятия пользователем и легко сопровождаемых документов могут стать технологии и системы, реализующие онтологический подход к поиску и обработке информации. Возможности онтологического подхода проиллюстрированы в статье на примере задач формирования образовательных программ, синтеза электронных учебников и пособий, поиска в коллекциях документов прецедентов для принятия решений.

Работа выполнена при финансовой поддержке РФФИ (код проекта 10-07-00401-а).

Список литературы

1. **Manuals,** interactive electronic technical: general content, style, format and user-interaction requirements. Military specification, MIL-M-87268.

2. **The international** technical publication specification. URL: http://www.aecma.org

3. **SCORM.** Shareable Content Object Reference Model. 2d Edition. — Advanced Distributed Learning, 2004.

4. **Норенков И. П.** Технологии разделяемых единиц контента для создания и сопровождения информационно-образовательных сред // Информационные технологии. 2003. № 8.

5. **Норенков И. П.** Генетические алгоритмы поиска решений в онтологических базах знаний // Информационные технологии. 2010. № 9.

6. **Сидоров С. Б., Приблудова Е. Н.** Модульное программирование. Нижний Новгород: НГГУ, 2010, 74 с.

7. **Вуль В. А.** Электронные издания. СПб.: Изд-во "Петербургский институт печати", 2001. 308 с.

8. **Башмаков А. И., Башмаков И. А.** Разработка компьютерных учебников и обучающих систем. М.: Филинъ, 2003. 616 с.

9. **Информатизация** образования: направления, средства, технологии / Под ред. С. И. Маслова. М.: Изд-во МЭИ, 2004. 868 с.

10. **Соловов А. В.** Электронное обучение: проблематика, дидактика, технология. Самара: Новая техника, 2006. 464 с.

11. **Шереметов Л. Б., Усков В. Л.** Виртуальные образовательные среды // Информационные технологии (приложение), 2002. № 5. 24 с.

12. **Норенков И. П., Уваров М. Ю.** База и генератор образовательных ресурсов // Информационные технологии. 2005. № 9. С. 60—65.

13. **Bandyopadhyay S., Maulik U., Holder L. B., Cook D. J.** Advanced Methods for Knowledge Discovery from Complex Data. URL: http://hall.org.ua/halls/wizzard/books/A1/advanced-methods-for-knowledge-discovery-from-complex-data.9781852339890.28122.pdf

14. **Технологии** анализа данных: Data Mining, Visual Mining, OLAP / А. А. Барсегян, М. С. Куприянов, В. В. Степаненко, И. И. Холод. СПб.: БХВ-Петербург, 2007. 384 с.

УДК 621.398-506.222

В. О. Толчеев, д-р техн. наук, доц.
Московский энергетический институт
(технический университет)
e-mail: tolcheevvo@mail.ru

Анализ проблемы и разработка процедуры выявления нечетких дубликатов научных статей по библиографическим описаниям

Рассматриваются методы обнаружения нечетких (неполных) дубликатов научных публикаций по библиографическим описаниям, проанализирована специфика решаемой задачи и разработана процедура выявления нечетких дубликатов на основе использования комитета коэффициентов ассоциативности. Процедура протестирована на специально сформированных выборках.

Ключевые слова: *обработка и анализ библиографической текстовой информации, обнаружение нечетких (неполных) дубликатов, связанные публикации, коэффициенты ассоциативности*

Введение

В настоящее время в области информационного поиска и обработки документальной информации активно развивается направление исследований, связанное с выявлением дубликатов и нечетких дубликатов. К *дубликатам* (*неуникальным публикациям*) принято относить документы с идентичным содержанием. *Нечеткими* (*неполными*) *дубликатами* или *почти дубликатами* считаются документы, в содержательную часть которых внесены незначительные изменения.

Интерес к данному направлению исследований объясняется потребностями практики. Обнаружение и удаление дубликатов позволяет решать следующие актуальные проблемы:

- устранение одинаковых документов, выдаваемых информационно-поисковой системой (ИПС) на запрос пользователя;
- исключение повторяющейся информации при обработке документальных массивов, полученных из разнородных источников;
- определение похожих по содержанию текстов, в частности выявление плагиата.

Фактически все известные исследования проводятся именно по этим направлениям [1—6]. В значительно меньшей степени в специализированной литературе представлены работы, рассматривающие более "нетрадиционные" вопросы, которые возникают в данной области. В то же время постановка и решение новых задач способны существенным образом расширить практическое применение процедур обнаружения нечетких дубликатов.

Традиционно научная общественность уделяет большое внимание проблеме плагиата. Известны случаи некорректного использования чужих текстов при подготовке публикаций, рефератов, дипломных, диссертационных работ, создании и сопровождении сайтов. Для выявления идентичных документальных фрагментов и Web-страниц разработан ряд эффективных программно-алгоритмических средств, в частности система "Антиплагиат", предназначенная для проверки текстовых документов на наличие заимствований из общедоступных сетевых источников [7]. Вместе с тем в настоящее время представляется актуальным не только анализ документальных массивов на наличие совпадающих фрагментов, но и выявление среди журнальных публикаций дубликатов и нечетких дубликатов (статей-клонов), в которых авторы под разными названиями с небольшими редакторскими правками публикуют практически одинаковые научные результаты.

На наш взгляд, существуют две основные причины появления таких статей-клонов. Во-первых, ужесточение требований ВАК к числу журнальных публикаций, необходимых для защиты докторских диссертаций. Во-вторых, стремление научных сотрудников увеличить показатель результативности научной деятельности (ПРНД) за счет статей в престижных журналах (высокий ПРНД позволяет, в конечном итоге, увеличить заработную плату, занять более высокие должности и т. п.). Обе причины вызывают появление похожих работ. Причем в ряде случаев авторы просто "переиздают" свои более ранние публикации, практически не внося в них содержательные изменения.

В сложившейся ситуации нельзя исключить появление неуникальных статей в различных журналах, специализирующихся в одной и той же научной области. При создании вариаций на основе базовой работы автор обычно использует перестановку слов и предложений, изменяет название, употребляет синонимы, поэтому простая проверка публикаций на наличие плагиата малоэффективна и требуется разработка новых специализированных алгоритмов. Чаще всего для проведения сравнительного анализа доступны только библиографические описания (БО) статей, которые обычно публикуются на сайтах журналов. БО состоит из фамилий авторов, названия, аннотации, ключевых слов, места и времени издания и другой вспомогательной информации.

В связи с изложенным выше представляется актуальным и своевременным создание технологии, позволяющей выявлять нечеткие дубликаты среди научных публикаций на основе анализа БО. Уточним, что в контексте наших исследований

под неуникальными работами понимаются журнальные статьи, которые имеют совпадающие или очень близкие названия и аннотации. Предполагается, что в этом случае в них излагается практически один и тот же материал, т. е. отсутствуют значимые отличия в представляемых научно-практических результатах.

Рассмотрим основные характерные особенности решаемой задачи.

Специфика решаемой задачи

1. Для разрабатываемой технологии принципиальным моментом является обнаружение неуникальных публикаций по БО. Это имеет свои очевидные недостатки и преимущества. К недостаткам, прежде всего, относятся "сильная усеченность" и невысокая информативность БО по сравнению с полнотекстовыми версиями статей (отсутствует возможность анализа всего публикуемого материала). Более того, часто БО состоят всего из нескольких предложений и требуется выявить нечеткие дубликаты на основе сравнения очень коротких текстов. Вместе с тем применение БО имеет свои несомненные преимущества: они находятся в свободном доступе на сайтах изданий, хорошо структурированы и имеют смысловые маркеры, которые можно использовать для обнаружения потенциальных дубликатов (автор, название, аннотация, место и время издания и т. п.).

2. В научной работе типичной является ситуация, когда специалист после опубликования результатов, например разработки оригинальной процедуры, продолжает проводить ее исследование и апробацию. В связи с этим появляются тематически близкие, *"связанные" ,публикации*, посвященные анализу различных характеристик разработанной процедуры. Такие публикации, несомненно, являются уникальными и отражают этапы проведения НИОКР. Как представляется, большинство авторских статей, посвященных одной тематике, являются именно "связанными" публикациями. Поэтому для минимизации ложных срабатываний автоматизированная процедура выявления нечетких дубликатов должна обладать высокой чувствительностью даже к незначительным содержательным отличиям, имеющимся в БО, и не допускать определения "связанных" публикаций в качестве нечетких дубликатов.

3. Важно отметить, что, как и для ряда других терминов из области информатики (например релевантности), понятие нечеткого дубликата весьма субъективно и чаще всего определяется на основе экспериментально установленных пороговых значений схожести документов. Высокий уровень субъективизма существенно затрудняет программное решение данной задачи и требует привлечения специалистов-предметников, которые способны вынести окончательное суждение об уникальности документа. Это суждение может быть сформировано как на основе просмотра БО статей-кандидатов в дубликаты, так и путем сопоставления полнотекстовых версий (в случае, если авторы (или издательства) готовы их предоставить).

4. В настоящее время выпускается огромное число научных журналов. Структура представления текстовых данных на сайтах изданий существенным образом отличается и неунифицирована. Это превращает загрузку БО в чрезвычайно трудоемкую и дорогостоящую процедуру. Более того, в условиях ограниченного доступа к ряду информационных ресурсов создание универсальных программно-алгоритмических средств малоэффективно.

5. Существенным моментом при выявлении нечетких дубликатов является значимость и "чувствительность" затрагиваемой проблемы для автора. Очевидно, что неверное суждение о степени уникальности публикации может нанести существенный удар по репутации научного сотрудника. При этом даже правильное решение может оказаться темой для длительных и малопродуктивных дискуссий о несовершенстве и невысокой точности созданных программно-алгоритмических средств, неоднозначности экспериментально и экспертно выявленных закономерностей.

6. В данной работе указывается на возможность появления нечетких дубликатов среди научных статей, приводятся возможные мотивы таких действий. Вместе с тем, на наш взгляд, можно с уверенностью утверждать, что появление неуникальных публикаций относится к исключительным, совершенно нетипичным случаям. Это означает, что требуется решить слабо формализованную задачу выявления чрезвычайно редких событий в большом документальном массиве.

Учитывая указанные выше специфические особенности решаемой задачи, конкретизируем область использования процедуры выявления нечетких дубликатов и необходимые условия для ее эффективного применения.

Во-первых, предполагается, что пользователями программно-алгоритмических средств будут, прежде всего, редколлегии журналов, которые получат возможность наряду с экспертным анализом проводить автоматизированную оценку степени уникальности публикаций. При этом окончательное решение будут принимать эксперты, а программный комплекс будет лишь сигнализировать о том, что та или иная публикация требует более тщательной проверки. Учитывая загруженность редколлегий и рецензентов, такой "программный ассистент" может принести существенную практическую пользу. Потенциальными пользователями разработанных программно-алгоритмических средств являются также диссертационные советы, ученые советы институтов РАН, научные коллективы кафедр, лабораторий и т. п.

Во-вторых, важным условием эффективного применения предлагаемой процедуры является доступность библиографической информации. Необходимо, чтобы все журналы выполняли требование ВАК о размещении в открытом доступе БО, а в перспективе предоставляли открытый доступ ко всем полнотекстовым статьям через два года после их издания (в настоящее время реальное положение дел существенным образом отличается от требований нормативных документов ВАК).

В-третьих, для снижения ресурсозатратности процедуры и уменьшения числа работ, отбираемых в качестве потенциальных дубликатов, целесообразно применять специальные маркеры, содержащиеся в БО, — фамилии авторов, время и место публикации, совпадение терминов в названиях. При определении неуникальных статей необходимо также использовать структурные особенности БО, позволяющие учитывать местоположение терминов (в названии или аннотации) [8]. Именно для публикаций, являющихся потенциальными кандидатами в дубликаты, имеет смысл проводить проверку в целях установления степени их уникальности.

Краткий обзор имеющихся подходов

В настоящее время существует ряд достаточно полных и доступных в Интернете обзоров, в которых дается описание алгоритмов выявления идентичных документов, делается их сравнительный анализ на основе экспериментальных исследований [1, 2, 9]. Поэтому представляется нецелесообразным проводить аналогичный обзор в рамках данной статьи.

Ранее отмечалось, что большинство имеющихся публикаций посвящено решению проблемы обнаружения дубликатов *Web*-страниц и полнотекстовых статей для обеспечения более высокого качества работы ИПС или очистки данных при их размещении в хранилищах [1, 2, 10, 11]. Изучение специализированной литературы не позволило выявить работы, предлагающие апробированные и теоретически (или экспериментально) обоснованные алгоритмы выявления неидентичных публикаций по их библиографическим описаниям. Возможно, это связано с тем, что такая постановка задачи достаточно нова и интерес к проблеме обусловлен недавно появившимися факторами (ужесточение требований ВАК и введение ПРНД). Как представляется, наиболее близкими к проблеме, рассматриваемой в данной работе, являются публикации о способах выявления неуникальных статей из разнородных библиографических баз данных по математике и поиске дубликатов в текстах проектной документации [3, 4].

Для обнаружения и исключения нечетких дубликатов используются два типа методов: синтаксические методы (анализ последовательностей, состоящих из символов, слов или предложений) и лексические методы (анализ информативных терминов). На практике чаще всего применяют метод шинглов, би-кластеризацию, фонетическое кодирование, а также расчет различных мер близости (расстояние редактирования Левенштейна, дистанция Джаро—Винклера, коэффициент ассоциативности Джаккарда, косинусоидальная мера) [2—9].

Как справедливо указывается в литературе, выбор конкретного метода в наибольшей степени зависит от цели разработки, особенностей предметной области, исходной информации и имеющихся ограничений. Применительно к проблеме, рассматриваемой в данной работе при определении наиболее приемлемого алгоритма, необходимо учитывать его трудоемкость, эффективность обработки коротких документов, чувствительность к изменениям в публикациях (модификации названий статей, перестановка, вставка и удаление терминов (словосочетаний, предложений) в аннотации). С точки зрения сформулированных требований при разработке автоматизированной процедуры выявления нечетких дубликатов особый интерес представляют вопросы изучения и применения умеренно трудоемких и достаточно универсальных *коэффициентов ассоциативности* (КА) [12].

В публикациях по обнаружению и исключению выбросов достаточно часто и успешно используется один из наиболее известных коэффициентов ассоциативности — коэффициент Джаккарда. Вместе с тем в специализированной литературе по кластеризации и анализу таблиц сопряженности обосновывается применение других КА, которые в ряде случаев могут оказаться более чувствительными к имеющимся различиям в текстах небольшого размера. В связи с этим представляется целесообразным провести экспериментальные исследования известных КА на специально сформированных выборках библиографических документов и оценить возможность их использования для выявления нечетких дубликатов.

Разработка процедуры выявления нечетких дубликатов

Формализуем задачу обработки и анализа библиографической текстовой информации. Далее предполагается, что документы представляются в виде векторной модели [13]:

$$\mathbf{X}_j = \begin{bmatrix} x_j^{(1)} \\ \vdots \\ x_j^{(i)} \\ \vdots \\ x_j^{(M)} \end{bmatrix}, \qquad (1)$$

где $x_j^{(i)}$ — вес термина i в документе j ($j = 1, ..., N$; N — число документов в выборке; $i = 1, ..., M$; M — число признаков).

В данной работе для определения весов терминов в библиографическом документе используется логическое взвешивание [14]. Логическое взвешивание заключается в том, что весу слова i присваивается значение 1, если оно не менее одного раза встречается в j-м документе, и значение 0 — в противном случае:

$$x_j^{(i)} = \begin{cases} 1, f_{ij} > 0 \\ 0, f_{ij} = 0. \end{cases} \cdot \quad (2)$$

Здесь f_{ij} — частота встречаемости слова i в документе j.

Применение более сложных способов взвешивания (например, *tf—idf*, *tfc*-взвешивание) нецелесообразно, так как проверка на "уникальность" научной статьи проводится не среди большой выборки текстовых документов, а между несколькими публикациями, которые были отобраны из исследуемого массива в качестве потенциальных дубликатов.

В статье используются коэффициенты ассоциативности, которые могут быть получены из общей формулы:

$$J = \frac{\alpha a}{\alpha a + \beta b + \gamma c}, \quad (3)$$

где a — число совпавших терминов в двух документах $\mathbf{X}_j$ и $\mathbf{X}_l$; b — число терминов, имеющихся в $\mathbf{X}_j$ и отсутствующих в $\mathbf{X}_l$; c — число терминов, имеющихся в $\mathbf{X}_l$ и отсутствующих в $\mathbf{X}_j$; α, β, γ — константы. На основе формулы (3) можно рассчитать коэффициент Джаккарда (J), коэффициент *Dice* и коэффициент Соукала—Сниса (SS) [12, 15]:

$$J = \frac{a}{a + b + c}; \quad (4)$$

$$Dice = \frac{2a}{2a + b + c}; \quad (5)$$

$$SS = \frac{a}{a + 2(b + c)}. \quad (6)$$

Значения коэффициентов, получаемых из формулы (3), лежат в интервале [0, 1], что облегчает интерпретацию результатов. Так, значение 1 свидетельствует о полном совпадении всех слов в двух текстах ($b = c = 0$), а значение 0 означает отсутствие общих терминов ($a = 0$).

Для экспериментального исследования КА было сформировано две выборки. Первая выборка состояла из 23 БО публикаций автора данной статьи в журналах и на конференциях. Вторая выборка содержала 39 БО докладов, которые были сделаны сотрудниками, аспирантами и студентами Института автоматики и вычислительной техники МЭИ на научно-технических семинарах "Современные технологии в задачах управления, автоматики и обработки информации" в 2006—2009 годах.

Проведенные исследования позволяют сделать следующие выводы:

1. Анализируемые коэффициенты ассоциативности достаточно уверенно идентифицируют уникальные документы. При этом значения всех КА принимают малые значения (КА < 0,5).

2. Значения коэффициентов ассоциативности в значительной степени зависят от способа расчета, обладают различной чувствительностью к размеру текстов и числу совпадающих-несовпадающих слов. Для получения более точного решения, учитывающего различные особенности анализируемых документов, представляется целесообразным провести объединение трех КА в общий комитет (комитет коэффициентов ассоциативности), принимающий решение на основе простого голосования [16, 17].

3. На сформированных выборках не было получено значений КА, близких к единице (КА $\geqslant$ 0,8). В то же время были выявлены тексты, для которых коэффициенты ассоциативности попадают в диапазон [0,7; 0,8], и поэтому требуется их более внимательное изучение. Основными характерными чертами таких документов является совпадение аннотаций (при различных названиях) и близкие формулировки названий при похожих аннотациях. Проведенное более тщательное экспертное изучение полного текста статей, которым соответствуют высокие значения коэффициентов ассоциативности, позволило отнести их к категории "связанных" публикаций.

Экспериментальные исследования показали, что использование комитета КА позволяет достаточно хорошо формализовать рассматриваемую задачу, обеспечивая хорошо интерпретируемые результаты разделения выборок на уникальные статьи (КА < 0,5), "связанные" публикации (КА $\in$ [0,5; 0,8]) и нечеткие дубликаты (КА $\geqslant$ 0,8).

На основе проведенных экспериментов была разработана следующая автоматизированная процедура выявления нечетких дубликатов.

Автоматизированная процедура выявления нечетких дубликатов

Шаг 1. Приведение БО, полученных из разнородных источников (сайтов журналов), к единому формату и удаление стоп-слов. Выявление статей, которые принадлежат одному и тому же автору (группе авторов) и опубликованы в течение 3 лет в различных изданиях.

Шаг 2. Расчет для публикаций, выявленных на предыдущем шаге, трех КА (см. формулы (4)—(6)) и их сравнение с пороговым значением (для избежания ложных срабатываний порог установлен равным 0,8).

Шаг 3. Публикации, у которых как минимум два коэффициента ассоциативности, имеют значения, превосходящие порог, считаются *потенциальными нечеткими дубликатами*.

Шаг 4. Для потенциальных нечетких дубликатов, обнаруженных на предыдущем шаге, проводится экспертный анализ на предмет установления степени их уникальности и формируется окончательное решение о наличии в исходном массиве нечетких дубликатов.

Направления дальнейших исследований

Как представляется, совершенствованию данной автоматизированной процедуры выявления нечетких дубликатов будет способствовать реализация следующих предложений.

1. Прежде всего, необходимо провести дополнительные экспериментальные исследования, в частности, сформировать три специальные выборки. Первая выборка должна состоять из статей, отобранных экспертным образом на основе просмотра библиографического списка авторефератов докторских диссертаций, которые размещены на сайте ВАК (критерий отбора — наличие нескольких публикаций автора в различных журналах в течение 3 лет перед защитой). Вторая выборка должна включать статьи из общедоступного информационного ресурса "Научная электронная библиотека" (eLIBRARY.RU), в котором имеется свободный доступ к достаточно большому числу библиографических описаний научных статей. В третьей выборке должны содержаться "смоделированные" документы, полученные из специально выбранных базовых публикаций путем вставки-удаления терминов (словосочетаний, предложений) и заменой слов синонимичными значениями (например, метод—процедура—алгоритм, разработка—создание—синтез и т. п.).

2. Включить в разработанную процедуру алгоритм, осуществляющий определение устойчивых словосочетаний в анализируемых документах и оценку степени совпадения выявленных словосочетаний.

3. Расширить сформированный комитет коэффициентов ассоциативности за счет разработки новых КА или модификации известных коэффициентов.

4. Изучить возможность составления и использования специального словаря стоп-слов, включающего часто встречающиеся глаголы (проанализировано, рассмотрено, реализовано и т. п.), малоинформативные существительные (статья, работа и т. п.), набор синонимических конструкций (метод—процедура—алгоритм—способ—подход; сравнение—сопоставление; разработка—синтез—создание и т. п.).

На наш взгляд, развитие разработанной автоматизированной процедуры на основе этих предложений будет способствовать повышению ее чувствительности к содержательным отличиям в библиографических описаниях научных публикаций и улучшит качество выявления нечетких дубликатов.

Список литературы

1. **Зеленков Ю. Г., Сегалович И. В.** Сравнительный анализ методов определения нечетких дубликатов для Web-документов // Труды 9 Всероссийской научной конференции "Электронные библиотеки: перспективные методы и технологии, электронные коллекции". Переславль-Залесский: Изд-во ИПС РАН, 2007. С. 166—174.

2. **Цыганов Н. Л., Циканин М. А.** Исследование методов поиска дубликатов веб-документов с учетом запроса пользователя // Сборник "Интернет-математика". Екатеринбург: Изд-во Уральского университета, 2007. С. 211—222.

3. **Рубцов Д. Н., Барахнин В. Б.** О возможности борьбы с дубликатами при запросах к разнородным библиографическим источникам // Труды 11-й Всероссийской научной конференции "Электронные библиотеки: перспективные методы и технологии, электронные коллекции". Петрозаводск, 2009.

4. **Игнатов Д. И., Кузнецов С. О., Лопатникова В. Б., Селицкий И. А.** Разработка и апробация системы поиска дубликатов в текстах проектной документации // Бизнес-информатика. 2008. № 4. С. 21—28.

5. **Цыганов Н. Л.** Методика поиска дублирующих записей с помощью алгоритма нечеткого сопоставления строк // Труды научной сессии МИФИ. Т. 2. М.: Изд-во МИФИ, 2007. С. 159—160.

6. **Косинов Д. И.** Использование статистической информации при выявлении схожих документов // Сборник "Интернет-математика". Екатеринбург: Изд-во Уральского университета, 2007. С. 84—90.

7. **Антиплагиат** / URL: http://www.antiplagiat.ru

8. **Некрасов И. В., Толчеев В. О.** Построение модели представления библиографического документа // Информационные технологии. 2005. № 11. С. 57—63.

9. **Graham A.** String Search. Technical Report TR-92-gas-01. School of Electronic Engineering Science / University College of North Wales. October 1992. URL: http://infoscope.ws/string_search/Stephen-92/index.html

10. **Baeza-Yates R. A.** Searching Subsequences // Theoretical Computer Science. 1991. 2(78). P. 363—376.

11. **Navarro G. A.** Guided Tour to Approximate String Matching // ACM Computing Surveys. 2001. 1(33). P. 31—88.

12. **Ким Дж., Мьюллер Ч., Клекка У.** и др. Факторный, дискриминантный и кластерный анализ. М.: Финансы и статистика. 1989. 215 с.

13. **Sebastiani F.** Machine Learning in Automated Text Categorization // ACM Computing Survey. 2002. 34(1). P. 1—47.

14. **Salton G., Buckley C., Allan J.** Automatic structuring of text files // Electronic Publishing. 1992. Vol. 5, N 1. P. 1—17.

15. **Елисеева И. И., Рукавишников В. О.** Группировка, корреляция и распознавание образов. М.: Статистика. 1977. 143 с.

16. **Растригин Л. А., Эренштейн Р. Х.** Метод коллективного распознавания. М.: Энергоиздат, 1981. 79 с.

17. **Толчеев В. О.** Синтез коллективов решающих правил для проведения классификации текстовых документов // Информационные технологии. 2007. № 10. С. 32—38.

УДК 004.75.021

Б. Амиршахи, аспирант,
Московский государственный университет
путей сообщения (МИИТ),
e-mail: bita.amirshahi@gmail.com

Кластеризация GRID-ресурсов для оптимизации информационного обмена при совместной обработке результатов распределенных вычислений

Рассматривается проблема объединения результатов распределенных вычислений для совместной обработки головным процессором. Для реализации параллельно-последовательной древовидной структуры обмена предлагается новый параллельный алгоритм кластеризации GRID-ресурсов, назначенных для решения задачи. Алгоритм обеспечивает значительно более высокое быстродействие по сравнению с ранее известными.

Ключевые слова: *GRID-вычисления, кластер, параллельные алгоритмы кластеризации, иерархическая кластеризация, минимальное покрывающее дерево, сложность*

Введение

Обязательным элементом схемы распределенных вычислений, характерной для GRID-технологий, является выполнение следующих действий:

1) назначение вариантов счета на процессоры вычислительного ресурса, выделенного для совместного решения задачи;

2) параллельные вычисления;

3) объединение результатов вычислений на головном (центральном) процессоре для их совместного анализа и обработки.

В основе вычислительного ресурса для GRID-вычислений лежат сети — как локальные, так и глобальные. Поэтому сетевые технологии информационного обмена являются определяющими. Здесь и возникла проблема: если первый пункт схемы допускает *одновременную* рассылку данных многим процессорам, то третий пункт, в случае непредпринимаемых мер, грозит *последовательным* обменом каждого периферийного процессора с головным. Учитывая время организации и выполнения единичного обмена, можно прийти к выводу о нецелесообразности распределенной обработки.

Перед такой проблемой оказался автор, пытаясь ускорить решение системы 1 000 000 линейных уравнений методом Крамера на 105 процессорах в сети Интернет.

Проблема не нова. Выходом из создавшегося положения является иерархическая кластеризация вычислительного ресурса с целью организации "древесной" параллельно-последовательной структуры сбора данных. Информация объединяется внутри кластеров нижнего уровня, затем между этими кластерами на более высоком уровне и т. д. до достижения головного процессора.

Известно, что суперкомпьютеры *Blue Jene* [12] предусматривают такую кластеризацию на аппаратно-программном уровне. Однако предусмотреть какие-либо возможности в общем случае применения вычислительных сетей не представляется возможным.

Известны алгоритмы кластеризации, которые можно применять при предварительной обработке вычислительного ресурса, выделяемого для решаемой задачи. Однако они не предусматривают параллельное выполнение и их сложность довольно высока. Это побудило автора разработать и испытать новый, практически приемлемый параллельный алгоритм кластеризации (ПАК). Результатом применения этого алгоритма явилось то, что кластеризация 105 процессоров выполняется почти в 100 раз быстрее, чем с помощью одного процессора.

Задача кластеризации выделенного пользователю ресурса решается Центром *GRID*-технологий.

1. Применение методов кластеризации для распределенных вычислений

1.1. Вычислительные кластеры

Кластеры используют в вычислительных целях, в частности в научных исследованиях. Для вычислительных кластеров существенными показателями являются высокая производительность процессора в операциях над числами с плавающей точкой (*flops*) и низкая латентность объединяющей сети, и менее существенные показатели — скорость операции ввода-вывода, которая в большей степени важна для баз данных и web-сервисов. Вычислительные кластеры позволяют уменьшить

время расчетов, по сравнению с одиночным компьютером, разбивая задание на параллельно выполняющиеся ветки, которые обмениваются данными по связывающей сети. Одна из типичных конфигураций — набор компьютеров, собранных из общедоступных компонентов, с установленной на них операционной системой Linux, и связанных сетью *Ethernet*, *Myrinet*, *InfiniBand* или другими относительно недорогими сетями. Такую систему принято называть кластером *Beowulf*. Специально выделяют высокопроизводительные кластеры (обозначаются англ. аббревиатурой *HPC Cluster — High-performance computing cluster*). Список самых мощных высокопроизводительных компьютеров (также может обозначаться англ. аббревиатурой *HPC*) можно найти в мировом рейтинге *TOP500*. В России ведется рейтинг самых мощных компьютеров СНГ *TOP500* Суперкомпьютеры.

GRID-системы распределенных вычислений не принято считать кластерами, но их принципы в значительной степени сходны с кластерной технологией. Главное отличие — низкая доступность каждого узла, т. е. невозможность гарантировать его работу в заданный момент времени (узлы подключаются и отключаются в процессе работы), поэтому задача должна быть разбита на ряд независимых друг от друга процессов. Такая система, в отличие от кластеров, не похожа на единый компьютер, а служит упрощенным средством распределения вычислений. Нестабильность конфигурации в таком случае компенсируется большим числом узлов.

1.2. Основные методы кластеризации

Под точкой данных ниже будем подразумевать процессор, нуждающийся в передаче рассчитанных результатов по иерархическим связям одновременно с другими процессорами.

Известны следующие методы кластеризации точек данных.

1. **Иерархическая кластеризация** отображается деревом, где концевые вершины — процессоры разбиваются на кластеры. Затем рекурсивно эти кластеры объединяются по парам, образуя кластеры более высокого уровня, и так — до достижения общего кластера корневого уровня.

2. **Разделенная кластеризация:** процессоры объединяются в кластеры на основе их "близкого" (по некоторой метрике) расположения относительно образовавшегося центра.

3. **"Ящик" кластеризации:** этот метод подразумевает разделение всего пространства компьютеров на подпространства, возможно перекрывающихся, "ящиков". Кластеризация при распределении вычислений определяется принадлежностью компьютеров "ящикам".

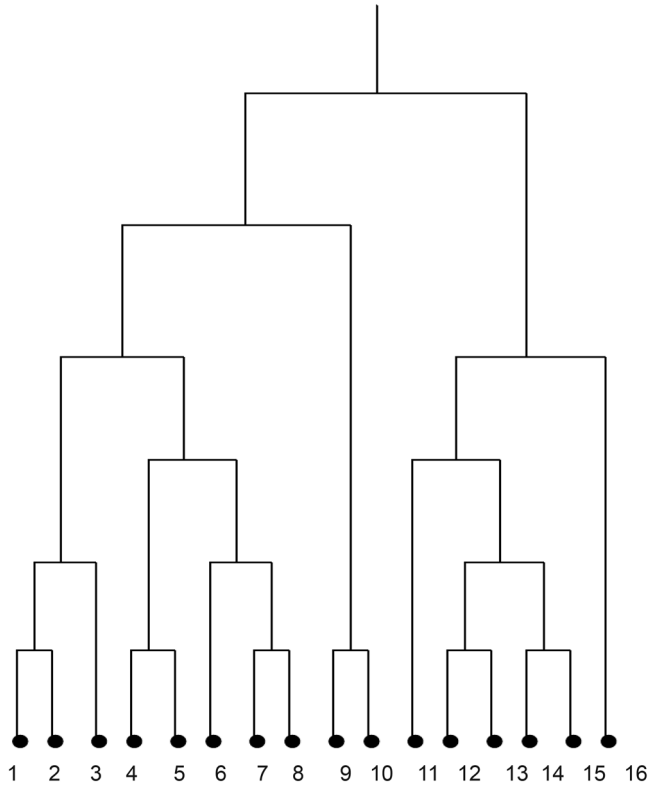


Рис. 1. Древовидная структура иерархической кластеризации

Разделенная кластеризация характеризуется неопределенностью, непредсказуемостью расположения компьютеров. Принцип "ящика" также предполагает неопределенность и затрудняет кластеризацию большого числа процессоров (например, — по всему земному шару).

Иерархическая кластеризация является идеальным методом, имеющим достаточно высокую сложность $O(n^2)$. Однако предлагаемый ниже *параллельный алгоритм* кластеризации значительно снижает этот недостаток и делает иерархические алгоритмы кластеризации более полезными для многих приложений.

Структуру иерархического кластера можно представить древовидной диаграммой (рис. 1).

Для кластеризации процессоров (точек данных), участвующих в счете, необходимо ввести понятие метрик расстояния. Их можно разделить на два класса.

1. **Графические методы.** Этими методами определяются межкластерные расстояния с использованием графа точек в двух кластерах.

Одинарная связь (*Single link*). Расстояние между любыми двумя кластерами определяется, как минимальное расстояние между двумя точками, находящимися в разных кластерах.

Средняя связь (*Average link*). Расстояние между любыми двумя кластерами — среднее расстояние между каждой парой точек, не принадлежащих одному кластеру.

Полная связь (*Complete link*). Определяет расстояние между любыми двумя кластерами как максимальное расстояние между двумя точками, не принадлежащими одному кластеру.

2. **Геометрические методы.** Эти методы определяют центры для каждого кластера и используют эти центры для определения расстояния между кластерами. Например:

- *центр тяжести* (*centroid*) находится на основе евклидова расстояния между центрами кластеров;
- *медиана* (*median*) определяется как невзвешенное среднее расстояние центров двух подкластеров, агломерированных в один кластер. Используется евклидово расстояние между центрами кластеров;
- *минимальное различие* (*minimum variance*): определяется как центр тяжести точек кластера. Расстояние между двумя кластерами находится как сумма квадратов расстояний каждой точки от центра своего кластера с учетом агломерации кластеров.

2. Анализ известных алгоритмов кластеризации

2.1. Алгоритмы кластеризации, допускающие распараллеливание

Ряд известных алгоритмов последовательной иерархической кластеризации приведен в работе [14], некоторые более поздние результаты приведены в работе [6]. Представляют практический интерес параллельные методы, ориентированные на сетевые технологии. Поэтому ниже будут рассмотрены три алгоритма параллельной кластеризации, которые были разработаны учеными *Olsen* [1] и *Discoll et al.* [2, 9] в 1993 г. В этих алгоритмах используются различные метрики для нахождения расстояния между компьютерами. Эти метрики, как и другие детали построения, будут приведены далее.

Алгоритм 1

Шаг 1. Построение упорядоченной структуры данных для возможности определения пар кластеров, наиболее близких друг другу (в смысле принятой метрики).

Шаг 2. Нахождение пары кластеров с минимальным расстоянием между ними и формирование их агломерата.

Шаг 3. Обновление структуры данных с учетом выполненных построений.

Шаг 4. Если структура не сведена к единственному кластеру, перейти к выполнению шага 2.

Важным элементом этого алгоритма является используемая структура данных и метод определения ближайших пар кластеров. Шаг 1 выполняется только один раз, а остальные шаги выполняются $N - 1$ раз, где N — число итераций.

Замечание о возможности реализации алгоритма на вычислительной системе с общей (разделяемой) памятью (PRAM). Создается двумерный массив всех межкластерных расстояний (каждый процессор отвечает за одну колонку) и одномерный массив хранения ближайшего соседа каждого кластера с указанием расстояния до него. Тогда можно определить пару самых близких кластеров по минимуму указанных расстояний между соседями.

Чтобы уточнить структуру данных, каждый процессор обновляет массив межкластерных расстояний, чтобы отразить новое расстояние между кластерами и новые агломерированные кластеры. Так как каждый процессор отвечает только за одно место в массиве, то можно сказать, что каждый процессор должен обновлять единственное место в этом массиве. Никакой операции не следует выполнять для агломерированных кластеров (нового кластера), поскольку новые расстояния зависят только от оставшихся кластеров.

Если ближайший сосед J кластера K агломерируется в новый кластер $I + J$, то новым ближайшим соседом K должен быть новый кластер $I + J$. Ближайший сосед нового кластера определяется с помощью минимизации расстояния других кластеров к нему.

Этот алгоритм при выполнении на n процессорах архитектуры PRAM имеет сложность $O(n)$.

Отметим, что этот алгоритм не эффективен на компьютере с распределенной памятью ввиду необходимости интенсивного обмена.

Алгоритм 2

Шаг 1. Построение упорядоченной структуры данных для быстрого нахождения ближайших соседей.

Шаг 2. Выбор произвольного кластера i_1.

Шаг 3. Формирование списка $i_2 = NN(i_1)$, $i_3 = NN(i_2)$... до $i_k = NN(i_{k-1})$ и $i_{k-1} = NN(i_k)$, где $NN(x)$ является самым близким соседом кластера x.

Шаг 4. Агломерация кластеров i_{k-1} и i_k.

Шаг 5. Обновление структуры данных.

Шаг 6. Если структура не сведена к единственному кластеру, перейти к выполнению шага 3, используя предыдущий i_{k-2} как новый i_1 при $k > 2$, при $k = 1$ следует выбрать i_1 произвольно.

Легко видеть, что ключ к быстрому выполнению этого алгоритма заключается в быстром определении ближайших соседей.

Отметим, что этот алгоритм, как и алгоритм 1, можно использовать для решения на архитектуре PRAM. Однако этот алгоритм более сложный, так как требует $3n$ раз выполнить поиск минимального расстояния, в то время, как алгоритм 1 требует такого поиска $2n$ раз.

Алгоритм 3

Данный алгоритм используется для особого случая одинарной связи (*Single link*) на локальной памяти компьютера. Здесь широко используется понятие *параллельного минимального покрывающего дерева*, определенного *Driscoll* et al. [2, 9].

Минимальное покрывающее дерево — это остовное дерево графа, имеющее минимальный возможный вес, где под весом дерева понимается сумма весов входящих в него ребер.

В предлагаемом автором (в следующем разделе) алгоритме параллельной кластеризации алгоритм построения такого дерева заимствован. Его применение позволило легко преобразовать минимальное покрывающее дерево в кластерную иерархию.

Алгоритм 3 предполагает следующие шаги.

Шаг 1. Построение структуры данных, удовлетворяющей требованию быстрой оценки расстояния каждого пункта от текущего минимального покрывающего дерева. (В начале выполнения алгоритма произвольный пункт является минимальным покрывающим деревом).

Шаг 2. Нахождение пункта p, не входящего в состав текущего минимального покрывающего дерева, который является самым близким к минимальному покрывающему дереву.

Шаг 3. Включение пункта p в минимальное покрывающее дерево.

Шаг 4. Обновление структуры данных.

Шаг 5. Если существует пункт, не вошедший в минимальное покрывающее дерево, перейти к шагу 2.

Используемая структура данных может просто представлять собой массив расстояний каждого кластера от минимального покрывающего дерева. Этот массив распространен через процессоры таким образом, что процессор, ответственный за кластер, сохраняет все расстояния для этого кластера. На *шаге 1* этот массив входит в массив расстояний каждого пункта от произвольного пункта старта.

Шаг 2 основан на поиске минимума значений в этом массиве. Для обновления структуры данных используется местоположение пункта p.

Так как *шаги 2* и *4* инициируют сложность $O(\log n)$ и выполняются n раз, то общая сложность алгоритма 3 составляет $O(n \log n)$.

2.2. Метрики кластеризации

В работе [8] можно ближе познакомиться с понятием метрики для проведения кластеризации: для агломерации или для определения расстояния до минимального покрывающего дерева. Поскольку в предложенном ниже алгоритме кластеризации целесообразно использовать только метрики Евклидову и Манхэттен, приведем таблицу этих практически применяемых метрик (табл. 1).

Таблица 1

Методы определения расстояний между точками данных

Расстояние	Формула
Евклидово	$\|a - b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$
Манхэттен	$\|a - b\|_1 = \sum_i \|a_i - b_i\|$

3. Предлагаемый алгоритм ПАК

3.1. Предпосылки

Многие методы с различными прикладными целями были разработаны, чтобы решить проблему кластеризации, включая метод *k-means* (иногда называемый методом *k*-средних) [5], карту самоорганизации (*Self organizing map*) [5], алгоритм кластера Маркова [4] и неконтролируемый алгоритм кластеризации для графов, основанных на потоках в сетях [10].

Самый главный недостаток всех методов заключается в том, что большинство алгоритмов имеют дело с относительно маленькими количествами данных. Когда речь идет о задачах большой размерности, эти алгоритмы оказываются вообще слишком медленными, чтобы быть фактически полезными. Другой проблемой является устойчивость по отношению к большим объемам структурируемых данных.

Анализ показал, что в основу предлагаемого алгоритма ПАК (параллельного алгоритма кластеризации) следует положить построение и анализ *минимального покрывающего дерева*. Именно такой подход и обеспечивает устойчивость в том случае, если объемы данных чрезвычайно высоки и компьютеры находятся далеко друг от друга.

3.2 Алгоритм Прима — ближайший прообраз ПАК

Единственным алгоритмом, который наиболее близок к предложенному алгоритму ПАК, является *алгоритм Прима* [3], который допускает распараллеливание при графической реализации метода одинарной связи (*Single link*). Этот алгоритм заключается в построении минимального покрывающего дерева взвешенного связного неориентированного графа, объединяющего точки дан-

Таблица 2

Сложность алгоритма Прима

Способ представления графа и приоритетной очереди	Асимптотика
Массив d, списки смежности (матрица смежности)	$O(V^2)$
Бинарная пирамида, списки смежности	$O((V + E)\log(V)) = O(E\log V)$
"Куча" Фибоначчи, списки смежности	$O(E + V\log V)$

ных, т. е. компьютеры, выработавшие данные для обмена. Алгоритм впервые был открыт в 1930 г. чешским математиком Войцехом Ярником, позже переоткрыт Робертом Примом в 1957 г., и, независимо от них, Э. Дейкстрой в 1959 г.

Построение начинается с дерева, имеющего одну (произвольную) вершину. В течение работы алгоритма дерево разрастается, пока не охватит все вершины исходного графа. На каждом шаге алгоритма к текущему дереву присоединяется самое "легкое" из ребер, соединяющих вершину из построенного дерева и вершину не из дерева.

О сложности [3] алгоритма Прима можно судить по табл. 2.

Алгоритм Прима является последовательным. Однако известны его параллельные модификации.

3.3. Построение ПАК

Пусть граф $G = (E, V)$ — граф, связывающий точки данных, где E — число дуг и V — число вершин.

Построим $MST(G)$ — минимальное покрывающее дерево графа G.

Алгоритм ПАК предполагает следующие основные шаги.

- Разделение G на s подграфов, $G_j = \{(E_j, V_j)\}$, $j = 1, ..., s$, где позже в этом разделе будет определяться стоимость s; V_j — множество вершин в G_j; $E_j \subset E$ — множество дуг, соединяющих вершины V_j.
- Выделение двудольных графов $B_{ij} = \{V_i \cup V_j, E_{ij}\}$, где V_i и V_j являются множествами вершин G_i и G_j, и $E_{ij} \subset E$ — множества дуг между вершинами V_i и V_j, $i \neq j$, для каждой такой пары подграфов из $\{G_i\}$.
- Параллельное построение минимального покрывающего дерева (T_{ii}) на каждом G_i и минимального покрывающего дерева (T_{ij}) на каждом B_{ij}.
- Построение нового графа $G^0 = \cup T_{ij}$, $1 \leqslant i \leqslant j \leqslant s$, по объединению всех минимальных покрывающих деревьев на предыдущем шаге. Граф G^0 является подграфом графа G с множеством вершин V и дуг деревьев T_{ij}, $1 \leqslant i \leqslant j \leqslant s$.
- Построение минимального покрывающего дерева ($MST(G^0)$) на графе G^0.

Здесь не приведено полученное автором математическое доказательство того, что $MST(G^0) = MST(G)$.

Таким образом, ключевая идея алгоритма заключается в том, что вычисляется MST (минимальное покрывающее дерево) для каждого подграфа и каждых вспомогательных двудольных графов, которые формируются на каждой паре подграфов *параллельно*. Тогда граф G^0 создается путем слияния построенных минимальных покрывающих деревьев, а также попутно получается $MST(G^0)$ древовидной структуры.

4. Моделирование и реализация ПАК

Для оценки сложности алгоритма ПАК целесообразно воспользоваться результатами анализа времени выполнения алгоритма Прима в части построения минимального покрывающего дерева на подграфах. При этом следует воспользоваться "кучей" Фибоначчи [15] для каждого подграфа G_j, и каждого двудольного графа B_{ij}, чтобы облегчить операцию "поиска следующей маленькой дуги" в алгоритме Прима, который имеет оценку сложности $O(|E| + |V|\log(|V|))$ для построения каждого подграфа G_i, и $O(|V_i||V_j| + (|V_i| + |V_j|\log(|V_i| + |V_j|))$ для формирования каждого двудольного графа.

Было использовано программное обеспечение Matlab 7 и следующие исходные условия.

Наборы данных составляют от 10 000 до 500 000 (с шагом 10 000) 10-, 20-, 30-, ..., 100-мерных векторов (размерность D). Каждый компонент набора данных является независимым и равномерно распределенным в интервале [0, 1] реальным значением. Использовались два метода оценки расстояния — *Евклидов* и *Манхэттен* (см. табл. 1).

Для каждого метода нахождения расстояния необходимо формировать зависимость времени построения MST $T(D, V)$ от V и D, где V — число объектов; D — размерности объектов. Для этого использовалась линейная регрессионная модель, отображенная табл. 2, где $v = V \cdot 0{,}0001$, а R — коэффициент корреляции множественной регрессии для полного графа $T_C(D, V)$ и для двудольного графа $T_B(D, V)$. Результаты, показанные в табл. 3, используются для анализа вычислительного времени построения MST первоначального графа.

Таблица 3

Линейная модель регресса конструкции МОД в течение времени

Расстояние	$T_C(D, V)$	R	$T_B(D, V)$	R
Евклидово	$(0{,}192D + 1{,}61)v^2$	0,999	$(0{,}385D + 3{,}80)v^2$	0,999
Манхэттен	$(0{,}206D + 1{,}58)v^2$	0,999	$(0{,}413D + 3{,}62)v^2$	0,999

Следующим шагом является объединение всех минимальных покрывающих деревьев, полученных на предыдущем шаге, и создание окончательного дерева.

Однако время построения полного графа зависит не только от D и V, но и от числа s подграфов. Очевидно, что чем больше число произведенных разбиений (ускоряющее одновременное построение минимального покрывающего дерева), тем большее число дуг образуется в графе G^0 (что следует из увеличения времени построения оригинального окончательного минимального покрывающего дерева на последнем шаге). Поэтому до нахождения времени объединения всех минимальных покрывающих деревьев $T_M(s, V)$ необходимо узнать оптимальное значение s.

На шаге предварительной обработки время выполнения предложенного параллельного алгоритма равно $\max\{\max\limits_{1 \leqslant i \leqslant s} RT(G_i), \max\limits_{1 \leqslant i \leqslant s} RT(B_{ij})\}$, где $RT(G_i)$ и $RT(B_{ij})$ — время построения графа G_i и B_{ij} соответственно. Наша цель — вычислить число разделов (s) множества V, который минимизирует $\max\{\max\limits_{1 \leqslant i \leqslant s} RT(G_i), \max\limits_{1 \leqslant i \leqslant s} RT(B_{ij})\}$.

Можно доказать, что, если $s = 2$, $\min\{\max\limits_{1 \leqslant i \leqslant s} RT(G_i), \max\limits_{1 \leqslant i \leqslant s} RT(B_{ij})\}$ достигнуто при условии $V_2 = 2V_1$. И для $s > 2$ оптимальное число разделов достигнуто при $V_i = 1/s$, $i = 1, 2, ..., s$.

(Здесь не приводится математическое доказательство.)

Теперь можно сказать, что время построения полного окончательного графа вычисляется по уравнению

$$T(D, V, s) = T_B(D, V/s) + T_M(s, V). \qquad (1)$$

Как обещано во введении, проверим эффективность алгоритма ПАК с множеством данных 1 000 000 точек. На первом шаге построим график зависимости времени построения минимального покрывающего дерева от числа подграфов, чтобы узнать оптимальное число $s_{\text{опт}}$ (рис. 2).

Для этого положим размерность $D = 40$.

На основе анализа полученного графика можно считать, что $s_{\text{опт}} = 14$.

Рассчитаем ускорение $SU_T(D, V)$, получаемое при построении минимального покрывающего дерева с использованием параллельных вычислений по сравнению с применением одного процессора:

$$SU_T(D, V) = \frac{T(D, V, 1)}{T(D, V, s(D, V))}. \qquad (2)$$

Результаты, отраженные в табл. 4, показывают, что алгоритм ПАК может решить проблему кластерной идентификации, на множестве данных

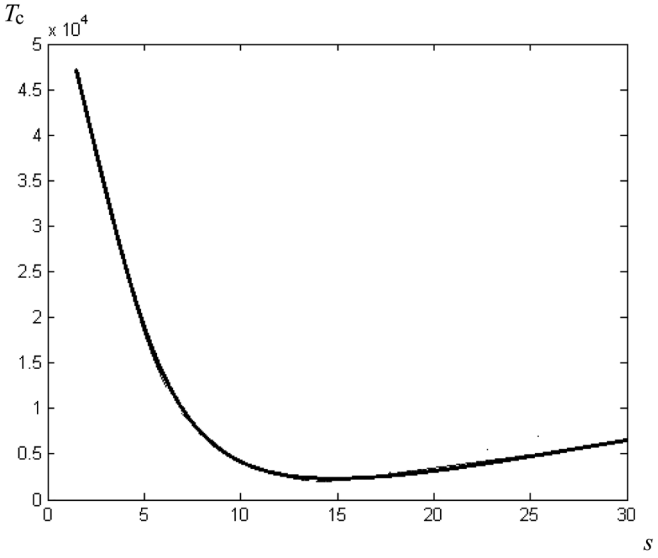


Рис. 2. График зависимости времени построения минимального покрывающего дерева от числа подграфов s

Таблица 4

Теоретический коэффициент ускорения SU параллельной конструкции MST по сравнению с реализацией на одном процессоре

V	D	s	Расстояние	
			Евклидово	Манхэттен
			SU	SU
250 000	20	9	20	21
500 000	40	10	42	43
750 000	60	13	65	67
1 000 000	80	14	90	93

с 1 000 000 точек на графе, почти в 90—93 раза быстрее, чем на одном процессоре. Это доказывает преимущества алгоритма ПАК.

Найдя оптимальное число подграфов, можно вычислить число процессоров, необходимых для того, чтобы решить поставленную задачу. На основе арифметической прогрессии число процессоров получается равным $s(s + 1)/2$, где s —число подграфов. В данном случае $s = 14$, т. е. для того чтобы получить оптимальную скорость обмена, необходимо 105 процессоров.

Заключение

Предложенный параллельный алгоритм кластеризации способен почти в 100 раз быстрее сформировать трафик для сбора данных при реализации распределенных GRID-вычислений.

Осталось лишь экспериментально установить условия комплексной эффективности распределенных вычислений на основе сетевых технологий, что является темой дальнейших исследований.

Список литературы

1. **Olson C. F.** Parallel Algorithms for Hierarchical Clustering // Parallel Computing. 1995. Vol. V21. P. 1313—1325.
2. **Минимальное** остовое дерево. URL: http://ru.wikipedia.org/wiki/Минимальное_остовое_дерево.
3. **Алгоритм** Прима. URL: http://ru.wikipedia.org/wiki/Алгоритм_Прима.
4. **Bezdek J. C.** Pattern Recognition with Fuzzy Objective Function Algorithms. New York: Plenum Press, 1981.
5. **Bishop C. M.** Neural Networks for Pattern Recognition. Oxford: Oxford Univ. Press, 1995.
6. **Day W. H. E., Edelsbrunner H.** Efficient algorithms for agglomerative hierarchical clustering methods // Journal of Classification. 1984. 1(1). 7(24).
7. **Olman V., Xu D., Xu Y.** CUBIC: Identification of Regulatory Binding Sites through Data Clustering // J. Bioinformatics and Computational Biology. 2003. Vol. 1, N 1. P. 21—40.
8. **Hierarchical** clustering. URL: http://en.wikipedia.org/wiki/Hierarchical_clustering
9. **Driscoll J. R., Gabow H. N., Shrairman R., Tarjan R. E.** Relaxed An alternative to Fibonacci heaps with applications to parallel Communications // ACM. Nov. 1988. N 31(11). P. 1343—1354.
10. **Enright A. J., Van Dongenl S., Ouzounis S. A.** An Efficient Algorithm for Large-Scale Detection of Protein Families // Nucleic Acids Research. 2002. Vol. 30, N 7. P. 1575—1584.
11. **Olman V., Mao F., Wu H., Xu Y.** Parallel Clustering Algorithm for Large Data Sets with Applications in Bioinformatics // IEEE / ACM Transactions on Computational Biography and Bioinformatics. 2009. April—june. Vol. 6, N 2.
12. **Blue Gene.** URL: http://ru.wikipedia.org/wiki/Blue_Gene
13. **Murtagh F.** Clustering in Massive Data Sets. Dordrecht: Handbook of Massive Data Sets, 2002. P. 501—543.
14. **Murtagh F.** A survey of recent advances in hierarchical clustering algorithms // Computer Journal. 1983. N 26. P. 354—359.
15. **Rosen K. H.** Handbook of Discrete and Combinatorial Mathematics. Texas: CRC Press, 1999.

УДК 004.272.43

А. Э. Саак, канд. техн. наук, доц.
Технологический институт
Южного федерального университета
в г. Таганроге,
e-mail: saak@tti.sfedu.ru

Локально-оптимальные ресурсные распределения

Компьютерное обслуживание в Grid-системах, многопроцессорных вычислительных системах требует распределения протяженного горизонтального множества прямоугольных координатных ресурсных элементов с переменными звеньями — значениями параметров вдоль осей координат целочисленной плоскости в квадратную рамку поля ресурсов вычислительной системы. Протяженность массива упомянутых требований на обслуживание превосходит значения параметров (измерений) поля ресурсов, и возникает задача симметричной локализации линейно протяженного множества ресурсных элементов в ресурсную оболочку — подмножество упомянутой рамки. Данная задача локализации подчиняется парному целевому критерию минимизации асимметрии измерений ресурсной оболочки и максимизации коэффициента заполнения оболочки заданными ресурсными прямоугольниками массива требований. Указываются алгоритмы локализации, зависящие от квадратного типа массива требований. Вводятся определения кругового, гиперболического, параболического массивов и устанавливаются оценки показателей локализации.

Ключевые слова: *Grid-система, многопроцессорная вычислительная система, диспетчирование, локальное расписание, оптимальное расписание, квадратичный тип массива требований пользователей, принцип минимума асимметрии измерений объемлющего ресурсного прямоугольника*

Введение

Среди основных целевых критериев оптимизации планарных расписаний распределения процессорных и временных ресурсов [1—6] (минимизация ресурсной оболочки, минимизация асимметрии той же оболочки, минимизация неиспользуемых элементов внутри ресурсной оболочки, минимизация числа стадий обслуживания исходного массива ресурсных прямоугольников [7] и некоторые другие) центральную роль играет критерий симметризации ресурсной оболочки локализованного внутреннего аддитивно-графического представления массива ресурсного спроса множества пользователей.

В работе предлагается выполнение локализации по критерию минимизации асимметрии объемлющего ресурсного прямоугольника для массивов кругового, гиперболического и параболического квадратичных типов. Ставится задача динамического упорядочения массивов упомянутых квадратичных типов, на каждом шаге которого достигается уменьшение асимметрии имеющейся аддитивной графики ресурсных прямоугольников спроса. Критерий симметрии ресурсной оболочки дополняется в работе показателем заполнения оболочки ресурсными прямоугольниками заявок. Оба показателя подлежат максимизации. При этом мы ограничиваемся неполной оптимизацией парного целевого критерия, достигая лишь некоторого улучшения показателей симметричной заполненности ресурсной оболочки по отношению к первоначально протяженному массиву спроса пользователей.

1. Локализация круговых массивов

Заявку пользователя для обслуживания диспетчером операционной системы многопроцессорной вычислительной системы (МВС) геометрически можно представить координатным ресурсным прямоугольником с горизонтальным измерением, равным числу единиц ресурса процессоров, и вертикальным измерением — временем, требуемым для обработки [6]. Заявку, требующую a единиц процессоров и b единиц времени, обозначим символом $a \times b$ или $[(a, b)]$.

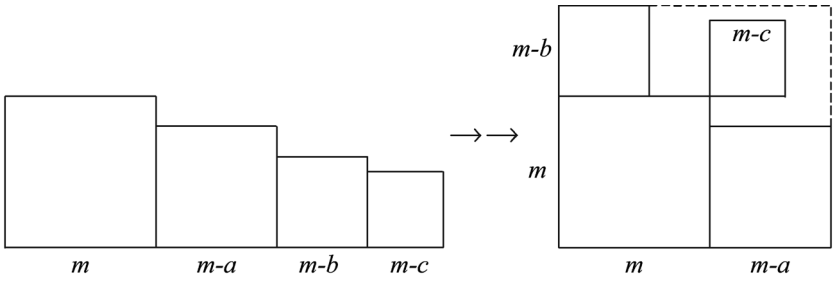


Рис. 1. Кольцевой алгоритм аддитивной графики четырех ресурсных квадратов

Имеется четверка ресурсных квадратов, упорядоченная по убыванию измерений,

$$a(j_1) \times a(j_1),\ a(j_1)\downarrow,\ j_1\uparrow, \quad j_1 = 1, 2, 3, 4.$$

Требуется найти ресурсную оболочку данных заявок по критерию симметрии измерений.

С этой целью вводим обозначения $m \times m$, $(m - a) \times (m - a)$, $(m - b) \times (m - b)$, $(m - c) \times (m - c)$ предыдущих ресурсных квадратов, $a \leqslant b \leqslant c$, и локализуем данные квадраты кольцевым алгоритмом. Максимальную ресурсную заявку помещаем в начало первого квадранта. Справа суперпозируем второй ресурсный квадрат, на верхнем основании первого суперпозируем третий ресурсный квадрат. Конечный квадрат синтезируем с начальным вершинно-диагональным сцеплением. В итоге получаем аддитивную графику (рис. 1) с ресурсной оболочкой $(m + m - a) \times (m + m - b)$.

Последняя имеет горизонтальную форму $(m + m - a) \geqslant (m + m - b)$, так как $a \leqslant b$ по условию, и показатель асимметрии $(2m - a) - (2m - b) = b - a$.

Горизонтальную форму ресурсных прямоугольников принимаем в качестве признака массива кругового типа.

Линейную вертикальную полиэдраль

$$a(j_2) \times b(j_2),\ a(j_2) \geqslant b(j_2),$$

$$j_2 \in \left[0, \left[\frac{k}{2}\right] - 1\right] \subset Z^1, \qquad (1)$$

упорядоченную по убыванию оснований ресурсных прямоугольников $a(j_2)\downarrow$, $j_2\uparrow$, суперпозируем последовательно слева от линии $Y_1 = a(0) = \max\limits_{j_2} a(j_2)$. Справа от той же линии суперпозируем вертикально линейную полиэдраль

$$\lambda(j_2) \times \beta(j_2),\ \lambda(j_2) \geqslant \beta(j_2),$$

$$j_2 \in \left[0, \left[\frac{k}{2}\right] - 1\right] \subset Z^1, \qquad (2)$$

также упорядоченную по убыванию оснований ресурсных прямоугольников $\lambda(j_2)\downarrow$, $j_2\uparrow$.

Получаем аддитивную графику ресурсных прямоугольников (1), (2) (рис. 2) с ресурсной оболочкой

$$(a(0) + \lambda(0)) \times \max\left\{\sum_{j_2=0}^{\left[\frac{k}{2}\right]-1} b(j_2);\ \sum_{j_2=0}^{\left[\frac{k}{2}\right]-1} \beta(j_2)\right\} \qquad (3)$$

суммарных максимальных оснований на максимум сумм вертикальных измерений каждой из полиэдралей.

При равенстве измерений ресурсной оболочки (3) множество (1), (2) относим к массивам с ограниченной асимметрией:

$$a(0) + \lambda(0) = \max\left\{\sum_{j_2=0}^{\left[\frac{k}{2}\right]-1} b(j_2);\ \sum_{j_2=0}^{\left[\frac{k}{2}\right]-1} \beta(j_2)\right\}. \qquad (4)$$

Если упомянутое множество прямоугольников образует спрос на ресурсные элементы и общее значение параметров ресурсной оболочки совпадает со стороной рамки $M \times M$ операционного поля МВС, то эксперимент ресурсных распределений предложенного синтеза вертикальных полиэдралей называем каноническим круговым распределением (рис. 3).

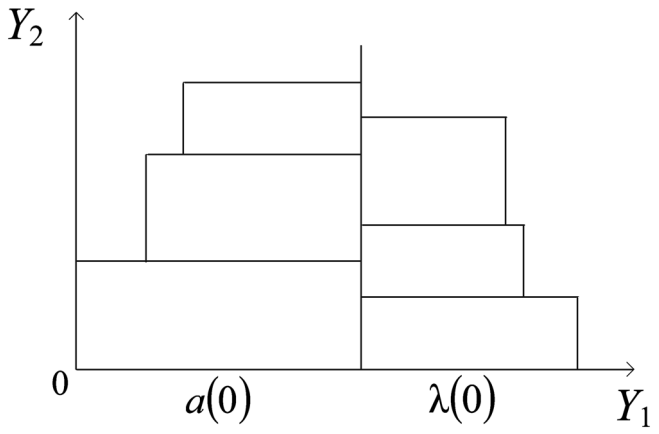


Рис. 2. Локализация массива кругового типа кольцевым алгоритмом

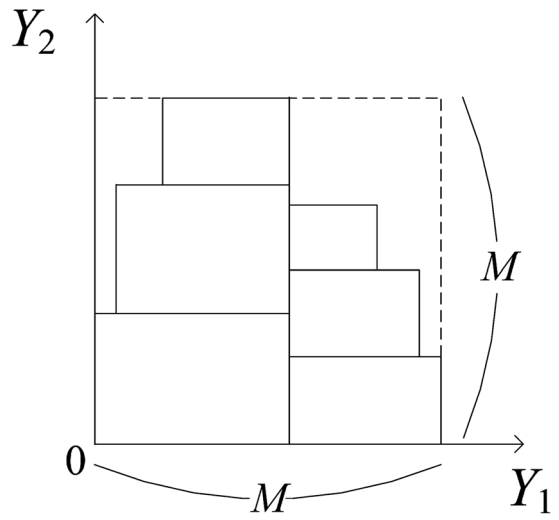


Рис. 3. Каноническое круговое распределение

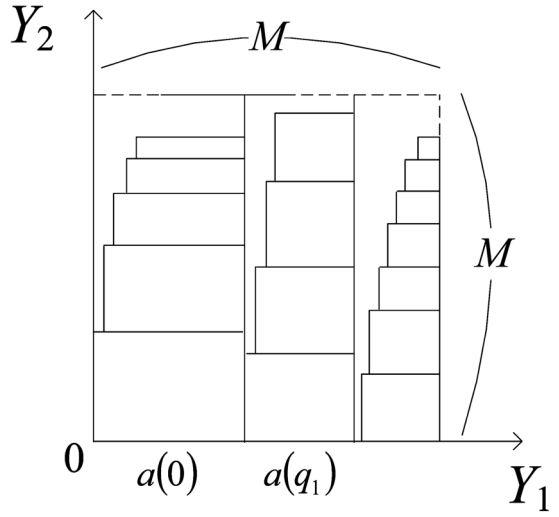


Рис. 4. Эксперимент ресурсных распределений кругового спроса

При числе k ресурсных прямоугольников в каждом из k модулей относим k-модульный спрос к каноническому круговому классу массивов при возможности факторизации спроса на массивы кругового типа с ограниченной асимметрией с выполнением соотношений [4], в которых общее значение параметров совпадает с M. В данном каноническом случае эксперимента ресурсных распределений число модулей одновременно указывает минимальное число стадий обслуживания.

Рассмотренную версию относим к блочно-каноническому случаю и переходим к глобально-канонической рамочной задаче.

Приведем общее определение эксперимента ресурсных распределений кругового спроса:

$$a(j_1) \times b(j_1),\ a(j_1) \geqslant b(j_1),\ a(j_1)\downarrow,\ j_1\uparrow.$$

С этой целью в начальной полосе $0 \leqslant Y_1 \leqslant a(0)$ проводим вертикальную последовательную суперпозицию ресурсных прямоугольников $\bigcup_{j_1=0}^{q_1-1} [(a(j_1),\ b(j_1))]$ слева от линии $Y_1 = a(0)$ до наилучшего приближения с недостатком $\sum_{j_1=0}^{q_1-1} b(j_1) = M - 0$ к высоте рамки (рис. 4). Здесь q_1 — мощность ресурсных прямоугольников, суперпозированных в начальной полосе. Далее строим линию $Y_1 = a(0) + a(q_1)$ и слева от данной линии выполняем предыдущее построение по отношению к оставшейся части массива $\sum_{j_1=q_1}^{q_1+q_2-1} b(j_1) = M - 0$, где q_2 — мощность ресурсных прямоугольников следующей полосы. Циклическое повторение указанных действий приведет к исчерпанию массива и локализации последнего в полосу $0 \leqslant Y_2 \leqslant M$ некоторой протяженности, составленной из предыдущих отрезков $[0,\ a(0)] \cup [a(0),\ a(0) + a(q_1)] \cup \ldots$

Укажем решение рамочной задачи для кругового спроса, локализованного в указанную полосу.

Вертикальные линейные полиэдрали, из которых составлена полоса $0 \leqslant Y_2 \leqslant M$, подлежащая рамочному ресурсному распределению по правилам координатности, аддитивности и цельности, обладают свойством убывания оснований ресурсных прямоугольников вдоль вертикальной координаты $j_2\uparrow$, $a\downarrow$. Здесь $a(j_1, j_2)$ — основание полиэдральной грани в упомянутой полосе. Поэтому требуемому распределению по линейкам $[0,\ M]$ подлежат звенья-измерения горизонтальной линии $Y_2 = 0$. Данная. цель достигается последовательной суперпозицией с наилучшим приближением с недостатком протяженности M—0 посредством указанных измерительных элементов $a(j_1, 0)$ — горизонтальных оснований ресурсных прямоугольников-заявок пользователей.

2. Локализация гиперболических массивов

Имеется пятерка ресурсных прямоугольников

$$a \times b,\ a_1 \times b_1,\ a_0 \times b_0,\ \lambda_1 \times \beta_1,\ \lambda \times \beta,$$

с убывающими высотами

$$b \geqslant b_1 \geqslant b_0 \geqslant \beta_1 \geqslant \beta,$$

растущими основаниями

$$a \leqslant a_1 \leqslant a_0 \leqslant \lambda_1 \leqslant \lambda$$

и центральным элементом минимальной асимметрии

$$a_0 \times b_0,\ |b_0 - a_0| =$$
$$= \min\{|b - a|,\ |b_1 - a_1|,\ |b_0 - a_0|,\ |\beta_1 - \lambda_1|,\ |\beta - \lambda|\}.$$

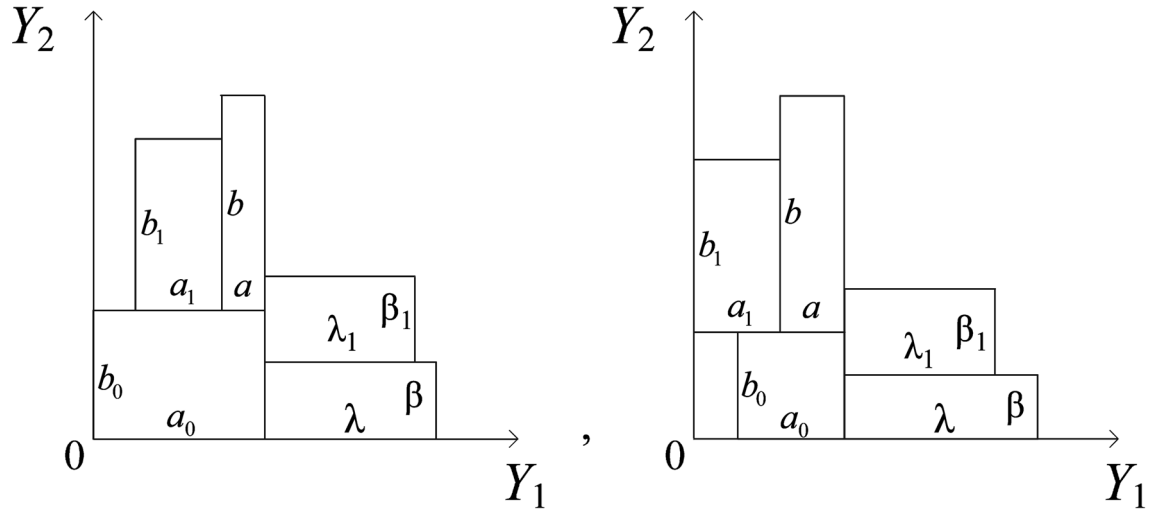


Рис. 5. Локализация массива гиперболического типа угловым алгоритмом

Требуется найти ресурсную оболочку данных заявок по критерию симметрии измерений.

Проведем локализацию угловым алгоритмом. Центральный элемент минимальной асимметрии располагаем слева от линии $Y_1 = \max\{a_0, a + a_1\}$. На уровне верхнего основания суперпозируем первые два ресурсных прямоугольника вдоль горизонтали от линии боковой стороны центрального элемента. Далее суперпозируем вертикально последние два ресурсных прямоугольника вправо от той же линии боковой стороны (рис. 5).

Получаем ресурсную оболочку $(\max\{a_0, a + a_1\} + \lambda) \times (b_0 + b)$ с показателем асимметрии $|(b_0 + b) - (\max\{a_0, a + a_1\} + \lambda)|$ и коэффициентом заполненности оболочки заданными ресурсными прямоугольниками $\dfrac{ab + a_1b_1 + a_0b_0 + \lambda_1\beta_1 + \lambda\beta}{(\max\{a_0, a + a_1\} + \lambda)(b_0 + b)}$.

Обобщая данный модельный случай, одномодульный массив

$$a(j_1) \times b(j_1),\ b(j_1)\downarrow,\ a(j_1)\uparrow,\ j_1\uparrow,$$

с центральным элементом минимальной асимметрии $|b(j_1^*) - a(j_1^*)| = \min\limits_{j_1} |b(j_1) - a(j_1)|$ относим к гиперболическому типу и ставим задачу построения ресурсной оболочки заданных ресурсных прямоугольников по критерию симметрии измерений.

Как и выше, строим линию $Y_1 = \max\left\{a(j_1^*), \sum\limits_{j_1 = 0}^{j_1^* - 1} a(j_1)\right\}$ по центральному элементу с минимальной асимметрией и значению суммы оснований предшествующих ресурсных прямоугольников. Слева от данной вертикали на уровне верхнего основания центрального элемента суперпозируем горизонтально элементы с номерами $j_1 < j_1^*$, а справа суперпозируем вдоль указанной вертикали ресурсные прямоугольники с $j_1 > j_1^*$ до исчерпания модуля. Получаем ресурсную оболочку

$$\left(\max\left\{a(j_1^*), \sum_{j_1 = 0}^{j_1^* - 1} a(j_1)\right\} + a(k-1)\right) \times$$

$$\times \left(\max\left\{b(j_1^*) + b(0), \sum_{j_1 = j_1^* + 1}^{k-1} b(j_1)\right\}\right) \qquad (5)$$

заданных ресурсных прямоугольников гиперболического массива.

При равенстве измерений ресурсной оболочки (5) гиперболический массив относим в класс модулей с ограниченной асимметрией измерений

$$\max\left\{a(j_1^*), \sum_{j_1 = 0}^{j_1^* - 1} a(j_1)\right\} + a(k-1) =$$

$$= \max\left\{b(j_1^*) + b(0), \sum_{j_1 = j_1^* + 1}^{k-1} b(j_1)\right\}. \qquad (6)$$

Если упомянутое множество образует спрос на ресурсные элементы и общее значение параметров (измерений) ресурсной оболочки совпадает со стороной рамки $M \times M$ операционного поля МВС, то эксперимент ресурсных распределений предложенного синтеза полиэдралей называем каноническим гиперболическим распределением (рис. 6).

При мощности k ресурсных прямоугольников в каждом модуле k-модульный спрос относим к каноническому гиперболическому классу массивов при возможности факторизации спроса на массивы гиперболического типа с ограниченной

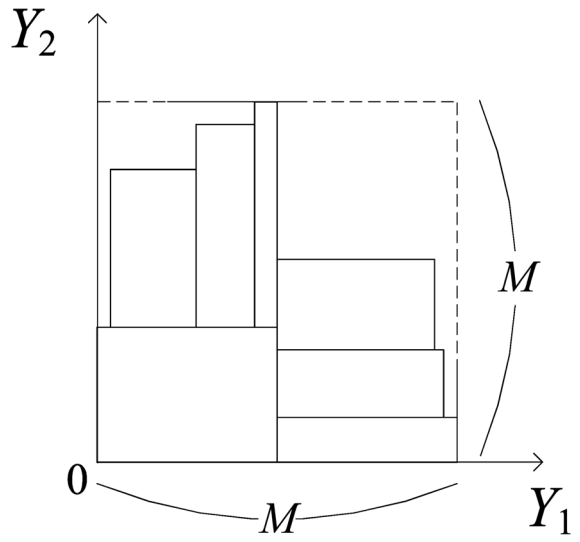


Рис. 6. Каноническое гиперболическое распределение

асимметрией с выполнением соотношений (6), в которых общее значение параметров совпадает с M. В данном каноническом случае эксперимента ресурсных распределений число модулей одновременно указывает минимальное число стадий обслуживания.

Рассмотренную версию относим к блочно-каноническому случаю и переходим к глобально-канонической рамочной задаче.

Воспроизведем построение, аналогичное приведенному выше в разделе 1, для массива ресурсных прямоугольников $a(j_1) \times b(j_1)$ с условием убывания высот и роста оснований $b(j_1)\downarrow$, $a(j_1)\uparrow$, $j_1\uparrow$, определяющим линейную полиэдраль гиперболического типа.

С этой целью выполняем транспонирование нумерации ресурсных прямоугольников $j_1 \to\to (k - j_1)$, $a(j_1) \to\to a(k - j_1)$. Новый массив будет состоять из ресурсных прямоугольников с убывающими основаниями: $a(k - j_1)\downarrow$, $j_1\uparrow$. По аналогии с предыдущим (см. разд. 1) локализуем заданный гиперболический массив заявок пользователей в полосу $0 \leqslant Y_2 \leqslant M$ некоторой протяженности.

Рамочная задача для гиперболического спроса, локализованного в указанную полосу, решается аналогично изложенному в разд. 1.

3. Параболическая полиэдраль

Предыдущие квадратичные типы массивов локализации — круговой и гиперболической — дополняются в данном разделе локализацией параболических массивов ресурсных прямоугольников — заявок пользователей. Тем самым, охватывается случай резко выраженной асимметрии измерений 1-го, 2-го родов (процессорных и временных).

Линейную полиэдраль ресурсных прямоугольников, состоящую из k модулей мощности k,

$$\bigcup_{i'=1}^{k} \bigcup_{j'=1}^{k} [(a_{i'}(j'),\ b_{i'}(j'))], \qquad (7)$$

относим к параболическому типу при выполнении соотношений

$$\sum_{j'=1}^{k} a_{i'}(j') = L_k = \text{const},\ i' = 1, 2, ..., k, \qquad (8)$$

$$\frac{1}{k} \sum_{j'=1}^{k} b_{i'}(j') = h_k = \text{const},$$

$$i' = 1, 2, ..., k, \quad h_k = 1 + \frac{o(k)}{k}. \qquad (9)$$

При ограниченности дисперсий в распределении высот ресурсных прямоугольников выводим отсюда асимптотическое значение столбцов $\sum_{i'=1}^{k} b_{i'}(j') = k + o(k)$ в матрице строк-модулей, расположенных вертикально внутри полосы шириной L_k. Модули располагаем последовательной минимизацией расстояний по вертикали, сохраняя аддитивность, координатность, цельность граней в матричном массиве. Последний заключаем в ресурсную оболочку $L_k \times (k + o(k)) \approx L_k \times k$.

Изложенный алгоритм локализации первоначально протяженной линейной полиэдрали принимаем в качестве эксперимента назначений канонического параболического спроса (7)—(9). При $L_k = M$ матричный синтез параболического спроса индуцирует канонический эксперимент ресурсных распределений параболического типа. Здесь $M \times M$ — рамка операционного поля ресурсных распределений МВС.

Основным источником массивов с выраженной асимметрией измерений можно считать умноже-

$$\bigcup_{j_1=0}^{k} \left[a(j_1) \times \bigcup_{j_2=0}^{k} b_{j_1}(j_2) \right] \to\to$$

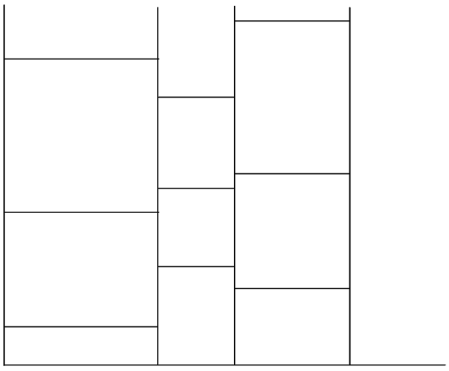


Рис. 7. Аддитивная графика массива триодной структуры параболического типа

ние измерений $a(j_1) \times b_{j_1}(j_2)$ массивов триодной структуры. В последней линейный полигон звеньев-измерений 1-го рода умножается на семейство линейных полигонов звеньев-измерений 2-го рода с характерной для параболических распределений асимметрией мощностей горизонтальной $\{a(j_1)\}_{j_1=0}^{k}$ и вертикальной $\{b_{j_1}(j_2)\}_{j_1,j_2=0}^{k}$ компонент аддитивной графики (рис. 7).

Модельный массив (рис. 8)

$$C_k^{(j_1)} \times C_{k-j_1}^{(|j_2|)},$$
$$|j_2| \in [0,\ k-j_1] \subset Z^1,\ j_1 \in [0,\ k] \subset Z^1, \qquad (10)$$

синтезируем в вертикальную полосу $\sum_{j_1=0}^{\left[\frac{k}{2}\right]} C_k^{(j_1)} \times$ $\times (1 + 2^k)$ посредством вертикальной суперпозиции верхней половины массива $j_2 \geqslant 0$.

С этой целью вертикальные слои первой части массива $0 \leqslant Y_1 \leqslant \sum_{j_1=0}^{\left[\frac{k}{2}\right]} C_k^{(j_1)}$, $Y_2(j_1) = \sum_{j_2=0}^{k-j_1} C_{k-j_1}^{(j_2)}$, дополним транспонированными слоями второй части массива $j_1 \to\to (k-j_1), j_1 \in \left[0, \left[\frac{k}{2}\right]\right] \subset Z^1$, суперпозируя последние вертикально вниз от общего уровня $2^k + 1$. Оценка $2^{k-j_1} + 2^{j_1} \leqslant 2^k + 1$, $j_1 \in \left[0, \left[\frac{k}{2}\right]\right] \subset Z^1$ суммы первоначального и транспонированного слоев $2^{k-j_1} = \sum_{j_2=0}^{k-j_1} C_{k-j_1}^{(j_2)}$, $2^{j_1} = \sum_{j_2=0}^{j_1} C_{j_1}^{(j_2)}$ показывает, что данные слои суперпозируются аддитивно с возможными пустотами. В итоге, для верхней части массива $j_2 \in [0,\ k-j_1] \subset Z^1$ получаем ресурсную оболочку $0 \leqslant Y_1 \leqslant \sum_{j_1=0}^{\left[\frac{k}{2}\right]} C_k^{(j_1)}$, $0 \leqslant Y_2 \leqslant 2^k + 1$ (рис. 9).

Нижняя часть массива $(-j_2) \in$ $\in [0,\ k-j_1] \subset Z^1$ в силу симметрии с верхней частью имеет идентичную ресурсную оболочку. Суперпозируя последнюю вправо о предыдущей, получаем окончательно ресурсную оболочку $2 \sum_{j_1=0}^{\left[\frac{k}{2}\right]} C_k^{(j_1)} \times (2^k + 1) \approx$ $\approx 2^k \times 2^k$ для полного модельного параболического массива умножения измерений (рис. 10).

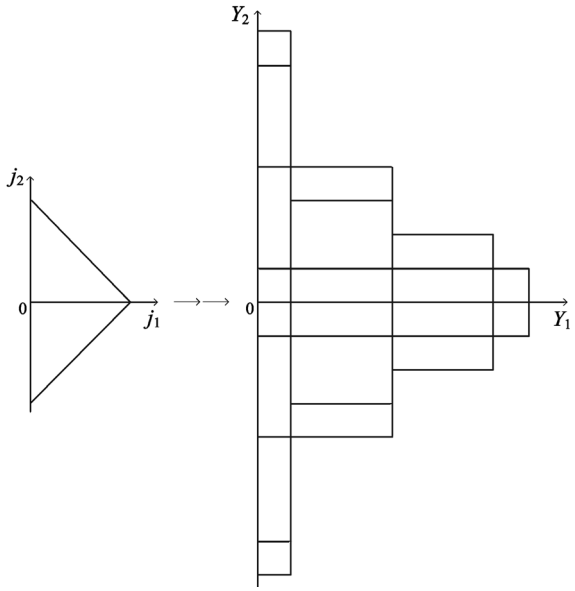


Рис. 8. Массив модельного параллельного спроса

Если $M = 2^k$, то изложенный алгоритм определяет канонический эксперимент ресурсных распределений в рамку $M \times M$ операционного поля МВС.

Альтернативный эксперимент ресурсных распределений модельного параллельного спроса параболического типа (10) заключается в аддитивной графике вертикальных слоев $\bigcup_{j_1=0}^{k} \left[C_k^{(j_1)} \times \bigcup_{j_2=0}^{k-j_1} C_{k-j_1}^{(j_2)} \right]$, $j_2 \in [0,\ k-j_1] \subset Z^1, j_1 \in [0,\ k] \subset Z^1$ ближнего координатного треугольника $0 \leqslant Y_1 + Y_2 \leqslant 2^k$, $2^k = M$, с последующим заполнением дальнего треуголь-

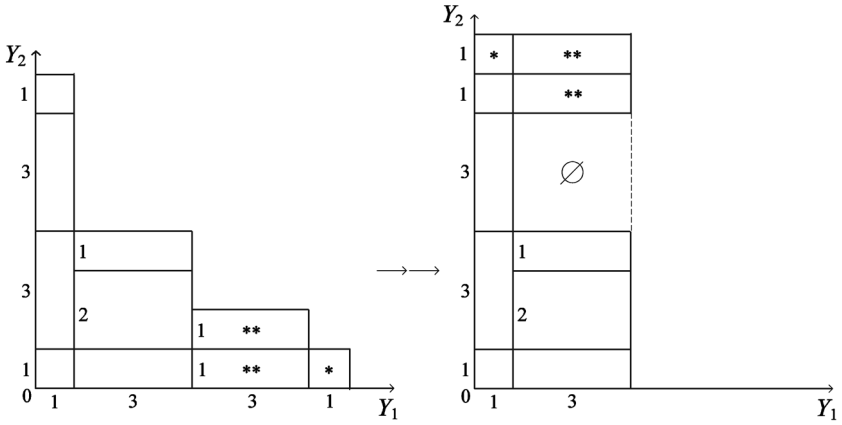


Рис. 9. Метод составных вертикальных слоев при $k = 3$

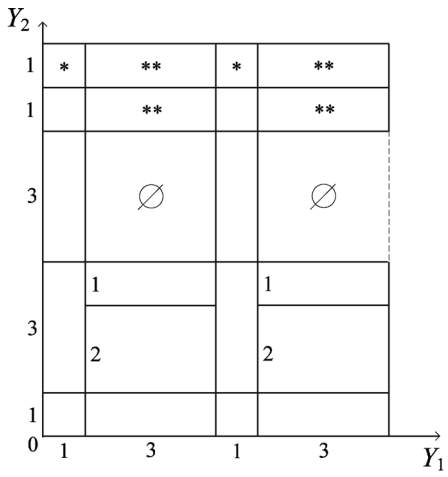


Рис. 10. Модельное параболическое ресурсное распределение

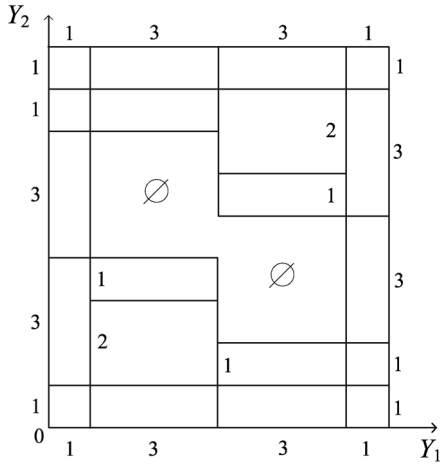


Рис. 11. Ресурсное распределение параболического типа с транспонированием

ника рамки $M \times M$ транспонированием нижней части массива спроса (рис. 11).

Заключение

Решение сформулированной во введении задачи локализации линейно протяженного массива ресурсных прямоугольников в квадратную рамку поля распределения ресурсов вычислительной системы увязывается в работе с предварительной типизацией массива по квадратичному типу и другим свойствам. Приводятся кольцевой, угловой и матричный алгоритмы локализации с необходимыми оценками и базовыми примерами. В случае параболического массива предложена модельная локализация массива умножения измерений факторов бисочетания. Указанные алгоритмы могут использоваться в диспетчере операционной системы как МВС, так и центра Grid-технологий.

Список литературы

1. **Барский А. Б.** Параллельные информационные технологии. М.: ИНТУИТ; БИНОМ. Лаборатория знаний, 2007. 503 с.
2. **Хорошевский В. Г.** Архитектура вычислительных систем. М.: Изд-во МГТУ им. Н. Э. Баумана, 2005. 512 с.
3. **Воеводин В. В., Воеводин Вл. В.** Параллельные вычисления. СПб.: БХВ-Петербург, 2002. 608 с.
4. **Барский А. Б.** Параллельные информационные технологии в основе Gird-системы // Информационные технологии. 2006. № 12. С. 54—60.
5. **Каляев И. А., Левин И. И., Семерников Е. А., Шмойлов В. И.** Реконфигурируемые мультиконвейерные вычислительные структуры / Изд. 2-е, перераб. и доп. / Под общ. ред. И. А. Каляева. Ростов н/Д: Изд-во ЮНЦ РАН, 2009. 344 с.
6. **Бакенрот В. Ю., Чефранов А. Г.** Эффективность приближенных алгоритмов распределения программ в однородной вычислительной системе // Изв. АН СССР. Техн. кибернетика, 1985. № 4. С. 135—148.
7. **Саак А. Э.** Локально-оптимальный синтез расписаний для Grid-технологий // Информационные технологии. 2010. № 12. С. 16—20.

УДК 621.395(075)

Т. М. Мансуров, д-р техн. наук, проф.,
И. А. Мамедов, канд. техн. наук, доц.,
Э. Т. Мансуров, аспирант,
Азербайджанский технический университет,
г. Баку, Азербайджан,
e-mail: tofiq-mansurov@rambler.ru

Разработка методики определения длины регенерационного участка *x*DSL-модемов сети абонентского доступа

Разработана методика определения предельной длины регенерационного участка xDSL-модемов сети абонентского доступа в зависимости от суммарной помехи и от переходных затуханий при параллельной работе на одном кабеле двух разнотипных модемов, каждый из которых работает по отдельной паре в однополосном дуплексном режиме. Полученная методика пригодна также для вариантов построения сети с произвольным числом параллельно работающих модемов с цифровыми линейными сигналами, которые имеют произвольные скорости передачи и используют однополосную дуплексную передачу по одной или более кабельным парам.

Ключевые слова: *xDSL-модем, абонентский доступ, симметричный и асимметричный доступ, регенерационный участок, направляющая система*

Введение

DSL-технология является достаточно новой технологией, позволяющей значительно расширить полосу пропускания действующих медных телефонных линий, соединяющих телефонные станции с индивидуальными пользователями. Любой абонент, пользующийся в настоящий момент обычной телефонной связью, имеет возможность с помощью технологии *x*DSL значительно увеличить скорость пользования услугами связи. Для организации линии DSL используются существующие линии телекоммуникационной сети (ТС) и данная технология хороша и тем, что не требует прокладывания дополнительных телефонных пар. Благодаря многообразию версий технологии *x*DSL пользователь может выбрать подходящую именно ему скорость передачи данных — от 32 кбит/с до более чем 50 Мбит/с. Данные технологии позволяют также использовать обычную телефонную линию для широкополосных систем. Современные технологии *x*DSL дают возможность организации высокоскоростного доступа Интернет в каждый дом или на каждое предприятие среднего и малого бизнеса, превращая обычные телефонные кабели в высокоскоростные цифровые каналы. Причем скорость передачи данных зависит только от качества и протяженности линии, соединяющей пользователя и провайдера. При этом провайдеры обычно дают возможность пользователю самому выбрать скорость передачи, наиболее соответствующую его индивидуальным потребностям.

Одной из основных проблем организации функционирования технологии *x*DSL является проблема организации абонентского доступа к сетевым услугам. Актуальность этой проблемы, в первую очередь, обусловлена бурным развитием сети Интернет, доступ к которой требует резкого увеличения эффективности и пропускной способности систем абонентского доступа. Несмотря на появление новых самых современных волоконно-оптических и беспроводных методов абонентского доступа основным средством сети доступа остаются традиционные медные двухпроводные абонентские линии многопарных кабелей сетей абонентского доступа (САД). Поэтому в настоящее время и в обозримом будущем стратегическим направлением повышения эффективности САД будет оставаться семейство *x*DSL-модемов [1, 2]. Эти модемы обеспечивают симметричную дуплексную передачу встречных цифровых потоков по двухпроводной сети абонентского доступа и представляют наибольший интерес для провайдеров телекоммуникационных услуг, производителей оборудования, пользователей корпоративного и частного секторов. Практическая апробация *x*DSL-модемов подтверждает то, что существующая сеть медных кабелей связи еще долго останется той основой, на которой строится телекоммуникационная инфраструктура.

Постановка задачи

Разработка методики определения длины регенерационного участка *x*DSL-модемов САД, организованного по низкочастотным симметричным телефонным кабелям абонентских линий, связана с анализом электрических параметров многопарных кабелей САД, классификацией основных технологий *x*DSL-модемов симметричного и асимметричного цифрового абонентского доступа, анализом технологий кодирования и требований к цифровым линейным сигналам (ЦЛС), наиболее перспективным в практической реализации, исследованием зависимости длины регенерационного участка *x*DSL-модемов симметричного доступа в зависимости от скорости и типа направляющих систем. Поэтому разработка методики определения предельной длины регенерационного участка и защищенности в зависимости от влияния сум-

марной помехи и от переходных влияний является актуальной задачей теоретического и практического характера.

Определение защищенности *x*DSL-модема от суммарной помехи

Проанализируем факторы, определяющие допустимую длину регенерационного участка *x*DSL-модемов. Известно, что длина регенерационного участка зависит от защищенности *x*DSL-модема от суммарной помехи, действующей на входе решающего устройства (РУ) [1, 3]. Полагая, что все шумы и помехи не зависят друг от друга, а мощность суммарной помехи равна сумме мощностей помех от отдельных источников, ожидаемую суммарную помехозащищенность $A_{з\Sigma}$ будем определять по следующей формуле [1, 2]:

$$A_{з\Sigma} = -10\lg\left[\sum_{i=1}^{N} U_{пi}^2/A_р^2\right] = -10\lg\left[\sum_{i=1}^{N} P_{пi}R_р/A_р^2\right] = \\ = -10\lg\left[\sum_{i=1}^{N} \mathrm{dec}(-0{,}1A_{зi})\right], \qquad (1)$$

где $\mathrm{dec}(x) \Rightarrow 10^x$; N — число источников помех; $U_{пi}$ и $P_{пi}$ — действующее напряжение и мощность i-й помехи на входе РУ; $A_р$ — амплитуда импульса сигнала на входе РУ; $R_р$ — входное сопротивление РУ; $A_{зi}$ — защищенность модема от i-го источника шума на входе РУ.

При этом мощность i-й помехи на входе РУ определяется выражением

$$P_{пi} = \int_0^{f_т} G_{пi}(f)df = \int_0^{f_т} G_{п.рi}(f)K_{у.к}(f)df, \qquad (2)$$

где $G_{п.рi}(f)df$ и $G_{пi}(f)df$ — энергетические спектры i-й помехи соответственно на входе и на входе РУ регенератора; $K_{у.к}$ — коэффициент передачи корректирующего усилителя (УК); $f_т$ — тактовая частота.

Соответственно выражение для защищенности модема от i-й помехи имеет вид

$$A_{зi} = -10\lg(P_{п1}R_р/A_р^2) = \\ = -10\lg\left\{R_р A_р^{-2}\int_0^{f_т} G_{пi}(f)df\right\}. \qquad (3)$$

При определении защищенности модема от i-й помехи должно быть известно значение коэффициента передачи УК по мощности $K_{у.м}$ и энергетический спектр i-й помехи. Коэффициент передачи УК выбирается в зависимости от степени искажения символов ЦЛС на выходе регенератора и соответствия формы символов на входе РУ критерию Найквиста, т. е. критерию отсутствия межсимвольных искажений. Если $U_{вх}(t)$ определяет форму одиночных символов ЦЛС на входе РУ, а $U_{вых}(t)$ — на выходе регенератора, то

$$K_{у.м}(f) = \frac{R_л}{R_р}\frac{\left|F[U_{вх}(t)]\right|^2}{\left|F[U_{вых}(t)]\right|^2 K_л^2(f)}, \qquad (4)$$

где $|F[U_{вх}(t)]|$ и $|F[U_{вых}(t)]|$ — прямое преобразование Фурье от функций $U_{вх}(t)$ и $U_{вых}(t)$; $R_л$ — волновое сопротивление ЛС; $K_л(f)$ — коэффициент передачи линии связи (ЛС) по напряжению.

Основными источниками помех в кабельных линиях связи являются помехи типа собственных шумов, обусловленные тепловыми шумами прилегающего регенерационного участка линии связи и шумами линейной части регенератора до входа решающего устройства, помехи от переходных влияний (ПВ) и помехи от межсимвольных искажений.

Анализ защищенности от переходных влияний

При анализе защищенности от ПВ между однотипными *x*DSL-модемами можно считать, что энергетический спектр помехи на выходе регенератора влияющей цепи равен энергетическому спектру сигнала $G_с(f)$ на выходе регенератора це-

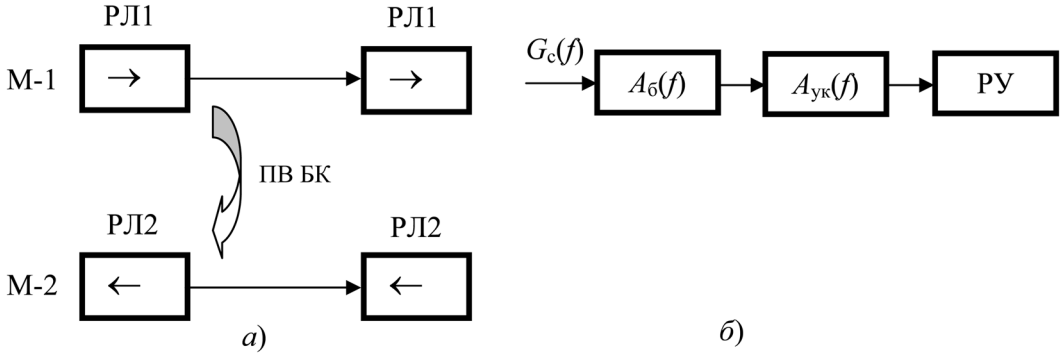


Рис. 1. Переходные влияния при работе двух модемов на встречных направлениях:
а — схема переходных влияний; *б* — структурная схема прохождения

пи, подверженной влиянию. Энергетический спектр сигнала, в свою очередь, определяется типом линейного кода передачи [2, 4].

Если по двум соседним парам ЦЛС передаются во встречных направлениях, то определяющими являются переходные влияния на ближний конец (БК) (рис. 1, *а*, где М-1 — влияющий, а М-2 — подверженный влиянию модем). Модель канала передачи, по которому помеха, обусловленная ПВ на БК, попадает на вход РУ М-2, приведена на рис. 1, *б*.

Здесь каждый спектральный компонент мощности мешающего сигнала на выходе РЛ1 попадает на вход регенератора РЛ2 с затуханием $A_б(f)$, которое называется переходным затуханием на БК. Затем переходная помеха проходит через УК РЛ2 и попадает на вход РУ. В этом случае энергетический спектр помехи на входе решающего устройства определяется следующим образом:

$$G_п^{(бк)}(f) = G_с(f)\mathrm{dec}[-0{,}1A_б(f)]K_{у.к}(f), \qquad (5)$$

где $G_с(f)$ — энергетический спектр сигнала на выходе регенератора влияющей цепи.

В выражении (5) переходное затухание на БК определяется как

$$A_б(f) = A_б(f_1) - 15\lg(f/f_1), \qquad (6)$$

где $A_б(f_1)$ — известное переходное затухание на БК на частоте f_1.

Тогда защищенность от ПВ на БК определяется по (3) с учетом (4), (5) и (6) следующей формулой:

$$A_{з.б} = -10\lg\{R_р/A_р^2)\mathrm{dec}[-0{,}1A_{з.б}(f_1)] \times \\ \times \int_0^{f_т} G_с(f)K_{у.к}(f)(f/f_1)^{1,5}df\}. \qquad (7)$$

Если по двум соседним парам ЦЛС передаются на совпадающих направлениях (рис. 2, *а*), то необходимо учитывать помеху как от ПВ на БК, так и от ПВ на дальний конец (ДК). Модель канала прохождения помехи на вход РУ модема, подверженного влиянию, приведены на рис.2, *б*.

Здесь блок 1 с затуханием, равным переходной защищенности (ПЗ) участка на ДК $A_д(f, l)$, характеризует частотную зависимость ПВ на ДК для ЛС длиной l, а блок 5 с затуханием, равным ПЗ на БК $A_б(f)$ — частотную зависимость ПВ на БК. Поскольку передатчик РЛ2 работает в ключевом режиме, его выходное сопротивление не согласовано с сопротивлением линии. Поэтому полагаем, что коэффициент отражения помехи от ПВ на БК с выхода передатчика РЛ2 практически равен 1,0 ($K_{отр} \approx 0$ дБ) и помеха начинает распространяться по ЛС (блок 7) в том же направлении, что и полезный сигнал. Блок 4 — некоторый условный сумматор, который учитывает сложение мощностей помех от переходных влияний на БК и ДК. Далее суммарная помеха поступает на вход решающего устройства РЛ2 через УК с коэффициентом передачи $K_{у.к}(f)$.

На основании представленной модели канала прохождения переходной помехи определим ее энергетический спектр на входе РУ М-2 в виде:

$$G_п^{(дк)}(f) = G_с(f)K_{у.к}(f)\{\mathrm{dec}[-0{,}1A_б(f) + \\ + a_{лт}\sqrt{f/f_т}\,] + \mathrm{dec}[-0{,}1A_д(f, l_р)]\}, \qquad (8)$$

где $a_{лт} = \alpha(f_т)l_р$ и $\alpha(f_т)$ — затухание ЛС длиной $l_р$ на 1,0 км на тактовой частоте ЦЛС.

В выражении (8) ПЗ на ДК $A_д(f, l_р)$ определяется следующим образом:

$$A_д(f, l_р) = A_{з.лд}(f, l_р) + \alpha(f)l_р,$$

а

$$A_{з.лд}(f, l_р) = \\ = A_{з.лд}(f_1, l_1) - 10\lg(l_р/l_1) - m10\lg(f/f_1), \qquad (9)$$

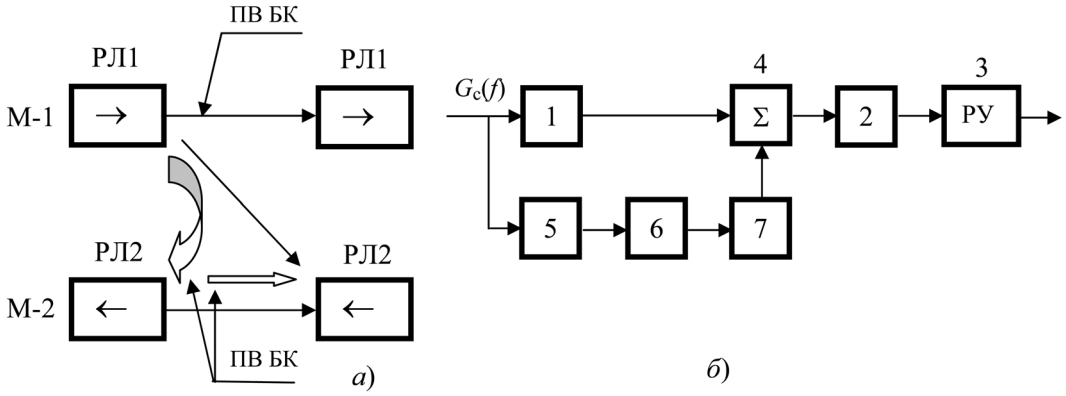


Рис. 2. Переходные влияния при работе двух модемов на совпадающих направлениях:
а — механизм переходных влияний; *б* — структурная схема прохождения переходной помехи

где $A_{з.лд}(f_1, l_1)$ — известная защищенность кабельной пары (не системы) от ПВ на ДК для длины l_1 на частоте f_1; защищенность модема от ПВ при работе на совпадающих направлениях определяется по (3) с учетом (4), (8) и (9):

$$A_{з.б} = -10\lg\left\{R_р A_р^{-2}\int_0^{f_т} G_с(f)\,\mathrm{dec}[-0{,}1a_{лт}\sqrt{f/f_т}\,] \times \right.$$
$$\times \{(l_р/l_1)(f/f_1)^m \mathrm{dec}[-0{,}1A_{з.лд}(f_1, l_1)] +$$
$$\left. + (f/f_1)^{1,5}\mathrm{dec}[-0{,}1A_б(f_1)]\}df\right\}. \qquad (10)$$

При анализе защищенности от собственных шумов энергетический спектр помехи на входе РУ определяется соотношением

$$G_п^{(ш)}(f) = kTD_{ш.у}(f)K_{у.м}(f), \qquad (11)$$

где $D_{ш.у}$ — динамический диапазон шума.

Если $D_{ш.у}(f) = D_{ш.у} = \mathrm{const}$, выражение для защищенности модема от собственных шумов на основании (3) и (11) имеет вид:

$$A_{з.ш} = -10\lg\left[kTD_{ш.у}R_р A_р^{-2}\int_0^{f_т} K_{у.к}(f)df\right]. \qquad (12)$$

Результирующая защищенность модема $A_{зΣ}$ определяется по (1) с учетом того, что по каждой паре передаются ЦЛС во встречных направлениях и, следовательно, в каждом направлении по каждой паре надо учитывать ПВ от соседних пар как на БК, так и на ДК. При этом

$$A_{зΣ} = -10\lg\left\{\sum_{i=1}^{N_1}\mathrm{dec}(-0{,}1A_{з.бi}) + \right.$$
$$\left. + \sum_{i=1}^{N_2}\mathrm{dec}(-0{,}1A_{з.дi}) + \mathrm{dec}(-0{,}1A_{з.ш})\right\}, \qquad (13)$$

где N_1 и N_2 — число пар (систем), оказывающих переходные влияния на БК и ДК соответственно.

Если в (13) подставить (8), (10) и (12) и учесть в них соответствующие значения $K_{у.к}(f)$ и $G_с(f)$, можно определить защищенность для модема любой технологии.

В качестве примера рассмотрим применение разработанной методики для анализа модема технологий HDSL и SDSL. В данном случае при использовании одного кабеля и K одновременно работающих модемов технологии HDSL (задействовано $2K$ пар кабеля) в (13) $N_2 = 2K - 1$. При этом $N_1 = N_2 - 1$ — для первого варианта построения эхокомпенсатора и $N_1 = N_2$ — для второго варианта [4]. Для K одновременно работающих модемов технологии SDSL (задействовано K пар кабеля) в (13) $N_1 = N_2 = K - 1$.

Если в качестве линейного кода в указанных модемах используется код 2B1Q, одиночные символы которого имеют треугольную форму и длительность, равную тактовому интервалу, то в выражении (4) преобразование $F[U_{вых}(t)]$ имеет следующий вид:

$$F[U_{вых}(t)] = \frac{A_с}{2f_т}\frac{\sin^2(\pi f/2f_т)}{(\pi f/2f_т)^2}, \qquad (14)$$

где $A_с$ — амплитуда символов ЦЛС на выходе регенератора.

Если на входе РУ символы имеют форму, близкую к косинусоидальной, то спектральная плотность записывается выражением

$$F[U_{вх}(t)] = (A_р/F_т)[\cos(\pi f/2f_т)^2]. \qquad (15)$$

При этом выражение для $K_{у.к}(f)$ принимает вид:

$$K_{у.к}(f) = 4\frac{A_р^2/R_р}{A_с^2/R_л}\frac{[\cos(\pi f/2f_т)]^4}{[\sin(\pi f/2f_т)/(\pi f/2f_т)]^4} \times$$
$$\times \mathrm{dec}(0,\ 1a_{лт}\sqrt{f/f_т}\,). \qquad (16)$$

Энергетический спектр сигнала определяется его статистическими характеристиками. Можно показать, что математическое ожидание ЦЛС в коде 2B1Q равно нулю, следовательно, в спектре отсутствует постоянная составляющая [2]. А так как не исключена вероятность появления нескольких последовательных символов с одинаковой полярностью, то максимум энергетического спектра должен быть смещен в область более низких частот, чем, например, у кода AMI. С достаточной степенью точности энергетический спектр сигнала в коде 2B1Q можно определить следующим выражением:

$$G_с(f) = 2{,}27G_0\sqrt{f/f_т}\cos^2(\pi f/2f_т), \qquad (17)$$

где G_0 — некоторая постоянная, определяемая из следующих соображений.

С одной стороны, мощность сигнала определяется как

$$P_с^* = \int_0^{f_т} G_с(f)df = 0{,}574G_0f_т.$$

С другой стороны, она равна мощности усредненной по времени реализации случайного процесса с указанной выше формой символов:

$$P_с^{**} = \lim_{T\to\infty}\frac{1}{T}\int_0^T\frac{U_с^2(t)}{R_л}dt = \frac{10}{54}\frac{A_с^2}{R_л}.$$

Значения поправочных коэффициентов в зависимости от затухания линейного тракта

$a_{лт}$, дБ	10	20	30	40	50	60	70	80	90	100
$J_1(a_{лт})$	0,755	1,175	1,185	1,056	0,901	0,758	0,637	0,536	0,454	0,387
$J_2(a_{лт})$	0,08	0,149	0,166	0,155	0,134	0,111	0,091	0,073	0,059	0,048

Определив G_0 из условия $P_c^* = P_c^{**}$ и подставив его в (17), получим окончательно

$$G_c(f) = \frac{0{,}733A_c^2}{R_{лт}f_т}\sqrt{f/f_т}\cos^2(\pi f/2f_т). \qquad (18)$$

Используя (16) и (18), общие выражения (7), (10) и (12) можно привести к виду:

$$A_{з.ш} = -10\lg\{4kTD_{ш.у}f_т R_л A_c^{-2} \times \times \operatorname{dec}(0{,}1a_{лт})(0{,}23a_{лт})^{-2}J_1(a_{лт})\}, \qquad (19)$$

$$A_{з.б} = -10\lg\{2(f_т/f_1)^{1,5}\operatorname{dec}[-0{,}1A_б(f_1)]J_2(a_{лт}) \times \times \operatorname{dec}(0{,}1a_{лт})/(0{,}23a_{лт})^2\}; \qquad (20)$$

$$A_{з.д} = -10\lg\{2(f_т/f_1)^{1,5}J_3\operatorname{dec}[-0{,}1A_б(f)] + + 2(f_т/f_1)^m(l_р/l_1)J_4\operatorname{dec}[-0{,}1A_{з.лд}(f_1, l_1)]\}. \qquad (21)$$

Здесь $J_1(a_{лт})$ и $J_2(a_{лт})$ — поправочные коэффициенты, зависящие от $a_{лт}$ (см. таблицу), определяемые методами численного интегрирования из выражений

$$J_1(a_{лт}) = \frac{(0{,}23a_{лт})^2}{\operatorname{dec}(0{,}1a_{лт})}\int_0^1(\pi/2)^4x^4\frac{\cos^4(\pi x/2)}{[\sin(\pi x/2)]^4} \times \times \operatorname{dec}(0{,}1a_{лт}\sqrt{x})dx;$$

$$J_2(a_{лт}) = 1{,}465\frac{(0{,}23a_{лт})^2}{\operatorname{dec}(0{,}1a_{лт})}\int_0^1(\pi/2)^4x^6\frac{\cos^6(\pi x/2)}{\sin(\pi x/2)} \times \times \operatorname{dec}(0{,}1a_{лт}\sqrt{x})dx;$$

J_3, J_4 — постоянные поправочные коэффициенты, определяемые из выражений

$$J_3 = 1{,}465\int_0^1(\pi/2)^4x^6\frac{\cos^6(\pi x/2)}{\sin^4(\pi x/2)}dx;$$

$$J_4 = 1{,}465\int_0^1(\pi/2)^4x^{m+4,5}\frac{\cos^6(\pi x/2)}{\sin^4(\pi x/2)}dx,$$

причем $J_3 = 0{,}035$, а $J_4 = 0{,}022$ при учете межпарных влияний в кабельной линии ($m = 2$) или $J_4 = 0{,}044$ при учете внутренних влияний ($m = 4$).

Определение предельной длины регенерационного участка

Предельная длина регенерационного участка определяется из условия равенства ожидаемой и допустимой защищенности модемов линейного регенератора [3, 4].

Допустимая защищенность для линейного кода 2B1Q, являющегося одним из видов многоуровневых кодов, определяется выражением:

$$A_{з.доп} = 10{,}65 + 11{,}42\lg\{-\lg(K_{ош1доп}l_р)\} + + 20\lg[(n-1)/2]. \qquad (22)$$

Численные расчеты проводились для сигнала в коде 2B1Q с тактовыми частотами 784 кГц (для скорости 1168 кбит/с) и 1160 кГц (для скорости 2320 кбит/с) для кабеля КСПП-1 × 4 × 1,2, имеющего следующие параметры: коэффициент затухания на частоте f_1 = 1024 кГц; $\alpha(f_1)$ = 8,0 дБ/км;

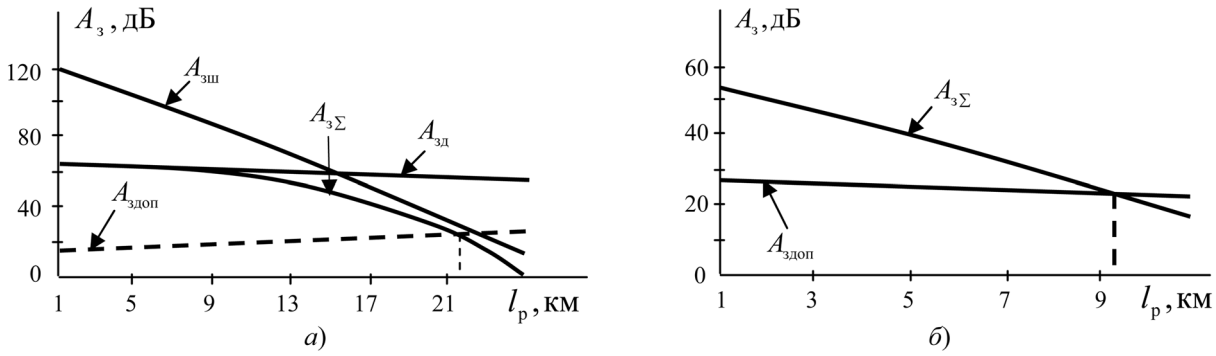


Рис. 3. Расчетные зависимости отдельных составляющих помехозащищенности модемов по технологии HDSL и сложной схемы эхокомпенсации:
а — для одного модема; *б* — для двух модемов

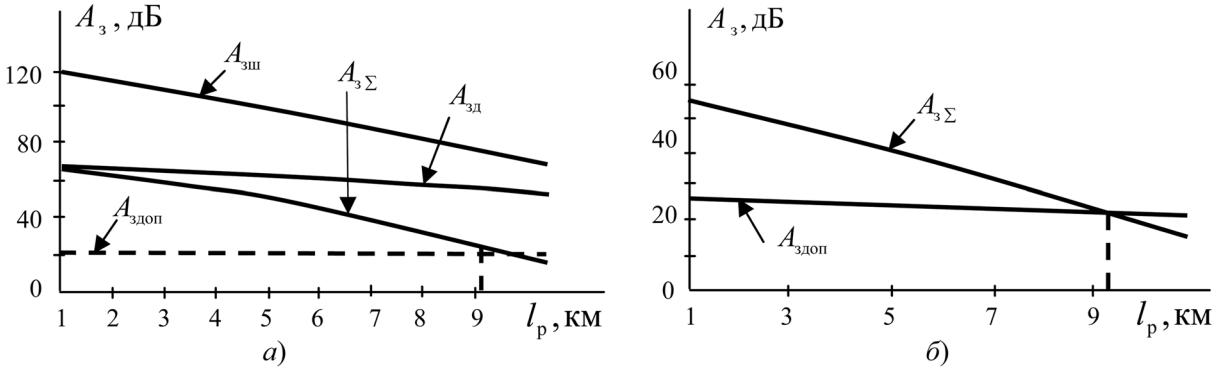


Рис. 4. Расчетные зависимости отдельных составляющих помехозащищенности модемов по технологии HDSL и простой схемы эхокомпенсации:
а — для одного модема; *б* — для двух модемов

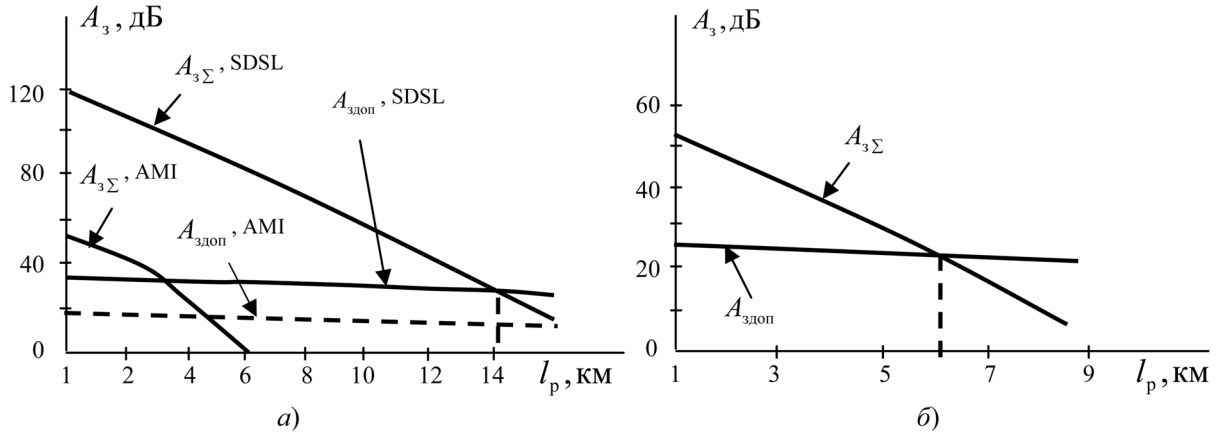


Рис. 5. Расчетные зависимости отдельных составляющих помехозащищенности модемов по технологии SDSL:
а — для одного модема; *б* — для двух модемов

переходное затухание на БК на частоте $f_2 = 512$ кГц; $A_б(f_2) = 56$ дБ; защищенность от переходных влияний на ДК на частоте $f_1 = 1024$ кГц для кабеля длиной $l_1 = 0{,}75$ км: $A_{з.лд}(f_1, l_1) = 45$ дБ. Перерасчет значений для других частот проводился по следующим формулам:

$$\alpha(f) = \alpha(f_1)\sqrt{f/f_1}\,;\; A_б(f_{1'}) = A_б(f_{2'}) - 15\lg(f_1/f_2). \quad (23)$$

Предельная длина регенерационного участка определялась графоаналитическим способом в точке пересечения расчетных зависимостей: допустимой защищенности — из выражения (22) и суммарной ожидаемой защищенности — из (13), (19), (20) и (21) — в функции от длины участка $l_р$.

Расчетные зависимости отдельных составляющих помехозащищенности модемов по технологии HDSL для первого и второго вариантов эхокомпенсации приведены на рис. 3 и 4 соответственно, а для модемов по технологии SDSL — на рис. 5. При этом рис. 2, *а*—4, *а* соответствуют работе по одному кабелю только одной цифровой системы передачи (ЦСП), вариант рис. 2, *б*—4,*б* — одновременной работе двух ЦСП.

Заключение

Разработанная методика позволяет выполнить расчет предельной длины регенерационного участка в зависимости от суммарной помехи и от переходных затуханий при параллельной работе на одном кабеле двух разнотипных модемов, каждый из которых работает по отдельной паре в однополосном дуплексном режиме. Данная методика пригодна также для вариантов построения сети с произвольным числом параллельно работающих модемов с ЦЛС, которые имеют произвольные скорости передачи и используют однополосную дуплексную передачу по одной или более кабельным парам.

Список литературы

1. **Парфенов Ю. А., Мирошников Д. Г.** Цифровые сети доступа. Медные кабели и оборудование. М.: Эко-Трендз, 2005. 288 с.
2. **Гордиенко В. Н., Тверецкий М. С.** Многоканальные телекоммуникационные системы. М.: Горячая линия—Телеком, 2005. 416 с.
3. **Павличенко Ю. А., Пашолок П. А., Антыков В. В.** Цифровой абонентский доступ. Одесса: ОНАС им. А. С. Попова, 2001. 92 с.
4. **Крендзель А. В., Соколов Н. А.** Сети абонентского доступа: структурные характеристики // Электросвязь. 1997. № 11. С. 13—15.

УДК 004.3.053

Н. И. Червяков, д-р техн. наук, проф., зав. каф.,
М. Г. Бабенко, аспирант,
Ставропольский государственный университет
e-mail: whbear@yandex.ru

Пороговая схема разделения секрета на эллиптической кривой

Предлагается совершенная схема порогового разделения секрета на эллиптической кривой над Z_q, где $q = \prod_{i=1}^{s} p_i$, p_i — попарно различные простые числа и $p_i > 3$ для всех $i = 1...s$.

Ключевые слова: *эллиптическая кривая, эллиптическая криптография, пороговые схемы разделения секрета на эллиптической кривой*

Введение. Постановка задачи

Современные информационные системы требуют особого подхода к сохранению секрета: люди не могут целиком доверять своему окружению, так как кто-то может быть подкуплен конкурентами. В этом случае проблему сохранения секрета в тайне решают с помощью схемы разделения секрета [1]. Первыми пороговую схему разделения секрета рассмотрели в своих работах Г. Блейкли [2] и А. Шамир [3] в 1979 г.

Схема Шамира позволяет секрет s_0 разделить на n частей $s_{0,1}$, $s_{0,2}$, ..., $s_{0,n}$ так, что выполняются следующие два условия:

- с помощью любых t и более частей $s_{0,i}$ можно восстановить секрет s_0;
- с помощью любых $t - 1$ частей нельзя ни восстановить секрет, ни получить о нем никакой дополнительной информации, кроме той, которая уже имеется.

Пороговые схемы, удовлетворяющие второму условию, называются *совершенными пороговыми схемами разделения секрета*. В работе [4] предложен алгоритм многоточечного разделения точек на эллиптической кривой, в котором используется спаривание из работы [5]. В работе [4] показано, что эта схема является совершенной и криптостойкой, если выполняются следующие четыре условия:

1) честных пользователей в схеме должно быть не менее $n - t + 1$;

2) задача нахождения дискретного логарифма в группе точек эллиптической кривой $E(GF(q))$ должна быть трудно вычислимой.

Примечание. $E(GF(q))$ — множество точек эллиптической кривой, заданной уравнением $y^2 = x^3 + ax + b$ над $GF(q)$, включая точку бесконечности O;

3) задача вычисления дискретного логарифма в мультипликативной группе $GF(\#E(GF(q)))$ должна быть трудно вычислимой (# — мощность множества);

4) распространено не более $O(\lg(q))$ точек.

Из условия 3 и условия применимости алгоритма Полига—Хеллмана—Зильбера, изложенного в работах [6—8], следует, что порядок группы точек эллиптической кривой $E(GF(q))$ при разложении на множители должен содержать большое простое число. Таким образом, нам необходим алгоритм, с помощью которого можно эффективно находить порядок группы точек эллиптической кривой. Одним из наиболее известных алгоритмов является алгоритм Шуфа, изложенный в работах [9—10], и его усовершенствование Элкисаном и Аткиным, известное как алгоритм SEA [11—13]. Сложность этих алгоритмов составляет $O(\log^8 q)$ и $O(\log^6 q)$ соответственно. Для выполнения условия 2 требуется, чтобы $q \geqslant 2^{255}$, из чего следует, что, используя самый быстрый универсальный алгоритм SEA для нахождения $\#E(GF(q))$, требуется 255^6 операций, что приблизительно равно $2 \cdot 10^{15}$, а так как современная вычислительная техника может выполнять 10^{10} операций в секунду, то потребуется 55 ч для нахождения мощности множества одной эллиптической кривой. Это делает данный алгоритм неприемлемым для использования в практических целях.

Альтернативным решением данной проблемы является построение криптосистемы на эллиптической кривой E, заданной уравнением в форме Вейерштрассе $y^2 = x^3 + ax + b$, над Z_q, где $a, b \in Z_q$, $q = \prod_{i=1}^{s} p_i$, p_i — различные простые числа и для всех $i = 1,..., s$ выполняется условие $3 < p_i < 10^6$. В этом случае можно использовать следующий алгоритм для нахождения мощности множества точек эллиптической кривой над $E(Z_q)$.

Алгоритм нахождения мощности множества точек на эллиптической кривой

Рассмотрим сравнение

$$y^2 \equiv x^3 + ax + b (\mathrm{mod}\, p_i). \quad (1)$$

1. Найдем число решений n_i сравнения (1), используя формулу

$$n_i = p_i + \sum_{x \in F_{p_i}} \left(\frac{x^3 + ax + b}{p_i} \right),$$

где $\left(\frac{x^3 + ax + b}{p_i} \right)$ — символ Лежандра.

2. Вычислим порядок по формуле

$$\#E(Z_q) = \prod_{i=1}^{s} n_i + 1. \quad (2)$$

Для построения q используем первые 200 простых чисел, начиная с пяти. Получим q, приблизительно равное 10^{519}. Предложенный выше алгоритм позволяет вычислить мощность множества точек одной эллиптической кривой в среднем за 2,4 мс для указанного q. Следовательно, данный алгоритм, применяемый при использовании небольших простых чисел p_i, является более эффективным, чем алгоритм SEA.

Поставим задачи:

1. Разработать криптосистему многоточечного разделения секрета на эллиптической кривой над Z_q.

2. Проанализировать ее безопасность.

Билинейное спаривание

Для построения схемы разделения секрета введем операцию спаривания на эллиптической кривой, аналогичную рассмотренной в работе [5]. Для любой точки $P \in E(GF(p_i))$ выполняется равенство $(n_i + 1)P = O$, где O — точка в бесконечности. Обозначим $E(\overline{Z}_q)$ множество точек эллиптической кривой, которая задана уравнением $y^2 = x^3 + ax + b$, над полями $GF(p_i^k)$. Для построения спаривания важно знать структуру группы точек эллиптической кривой. Ниже приведена теорема, описывающая структуру группы, когда эллиптическая кривая задана над полем $GF(p_i^k)$, где p_i — простое число, $k \geqslant 1$ и порядок группы точек эллиптической кривой выражается формулой $\#E(GF(p_i^k)) = p_i^k + 1 - t$.

Теорема 1 [14]. Если $t^2 = p_i^k$, $2p_i^k$, $3p_i^k$, то группа — циклическая. Если $t^2 = 4p_i^k$, то группа изоморфна $Z_{\sqrt{p_i^k} - 1} \oplus Z_{\sqrt{p_i^k} - 1}$ в случае $t = 2\sqrt{p_i^k}$, или изоморфна $Z_{\sqrt{p_i^k} + 1} \oplus Z_{\sqrt{p_i^k} + 1}$ в случае $t = -2\sqrt{p_i^k}$. Если $t = 0$, $p_i^k \neq 3 (\mathrm{mod}\, 4)$, то группа циклическая. Если $t - 0$, $p_i^k \equiv 3 (\mathrm{mod}\, 4)$, то группа или циклическая или изоморфна $Z_{\frac{p_i^k + 1}{2}} \oplus Z_2$.

Из теоремы 1 следует, что в случае, если $t^2 = 4p_i^k$ и r — четное число, группа точек $E(GF(p_i^k))$ представляется в виде прямой суммы $Z_{\sqrt{p_i^k} - 1} \oplus Z_{\sqrt{p_i^k} - 1}$ или $Z_{\sqrt{p_i^k} + 1} \oplus Z_{\sqrt{p_i^k} + 1}$. Обозначим эти группы $E[l_i]$, где $l_i = \sqrt{p_i^k} - 1$ или $l_i = \sqrt{p_i^k} + 1$.

Так как $E[l_i]$ представляется в виде прямой суммы циклических групп, то можно зафиксировать некоторую образующую пару точек G_i и H_i таким образом, чтобы любую точку в $E[l_i]$ можно было представить с помощью них. Рассмотрим точки $P_i = a_{1,i}G_i + b_{1,i}H_i$ и $Q_i = a_{2,i}G_i + b_{2,i}H_i$, принадлежащие $E[l_i]$, где $a_{1,i}$, $a_{2,i}$, $b_{1,i}$, $b_{2,i} \in [0, l_i - 1]$. Для некоторых зафиксированных целых α_i, $\beta_i \in [0, l_i - 1]$ определим функцию следующим образом:

$$L_{\alpha_i, \beta_i} : E[l_i] \times E[l_i] \to E[l_i]$$

$$\text{и } L_{\alpha_i, \beta_i}(P_i, Q_i) = (a_{1,i}, b_{1,i} - a_{2,i}b_{2,i})(\alpha_i G_i + \beta_i H_i),$$

исключая тривиальный случай, когда α_i и β_i одновременно равны нулю.

Пусть G_1, G_2 и G_3 — три Абелевы группы. Билинейное спаривание является отображением $e : G_1 \times G_2 \to G_3$ среди этих трех групп, и отображение должно удовлетворять свойству билинейности: для α, $\beta \in G_1$, γ, $\delta \in G_2$ справедливо $e(\alpha + \beta, \gamma) = e(\alpha, \gamma) + e(\beta, \gamma)$, $e(\alpha, \gamma + \delta) = e(\alpha, \gamma) + e(\alpha, \delta)$.

Следующая теорема показывает, что функция L_{α_i, β_i} задает билинейное спаривание.

Теорема 2 [5]. Функция L_{α_i, β_i} обладает следующими свойствами:

1. Тождественность.

Для всех $P_i \in E[l_i]$

$$L_{\alpha_i, \beta_i}(P_i, P_i) = O.$$

2. Билинейность.

Дл всех P_i, Q_i, $R_i \in E[l_i]$

$L_{\alpha_i, \beta_i}(P_i + Q_i, R_i) = L_{\alpha_i, \beta_i}(P_i, R_i) + L_{\alpha_i, \beta_i}(Q_i, R_i)$ и

$L_{\alpha_i, \beta_i}(P_i, Q_i + R_i) = L_{\alpha_i, \beta_i}(P_i, Q_i) + L_{\alpha_i, \beta_i}(P_i, R_i)$.

3. Антисимметричность.

Для любых P_i, $Q_i \in E[l_i]$

$$L_{\alpha_i, \beta_i}(P_i, Q_i) = -L_{\alpha_i, \beta_i}(Q_i, P_i).$$

4. Невырожденность.

Для всех $P_i \in E[l_i]$

$$L_{\alpha_i, \beta_i}(P_i, \mathrm{O}) = O.$$

Кроме того, если $L_{\alpha_i, \beta_i}(P_i, Q) = O$ для всех $Q_i \in E[l_i]$, тогда $P_i = O$.

Функция L_{α_i, β_i} называется спариванием, поскольку она отображает $E[l_i] \times E[l_i]$ в $E[l_i]$ (аналогично традиционным спариваниям Вейля и Тейта).

Пусть $L_{\alpha,\beta}(P, Q) = L \in E(Z_{q^k})$ и $L_{\alpha_i, \beta_i}(P_i, Q_i) = L_i \in E(GF(p_i^k))$. Зададим спаривания $L_{\alpha,\beta}(P, Q)$ как $X_{L_i} \equiv X_L \bmod p_i$, $Y_{L_i} \equiv Y_L \bmod p_i$, $Z_{L_i} \equiv Z_L \bmod p_i$ с помощью множества точек G_i, H_i и целых чисел α_i, $\beta_i \in [1, ..., l_i - 1]$, где $L = (X_L:Y_L:Z_L)$, $L_i = (X_{L_i}:Y_{L_i}:Z_{L_i})$, $P = (X_P:Y_P:Z_P)$, $Q = (X_Q:Y_Q:Z_Q) \in Z_{q^k}$, $P_i(X_{P_i}:Y_{P_i}:Z_{P_i})$, $Q(X_{Q_i}:Y_{Q_i}:Z_{Q_i}) \in GF(p_i^k)$, $X_{P_i} \equiv X_P \bmod p_i$, $Y_{P_i} \equiv Y_P \bmod p_i$, $Z_{P_i} \equiv Z_P \bmod p_i$, $X_{Q_i} \equiv X_Q \bmod p_i$, $Y_{Q_i} \equiv Y_Q \bmod p_i$, $Z_{Q_i} \equiv Z_Q \bmod p_i$ и $i = 1, ..., s$.

Замечание. Значения X_L, Y_L, Z_L представляются в системе остаточных классов числами X_{L_i}, Y_{L_i}, Z_{L_i} по основаниям p_i.

Схема разделения точек на эллиптической кривой

Пусть задано множество точек, которые нам нужно разделить между участниками $\{M_1, M_2, ..., M_m\} \subseteq E\lfloor Z_q \rfloor$.Это может быть сообщением, представленным в виде точек, или ключом к криптосистеме.

Начальная фаза

Обозначим D — дилер и участники — n схемы разделения точек $\{U_1, U_2, ..., U_n\}$. Дилер совершает следующие шаги для определения параметров, использующих схемы.

1. Дилер D выбирает эллиптическую кривую E над Z_q, где $q = \prod_{i=1}^{s} p_i$, p_i — попарно различные простые числа так, чтобы $q \geqslant 2^{255}$. D выбирает целое четное число k и вычисляет множество чисел l_i.

2. Дилер D выбирает образующую пару $\{G_i, H_i\} \in E[l_i]$ и целые числа α_i, $\beta_i \in [1, l_i - 1]$, которые задают спаривание L_{α_i, β_i}.

3. Дилер D опубликовывает открытый ключ $\{E, q, l_i, k, \alpha G_i + \beta H_i\}$.

Фаза секретного распределения

Дилер использует следующий алгоритм распространения секретов между n участниками так, чтобы любые z или более участников могли легко восстановить секрет, а любые $z - 1$ или меньше участников не могли этого сделать и получить какую-либо дополнительную информацию о секрете.

1. Дилер D вычисляет матрицу A:

$$A = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & 2 & 2^2 & \dots & 2^{z-1} \\ 1 & 3 & 3^2 & \dots & 3^{z-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & (n-1) & (n-1)^2 & \dots & (n-1)^{z-1} \end{pmatrix}.$$

2. Дилер D выбирает $2zs$ случайных чисел $\{\tilde{a}_{j,i}, \tilde{b}_{j,i}\}$, каждое $\tilde{a}_{j,i}, \tilde{b}_{j,i} \in [0, l_i - 1]$ для $1 \leqslant j \leqslant z$ и $1 \leqslant i \leqslant s$.

3. Дилер D вычисляет $(a_{1,i}, a_{2,i}, ..., a_{n,i})^{\mathrm{T}} = A \cdot (\tilde{a}_{1,i}, \tilde{a}_{2,i}, ..., \tilde{a}_{t,i})^{\mathrm{T}}$ и $(b_{1,i}, b_{2,i}, ..., b_{n,i})^{\mathrm{T}} = A \cdot (\tilde{b}_{1,i}, \tilde{b}_{2,i}, ..., \tilde{b}_{t,i})^{\mathrm{T}}$.

4. В завершении D сообщает i пар $\{a_{j,i}, b_{j,i}\}$ (секретный ключ) пользователю U_j для всех $1 \leqslant j \leqslant n$, где $1 \leqslant i \leqslant s$.

Фаза разбиения точек

После распространения секретов дилер может распространять точки среди n участников, используя следующий алгоритм.

1. Для того чтобы дилер D распространил m точек — $\{M_1, M_2, ..., M_m\}$, он генерирует случайные числа $c_{j,i}$, $d_{j,i} \in [0, l_i - 1]$ и вычисляет значение $Q_{j,i} = c_{j,i}G_i + d_{j,i}G_i$ для всех $1 \leqslant j \leqslant m$, $1 \leqslant i \leqslant s$.

Примечание. Каждая точка имеет вид $M_j = \{M_{j,i} | 1 \leqslant i \leqslant s\}$.

2. Дилер D вычисляет $R_{j,i} = L_{\alpha_i, \beta_i}(Q_{j,i}, P_{j,i}) + M_{j,i}$ для всех $1 \leqslant j \leqslant m$, $1 \leqslant i \leqslant s$ и публикует $\{c_{j,i}, d_{j,i}, R_{j,i}\}$ (открытый ключ).

Алгоритм восстановления точки

Любые z пользователей — $\{U_{u_1}, U_{u_2}, ..., U_{u_z}\}$ — могут восстановить m точек $\{M_j\}_{1 \leqslant j \leqslant m}$, используя следующую последовательность действий.

1. Каждый U_{u_w} берет массив целых чисел $\{c_{w,i}, d_{w,i}\}$ из открытого ключа, где $1 \leqslant w \leqslant z$, $1 \leqslant i \leqslant s$.

2. Каждый U_{u_w} вычисляет $Q_{j,w,i} = L_{\alpha_i, \beta_i}(Q_{j,i}, P_{u_w, i})$, где $P_{u_w, i} = a_{u_w, i} G_i + b_{u_w, i} H_i$ и $Q_{j,i} = c_{j,i} G_i + d_{j,i} H_i$, где $1 \leqslant w \leqslant z$ и $1 \leqslant i \leqslant s$.

3. Каждый пользователь вычисляет $T_{j,i} = \sum_{w=1}^{t} y_j Q_{j,w,i}$, где $y_w = \prod_{j=1, w \neq i}^{z} \frac{1}{u_w - u_j}$ и $1 \leqslant i \leqslant s$.

4. Каждый пользователь берет точку $R_{j,i}$ из открытого ключа и восстанавливает $M_{j,i}$ как $R_{j,i} - T_{j,i}$.

5. Из множества $M_{j,i}$ восстанавливается точка M_j, где $1 \leqslant i \leqslant s$ и $j \in [1, ..., m]$.

Анализ безопасности

Покажем, что построенная (z, n)-пороговая схема разделения секрета является совершенной. Для этого докажем следующую теорему.

Теорема 3. Если по крайней мере $n - z + 1$ пользователей честных, то любые $z - 1$ пользователей ничего не узнают о $\tilde{P}_{t,i}$, где i — зафиксированное целое число, принадлежащее отрезку $[1...s]$.

Доказательство

Пусть произвольных $z - 1$ пользователей — $\{U_{j_1, i}, U_{j_2, i}, ..., U_{j_{t-1}, i}\}$ — знают свой секрет, и E, q, l_i, k, $\alpha_i G_i + \beta_i H_i$, $\{a_{w_j, i}, b_{w_j, i}\}$, где $1 \leqslant j \leqslant z - 1$. Пусть

$$B = \begin{pmatrix} 1 & j_1 & j_1^2 & \dots & j_1^{z-1} \\ 1 & j_2 & j_2^2 & \dots & j_2^{z-1} \\ 1 & j_3 & j_3^2 & \dots & j_3^{z-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & j_{z-1} & j_{z-1}^2 & \dots & j_{z-1}^{z-1} \end{pmatrix},$$

тогда $(a_{j_1, i}, a_{j_2, i}, ..., a_{j_{z-1}, i})^{\mathrm{T}} = B \cdot (\tilde{a}_{1, i}, \tilde{a}_{2, i}, ..., \tilde{a}_{z-1, i})^{\mathrm{T}}$.

Ранг B равен $z - 1$, так как матрица B состоит из $z - 1$ столбцов матрицы Вандермонда и $j_i \neq j_k$, если $i \neq j$. Из теории линейных уравнений $\tilde{a}_{z,i}$ — любое значение из $[0, l_i - 1]$. Тот же самый результат сохраняется и для $\tilde{b}_{z,i}$, поэтому любые $z - 1$ пользователей ничего не узнают о $\tilde{P}_{z,i}$.

Теорема доказана.

Приведем лемму из работы [4], касающейся случая, когда $L_{\alpha_i, \beta_i}(P_i, Q_i) = \mathrm{O}$, которая потребуется при доказательстве теоремы о вероятности выбора случайной точки.

Лемма 1 [4]. Если $\alpha_i \beta_i \neq 0$ и $P_i \neq Q_i$, то $L_{\alpha_i, \beta_i}(P_i, Q_i) = O$ тогда и только тогда, когда P_i принадлежит множеству $\{kQ_i : 0 \leqslant k < l_i\}$.

В следующей теореме мы покажем, что вероятность выбора наугад точек $P_{z,i}$, задающих точку P_z, очень мала.

Теорема 4 (теорема о случайном выборе точки). Не зная точки $\tilde{P}_z$, вероятность выбора случайной точки $P_z \in E(Z_{q^k})$ такой, чтобы выполнялась $M_{j,i} = R_{j,i} - L_{\alpha_i, \beta_i}(Q_{j,i}, \tilde{P}_{z,i})$ для всех $i = 1, ..., s$, равна $\prod_{i=1}^{s} \frac{1}{l_i}$.

Доказательство

Вероятность случайного выбора какой-то одной зафиксированной точки $\tilde{P}_{z,i}$ такой, чтобы $M_{j,i} = R_{j,i} - L_{\alpha_i, \beta_i}(Q_{j,i}, P_{t,i})$, равна тому, что $L_{\alpha_i, \beta_i}(Q_{j,i}, P_{z,i} - \tilde{P}_{z,i}) = \mathrm{O}$. Из леммы 1 получим, что $P_{z,i} \in \{eQ_{j,i} + \tilde{P}_{t,i} | 0 \leqslant e \leqslant l_i\}$. Следовательно, согласно равенству $M_{j,i} = R_{j,i} - L_{\alpha_i, \beta_i}(Q_{j,i}, P_i)$, вероятность случайного выбора одной зафиксированной точки $P_i \in E[l_i]$ равна $\frac{1}{l_i^2}$, значит, вероятность выбора l_i случайных точек равна $\frac{l_i}{l_i^2} = \frac{1}{l_i}$.

Вероятность случайного выбора независимых s точек $\tilde{P}_{z,i}$, где $1 \leqslant i \leqslant s$, вычисляется по формуле произведения вероятностей. При условии, что вероятность случайного выбора i точки равна $\frac{1}{l_i}$, получим вероятность выбора случайной точки

$P_z \in E(Z_{q^k})$, так что $M_{j,i} = R_{j,i} - L_{\alpha_i, \beta_i}(Q_{j,i}, P_i)$ для всех $i = 1, .., s$, равной $\prod_{i=1}^{s} \frac{1}{l_i}$.

Теорема доказана.

Выводы

Было доказано, что предложенная схема разделения секрета является безопасной при выполнении двух условий:

- по крайней мере $n - z + 1$ пользователей в схеме честные;
- распространено не больше, чем $Q(\lg(q))$ точек.

Преимущество в реализации (z, n)-пороговой схемы разделения секрета на эллиптической кривой над Z_q заключается в том, что при использовании проективной системы координат можно эффективно использовать систему остаточных классов, что существенно ускоряет процесс шифрования и дешифрования информации.

Список литературы

1. **Брюс Ш. Ш.** Прикладная криптография: Протоколы, Алгоритмы, Исходные тексты на С. М.: Триумф, 2002. 408 с.

2. **Blakley G. R.** Safeguarding cryptographic keys // Proc. AFIPS 1979 National Computer Conference. N. Y., 1979. V. 48. P. 313—317.

3. **Shamir A.** How to Share a Secret // Comm. ACM. 1979. V. 22. N 1. P. 612—613.

4. **Duo L., Dongping H., Ping L., Yiqi D.** New schemes for sharing points on an elliptic curve // Computers and Mathematics with Applications. 2008. Vol. 56. P. 1156—1561.

5. **Lee H.-S.** A self-pairing map and its applications to cryptography // Applied Mathematics and Computation. 2004. Vol. 151. P. 671—678.

6. **Василенко О. Н.** Теоретико-числовые алгоритмы в криптографии. М.: Изд. МЦНМО, 2003.

7. **Нечаев В. И.** Элементы криптографии. М.: Высшая школа, 1999.

8. **Черемушкин А. В.** Лекции по арифметическим алгоритмам в криптографии. М.: Изд. МЦНМО, 2002.

9. **Schoof R.** Elliptic curves over finite fields and the computation of square roots mod p // Math. Comp. 1985. Vol. 44. P. 483—494.

10. **Schoof R.** Counting points on elliptic curves over finite fields // J. Theor. Nombres Bordeaux. 1995. Vol. 7. P. 219—254.

11. **Dewaghe I.** Remarks on the Schoof-Elkies-Atkin algorithm // Mathematics of computation. 1998. V. 67. N 223. P. 1247—1252.

12. **Lercier R.** Computing isogenies in GF(2') // Algorithmic number theory, Lecture Notes in Computer Science. 1996. Vol. 1122. P. 197—212.

13. **Lercier R., Morain F.** Counting the number of points on elliptic curves over finite fields: strategies and performances // Eurocript-95, Lecture Notes in Computer Science. 1995. Vol. 921. P. 79—94.

14. **Болотов А. А., Гашков С. Б., Фролов А. Б., Часовских А. А.** Алгоритмические основы эллиптической криптографии. М.: Изд-во МЭИ, 2004. 499 с.

УДК 004.658.2,004.423.42

Д. А. Шкляев, вед. программист,
Институт систем информатики им. А. П. Ершова СО РАН, г. Новосибирск
e-mail: dmitrichkl@yahoo.com

Формальная верификация понятий отказоустойчивости для распределенных баз данных

Рассматриваются формальная спецификация и автоматизированная верификация систем обработки транзакций, используемых в распределенных базах данных. В таких системах стандартный набор свойств ACID должен быть обеспечен комбинацией протоколов контроля параллелизма и восстановления. В существующей литературе такие протоколы обычно изучаются раздельно, и проблема их взаимодействия нередко игнорируется. Для изучения формальной верификации комбинированного набора протоколов мы специфицируем систему обработки транзакций, интегрирующих строгое двухфазное блокирование, протокол восстановления undo/redo и двухфазное атомарное завершение. Мы доказали с помощью интерактивного доказывателя теорем PVS, что в нашей системе выполняются свойства атомарности, долговечности и сериализуемости.

Ключевые слова: *базы данных, протоколы контроля параллелизма, протоколы восстановления, отказоустойчивость, формальная спецификация, автоматизированная верификация, интерактивный доказыватель теорем*

Введение

Транзакция определяется как логически неделимая последовательность операций на базе данных. Для систем обработки транзакций (СОТ) характерна значительная сложность, потому что они должны обеспечить защиту от ошибок, вызываемых двумя классами причин: параллелизмом (concurrency) и отказами аппаратного обеспечения. Чтобы предотвращать такие ошибки, распределенная СОТ должна обеспечивать по крайней мере следующие механизмы: контроль параллелизма, централизованное восстановление на отдельном узле и распределенное восстановление.

Когда транзакции осуществляют доступ к базе данных поочередно, они могут мешать друг другу. Порядок перемежения транзакций дается в *расписании* (schedule), которое представляет собой последовательность операций, таких как чтение или запись отдельных данных, где каждая операция принадлежит определенной транзакции. Чтобы предотвратить конфликты между транзакциями, протоколы контроля параллелизма должны обеспечить *сериализуемость*. Они должны допускать только такие расписания, которые эквивалентны некоторому *серийному* расписанию, т. е. расписанию, не имеющему чередования между операциями разных транзакций.

Протоколы централизованного восстановления занимаются двумя типами неудач (отказов): неудачами транзакций и отказами памяти. Когда транзакция неспособна завершить свое исполнение по какой-либо причине, например, при конфликте с другой транзакцией, она *абортируется* (досрочно прерывается), и протокол восстановления должен устранить (стереть) ее частичные результаты. Он также должен обеспечить, чтобы результаты *фиксированных*, т. е. успешно завершенных транзакций, никогда не пропадали. Эти две задачи осложняются отказами памяти, которые могут стереть часть памяти и также вынудить прерывание некоторых транзакций.

Когда транзакция обновляет данные в нескольких вычислительных узлах, все они должны достигнуть соглашения о том, что нужно ли фиксировать или абортировать эту транзакцию. Взаимодействие между узлами, необходимое для достижения такого решения, должно быть обеспечено протоколами распределенного восстановления, обычно в форме протокола *распределенного атомарного фиксирования*.

Полный набор требований для СОТ обычно дается в форме сокращения ACID (*atomicity* — атомарность, *consistency* — консистентность (согласованность), *isolation* — изоляция и *durability* — долговечность) [1].

СОТ обеспечивают самые важные сервисы многих современных приложений, например банковских, и часто используются в приложениях, критичных с точки зрения безопасности. Поэтому гарантировать их корректность чрезвычайно важно. Одним из методов обеспечения корректности является формальная верификация. Однако в данном случае она связана со значительными трудностями, потому что нам нужно доказать наличие нескольких весьма различных свойств для комбинации распределенных отказоустойчивых прото-

колов. Для изучения верификации СОТ мы выбираем один конкретный характерный протокол, определяем подходящий формальный метод для его спецификации и верификации, а также средства компьютерной поддержки данного метода. Поэтому в оставшейся части данного введения мы отвечаем на следующие вопросы:

1. В чем состоят свойства корректности для СОТ?

2. Какой конкретный протокол был выбран для изучения и почему?

3. Как получить автоматизированную поддержку для спецификации и верификации?

4. В чем состоят результаты нашей верификации?

1. Свойства ACID. СОТ должна удовлетворять четырем свойствам, которые перечислены ниже.

- *Атомарность* означает, что результаты транзакции либо отражаются в базе данных полностью, либо вообще не отражаются (в зависимости от фиксирования или абортирования данной транзакции).
- *Консистентность* требует, чтобы некоторые инварианты (ограничения) сохранялись каждой транзакцией в каждом состоянии базы данных.
- *Изоляция* означает, что разные транзакции не мешают друг другу. Обычно изоляция заменяется более точным понятием сериализуемости.
- *Долговечность* значит, что результаты фиксированных транзакций должны выдерживать даже многократные отказы памяти. Это свойство тесно связано с сериализуемостью, и их обычно изучают вместе.

2. Рассмотренные протоколы. В данной статье мы рассматриваем протоколы восстановления в более интегрированной форме, чтобы обеспечить более убедительную верификацию по сравнению с тем случаем, когда объединяются изолированные результаты по этим протоколам. Это также облегчает переиспользование верификации для более реалистичных архитектур. Мы даем здесь экспериментальную систему обработки транзакций, тесно объединяющую строгое двухфазное блокирование, восстановление undo/redo и двухфазную фиксацию. Наша интеграция сосредоточена на интеракции между устойчивой и неустойчивой памятью и на управлении журналом; детали обмена сообщениями в двухфазной фиксации не рассматриваются. Мы также разработали формальные определения атомарности, долговечности и сериализуемости для отказоустойчивой системы.

3. Автоматическая поддержка. Ручная формальная верификация сложного протокола может содержать ошибки с не меньшей вероятностью, чем сам протокол. Поэтому в нашем случае какая-то форма компьютерной поддержки является незаменимой. Изучаемый здесь протокол весьма зависит от сложных структур данных и обладает непростыми условиями корректности. Поэтому полностью автоматическая верификация едва ли возможна, и мы решили использовать интерактивный доказыватель теорем. Мы выбрали PVS [2] из-за его удобного языка спецификации и легкости использования.

4. Результаты верификации. Свойства атомарности, долговечности и сериализуемости были полностью доказаны интерактивным доказывателем PVS. Доказательство атомарности и долговечности (тесно связанных в нашей модели) было наиболее трудоемким. Вся верификация была очень сложной и потребовала 1,5 месяца, но она значительно улучшила наше понимание протокола.

1. Атомарность, долговечность и сериализуемость

База данных состоит из набора *данных*, распределенных по нескольким узлам. Каждый узел имеет как *устойчивую* (энергонезависимую), так и *неустойчивую* (энергозависимую) память. Неустойчивая память (как правило, RAM) является быстрой, но ограниченной по объему вследствие своей дороговизны, и в ней может находиться лишь часть базы данных. Устойчивая память (обычно жесткий диск) дешева и велика по вместимости, но является медленной. В любой момент времени каждая единица данных имеет *устойчивое значение* в устойчивой памяти, а также может иметь *неустойчивое значение* в неустойчивой памяти.

Мы рассматриваем протоколы, в которых транзакции осуществляют атомарные операции на данных. Каждая операция обозначается парой, состоящей из ее названия и того узла, на котором она была выполнена. Самые важные операции — это чтение и запись, являющиеся единственными операциями транзакций, непосредственно использующими значения данных. Пусть $(Read(T, x, v), st)$ обозначает операцию чтения, выполняемую транзакцией T в узле st на данном x и получающую значение v, и $(Write(T, x, v), st)$ — операцию записи, выполняемую транзакцией T в узле st на данном x и присваивающую x значение v. Операции $(Commit(T), st)$ и $(Abort(T), st)$ обозначают соответственно успешное и неуспешное завершение транзакции T в узле st. Заметим, что аборт транзакции может быть инициирован самой транзакцией, а может быть и вынужден системой (например, после отказа памяти). Мы также рассматриваем операции для управления памятью. Бесконечная последовательность операций на базе данных называется *расписанием*. $S[i]$ обозначает операцию с индексом i в расписании S.

Атомарность и долговечность. Мы говорим, что *T фиксируется* в узле st в списке S, обозначая это как $Commits(T, st, S)$, если S включает операцию $(Commit(T), st)$. Мы говорим, что T фиксируется в списке S, обозначая это как $Commits(T, S)$, если

существует узел *st* такой, что выполняется *Commits*(*T*, *st*, *S*). Подобно этому, аборт *T* в узле *st* и ее "глобальный" аборт представляются операциями *Aborts*(*T*, *st*, *S*) и *Aborts*(*T*, *S*) соответственно. После этого мы можем представить атомарность и долговечность комбинацией двух свойств:

1) *консистентности решения*. Решения фиксировать или абортировать транзакцию *консистентны* в списке *S*, если

$$\forall T\colon Commits(T, S) \Rightarrow \neg Aborts(T, S);$$

2) *немедленности обновления* (*"update in place"*). Неформальное определение атомарности, данное во введении, использует выражение "отраженный в базе данных", которое не следует понимать буквально. Действительно, в нашей модели стабильные и нестабильные значения каждого данного в любой момент могут различаться, и операция чтения может получить любое из них. Поэтому было бы более уместно определять присутствие в базе данных только в терминах непосредственно наблюдаемого поведения, т. е. результатов взаимодействия пользователя с базой данных посредством транзакций. В данной работе единственными операциями транзакций, имеющими доступ к значениям данных, являются чтение и запись, и поэтому формальное определение атомарности должно устанавливать соответствие между значениями, получаемыми операциями чтения, и значениями, производимыми операциями записи. Здесь мы рассматриваем только протоколы, приводящие к немедленному обновлению данных (*"update in place"*). В таких протоколах каждая операция чтения должна получать "последнее фиксированное значение" данного, т. е. последнее значение, записанное в данное, которое было проведено фиксированной транзакцией.

Для упрощения формального определения немедленного обновления мы предполагаем, что каждый список начинается операциями фиктивной начальной транзакции *T*0, записывающей начальные значения всех данных и после этого фиксирующейся во всех узлах. Мы также рассматриваем только операции чтения фиксированных транзакций, потому что значения, полученные абортированными транзакциями, не имеют никакого значения. Для краткости мы опускаем узел в записи операций чтения и письма, когда его значение несущественно, т. е. например, (*Write*(*T*, *x*, *v*), *st*) иногда заменяется на *Write*(*T*, *x*, *v*).

Мы говорим, что *S не включает неабортированных записей данного x между индексами i и j* ($i < j$), обозначая это как *NoWrites*(*x*, *i*, *j*, *S*), если для любых *T*, *x* и *k* (такого, что $i < k$ и $k < j$) имеет место следующее: если $S[k] = Write(T, x, v)$, то *Aborts*(*T*, *S*). Используя это сокращение, мы говорим, что *S обеспечивает немедленное обновление данных*, если выполняется следующее: если $S[j] = Read(T2, x, v)$ и *Commits*(*T2*, *S*), то существуют *i* и *T1* такие, что $i < j$, $S[i] = Write(T1, x, v)$, *Commits*(*T1*, *S*) и *NoWrites*(*x*, *i*, *j*, *s*), причем если $T1 \neq T2$, то *T1* должна фиксироваться до операции чтения транзакции *T2*. Наконец, мы говорим, что *S* является *атомарным* и *долговечным*, если в нем выполняется консистентность решения и немедленность обновления данных.

Сериализуемость. Здесь мы повторяем некоторые определения из работы [3]. Пусть *S*, *SA*, *SB* и *SS* обозначают конечные списки, состоящие только из фиксированных операций чтения и письма. В таких списках каждая операция чтения получает в точности *последнее записанное значение* данного. Мы говорим, что операции *a1* и *a2 конфликтуют* в *S*, если они выполнены на одном и том же данном и по крайней мере одно из них является записью. *SA* и *SB элементарно эквивалентны*, если $SA = SC\ a1\ a2\ SD$, $SB = SC\ a2\ a1\ SD$ для некоторых *SC* и *SD*, и операции *a1* и *a2* не конфликтуют. *SA* и *SB конфликтно эквивалентны*, если существует конечная последовательность списков $S_0, S_1, \ldots, S_k$, $k \geqslant 0$, таких, что $SA = S_0$, $SB = S_k$ и для всех $i < k$ списки S_i и S_{i+1} элементарно эквивалентны. Список *SS* называется *серийным*, если он не имеет чередования между операциями разных транзакций. Наконец, мы говорим, что список *S конфликтно сериализуем*, обозначая это как *Conf_serializable(S)*, если существует серийный список *SS* такой, что *S* и *SS* конфликтно эквивалентны.

Теперь мы можем распространить определения из работы [3] на более общую модель, используемую в данной статье. Для бесконечного расписания *S*, состоящего из операций как фиксированных, так и абортированных транзакций, мы обозначаем как *C*(*S*) его *фиксированную проекцию*, т. е. расписание, полученное из *S* путем удаления всех операций, кроме чтения и письма фиксированными транзакциями. Если атомарность для *S* доказана, то в *C*(*S*) каждое чтение вновь получает последнее записанное значение, и поэтому можно применить предыдущее определение конфликтующих операций. Для расписания *S* мы обозначаем как *prefix*(*S*, *k*) его префикс длины *k*. Расписание *S* называется *отказоустойчиво конфликтно сериализуемым*, если $\forall k$: *Conf_serializable(prefix(C(S), k))*.

2. Методы обработки транзакций

Здесь мы даем неформальное описание основных методов обработки транзакций, используя многие определения из работы [4].

Управление памятью. В целях повышения быстродействия транзакции осуществляют операции чтения и письма на версиях данных, находящихся в неустойчивой памяти. Поэтому операция запи-

си не приводит к немедленному изменению значения данного в устойчивой памяти. Обмен данными между неустойчивой и устойчивой памятью осуществляется операциями *Fetch* и *Flush*. *Fetch(x)* копирует *x* из устойчивой памяти в неустойчивую, в то время как *Flush(x)* перемешает *x* из неустойчивой памяти в устойчивую. Операция сбрасывания (*Flush*) обычно выполняется, когда в неустойчивой памяти не остается места для новых данных. Если транзакция пытается прочитать данное, находящееся только в устойчивой памяти, то это данное сначала должно быть извлечено (*Fetch*) в неустойчивую память, а потом прочитано.

Существуют два типа отказов памяти: системные, когда теряется содержимое неустойчивой памяти, и медийные, когда теряется также содержимое устойчивой памяти. Здесь мы рассматриваем только системные отказы, потому что медийные отказы относительно редки, а восстановление после них весьма затруднительно.

Отмена и повтор. Мы говорим, что механизм восстановления *требует отмену* (*undo*), если он позволяет сбрасывать данные, записанные еще не фиксированной транзакцией. Если в этот момент произойдет системный отказ, то к началу восстановления устойчивая база данных будет содержать результаты не фиксированных транзакций, которые обычно должны быть *отменены* для обеспечения атомарности последующих чтений. Мы говорим, что механизм восстановления *требует повтор* (*redo*), если он позволяет фиксировать транзакцию до того, как все записанные ей значения были сброшены из неустойчивой в устойчивую память. При системном отказе в этот момент, к началу восстановления, в устойчивой базе данных будут отсутствовать какие-то результаты фиксированных транзакций, поэтому они должны быть *повторены* (перезаписаны). В данной статье мы рассматриваем протокол, который требует и отмену, и повтор. Такие протоколы позволяют сбрасывать данные из неустойчивой памяти в тот момент, который наиболее выгоден с точки зрения производительности системы. Однако за такое повышение производительности приходится расплачиваться значительной сложностью протоколов.

Как известно, чтобы обеспечить возможность отмены и повтора после системного отказа, мы должны хранить информацию о значениях, записанных транзакцией, в *журнале изменений*, хранящемся в устойчивой памяти. В базисном алгоритме отмены/повтора из работы [4] журнал имеет форму последовательности записей вида [*T*, *x*, *v*], содержащих значение *v*, которое транзакция *T* присвоила данному *x*, и такая запись добавляется к журналу после каждого соответствующего присвоения. Журнал также содержит множества фиксированных и абортированных транзакций и другую информацию. Когда неустойчивая память теряется при системном отказе, протокол изучает журнал, находит последние фиксированные значения всех данных, которые когда-либо были изменены, и восстанавливает их в качестве новых значений этих данных. В следующих двух параграфах мы объясняем два изменения, которые были внесены в данной работе в базисный алгоритм. Первое из них объясняется соображениями эффективности, второе — стремлением упростить верификацию.

Мы мало бы выиграли от использования неустойчивой памяти, если бы каждая операция записи также получала доступ к устойчивой памяти. Поэтому важно минимизировать число значений, хранимых в журнале. В нашем протоколе большинство операций чтения и письма проводятся на неустойчивой памяти и не меняют журнал, а значения, записанные транзакцией, добавляются к журналу незадолго до ее фиксирования (а именно, во время операции *предфиксирования*). Легко видеть, что этого достаточно для обеспечения того, чтобы журнал всегда содержал последнее фиксированное значение каждого данного. Эти значения используются только операциями перезапуска системы и абортирования.

Во время перезапуска после системного отказа протокол из работы [4] *сканирует* журнал, т. е. последовательность записей вида [*T*, *x*, *v*], в целях нахождения множества обновленных данных и их последних фиксированных значений. В нашем протоколе множество обновленных данных учитывается во время нормальной работы системы, и только *последнее* фиксированное значение содержится в журнале в любой момент времени. В результате реализация операций перезапуска и абортирования является весьма простой по сравнению с [4], и цикл в ней не требуется. Отсутствие цикла в моделировании операций значительно упрощает верификацию.

Строгое двухфазное блокирование. В современных базах данных строгое двухфазное блокирование (2ФБ) является наиболее распространенным методом для обеспечения того, чтобы все исполнения являлись и строгими, и сериализуемыми. Строгое 2ФБ дает транзакции доступ к данному только в том случае, если она в настоящий момент наложила на него блокировку. Базисный протокол, рассматриваемый в этой статье, имеет только два типа блокировок: совместный и эксклюзивный.

Совместный. Если транзакция *T* наложила совместную блокировку на данное *x*, то *T* может читать *x*, но не может записывать его значение. Произвольное число транзакций может одновременно иметь совместную блокировку на данное.

Эксклюзивный. Если транзакция *T* наложила эксклюзивную блокировку на данное *x*, то *T* может и читать, и записывать *x*. Не более одной транзак-

ции разрешается иметь эксклюзивную блокировку на данное в любой момент времени. Мы также следуем известному требованию, что транзакция может записывать данное не более одного раза [5].

Для доступа к данному транзакция сначала должна наложить на него блокировку соответствующего типа. Если данное уже имеет блокировку несовместимого типа, то запрос на блокировку отклоняется. Строгое 2ФБ требует, чтобы каждая транзакция накладывала и освобождала блокировки в двух сменяющих друг друга фазах:

фазе роста — транзакция может получать блокировки, но не может их освобождать;

фазе сокращения — транзакция фиксируется или абортируется и одновременно освобождает все свои блокировки.

Двухфазное фиксирование (2ФФ). Базовый протокол 2ФФ работает следующим образом. Один из его узлов (называемых *участниками*) также выступает в роли *координатора* для руководства решением о судьбе транзакции. В этом случае в первой фазе после получения приказа *голосовать* от координатора каждый участник посылает координатору свой голос: ДА или НЕТ, выражая свое желание фиксировать или абортировать транзакцию. Координатор собирает все голоса и принимает *решение*. Если от всех участников был получен голос ДА, то принимается решение фиксировать транзакцию; в противном случае транзакция абортируется. Во второй фазе координатор рассылает решение всем участникам. Если участник получает решение от координатора, он фиксирует или абортирует транзакцию в соответствии с этим решением. Если решение не доходит из-за отказа координатора, участник должен найти какой-то другой способ принять решение.

Каждый протокол 2ФФ должен удовлетворять по крайней мере следующим свойствам [6]:

AC1: Все участники, достигающие решения, принимают то же самое решение.

AC2: Если какой-либо участник принимает решение о фиксации, то все участники до этого должны были проголосовать ДА.

AC3: Если все участники голосуют ДА и не происходит отказов, то все участники принимают решение о фиксации.

AC4: Каждый участник решает не более одного раза (т. е. решение необратимо).

В работе [4], где базисный протокол отмены/повтора не поддерживает распределенную фиксацию, все транзакции, выполняемые на каком-либо узле, абортируются после системного отказа в этом узле. Заметим, что если узел голосует ДА при исполнении 2ФФ, решающего о судьбе транзакции T, и на этом узле происходит системный отказ до получения решения от координатора, то абортирование T может противоречить решению, принятому другими узлами. Для поддержки протокола 2ФФ мы внесли нетривиальное изменение в базисный протокол отмены/повтора, добавив в него операцию *предфиксирования* (*Precommit(T)*, *st*). Эта операция помещает значения, записанные T, в журнал изменений, размещенный в узле *st*, и одновременно узел *st* посылает голос ДА в исполнении 2ФФ, решающем о фиксировании или абортировании T. Это позволяет избежать абортирования T, если происходит системный отказ до прибытия решения от координатора, потому что утраченные значения, записанные T, теперь могут быть не отменены, а повторены, и T может продолжить свое исполнение.

3. Формальное описание протокола

Структура данных. Пусть *DataI* обозначает множество имен всех данных, расположенных в одном из распределенных узлов *st*. Нашей целью является описание протокола, исполняемого узлом *st*. Сначала мы определим структуру данных протокола. Неустойчивая память состоит из: 1) неустойчивых значений некоторых данных из *DataI* и 2) блокировок: эксклюзивных (*xlocks*) и совместимых (*slocks*). В определении, данном ниже, *Trans* и *Values* — это неинтерпретированные типы данных, представляющие соответственно транзакции и значения данных в *DataI*. Мы предпочитаем использовать тотально определенные функции и потому обозначаем отсутствие значения символом ε.

Неустойчивая память:

1) *vvalue: DataI −> Values U {ε},*
2) *locks:*
   a) *xlocks: Trans −> setof[DataI],*
   b) *slocks: Trans −> setof[DataI].*

Устойчивая память содержит устойчивые значения всех данных в *DataI* и журнал изменений. Журнал включает:

a) статус каждой транзакции в *st* (неактивный, активный, предфиксированный, фиксированный или абортированный);

b) последние фиксированные значения всех данных в *DataI*;

c) предфиксированные значения некоторых данных в *DataI* (т. е. значения, записанные транзакциями, которые предфиксировались, но еще не фиксировались и не абортировались;

d) эксклюзивные (*Pxlocks*) и совместимые (*Pslocks*) блокировки предфиксированных транзакций;

e) множество данных, которые когда-либо обновлялись в неустойчивой памяти;

f) булевскую переменную, сигнализирующую о необходимости восстановления после системного отказа.

Устойчивая память:

1) *svalue : DataI −> Values,*

2) *log:*

a) *status : Trans −> {inact, act, pcom, com, ab},*

b) *cvalue : DataI −> Values,*

c) *pcvalue : DataI −> Values U{ε},*

d) *Plocks:*

1) *Pxlocks : Trans −> setof[DataI],*

1) *Pslocks : Trans −> setof[DataI],*

e) *updated : setof[DataI],*

f) *crashed : boolean.*

В начальном состоянии системы эти значения таковы (где *T0* — это фиктивная начальная транзакция):

$\forall$ *x, T:*

vvalue(x) = ε ∧ *xlocks(T)* = ∅ ∧ *slocks(T)* = ∅ ∧
svalue(x) = произвольное значение (записанное T0) ∧
status(T) = if (T = T0) then com else inact ∧
cvalue(x) = svalue(x) ∧ *pcvalue(x)* = ε ∧
Pxlocks(T) = ∅ ∧ *Pslocks(T)* = ∅ ∧
updated = ∅ ∧ *crashed = false.*

Мы специфицируем наш протокол машиной состояний с восемью атомарными операциями. Ниже мы показываем предусловие и результат каждой из них (опуская название узла, потому что здесь оно равно *st* для каждой операции). Результат дается в императивном стиле, близком к спецификациям PVS. Подразумевается, что переменные, которым не делается присвоение, остаются неизменными. Например, *xlocks(T) := xlocks(T)* ∪ *{x}* означает, что *T* накладывает на *x* эксклюзивную блокировку, а эксклюзивные блокировки всех остальных транзакций не меняются.

Заметим, что *Read*(*T*, *x*, *v*) означает, что значение *v* было прочитано, и предусловие этой операции требует, чтобы *v* было получено "правильным образом" (в соответствии с протоколом). Выше объяснялось, что если у данного *x* отсутствует неустойчивое значение, то оно должно быть взято из устойчивой памяти и после этого прочитано.

1) ***Real(T, x, v)***

Предусловие:

v = if vvalue(x) ≠ ε *then vvalue(x) else svalue(x),*
$\forall$ *T1 : x* ∈ *xlocks(T1)* ⇒ *T1 = T,*
status(T) = inact ∨ *status(T) = act.*

Результат:

vvalue(x) := v,
slocks(T) := if x ∉ *xlocks(T) then slocks(T)* ∪
∪ *{x} else slocks(T),*
status(T) := act.

2) ***Write(T, x, v)***

Предусловие:

$\forall$ *T1 : x* ∉ *xlocks(T1),*
$\forall$ *T1 : x* ∈ *slocks(T1)* ⇒ *T1 = T,*
status(T) = inact ∨ *status(T) = act.*

Результат:

vvalue(x) := v,
xlocks(T) := xlocks(T) ∪ *{x},*
*slocks(T) := slocks(T)**{x},*
status(T) := act,
updated := updated ∪ *{x}.*

3) ***Flush(x)***

Предусловие: *vvalue(x)* ≠ ε

Результат: *svalue(x) := vvalue(x).*

4) ***Precommit(T)***

Предусловие: *status(T) = act.*

Результат:

status(T) := pcom,
pcvalue := $\forall$ *x : if x* ∈ *xlocks(T) then vvalue(x) else pcvalue(x),*
Pxlocks(T) := xlocks(T), Pslocks(T) := slocks(T).

5) ***Commit(T)***

Предусловие: *status(T) = pcom.*

Результат:

status(T) := com,
cvalue := $\forall$ *x : if x* ∈ *xlocks(T) then pcvalue(x) else cvalue(x),*
pcvalue := $\forall$ *x : if x* ∈ *xlocks(T) then* ε *else pcvalue(x),*
Pxlocks(T) := ∅, *Pslocks(T) :=* ∅,
xlocks(T) := ∅, *slocks(T) :=* ∅.

6) ***Abort(T)***

Предусловие: *status(T) = act* ∨ *status(T) = pcom.*

Результат:

vvalue := $\forall$ *x : if x* ∈ *xlocks(T) then cvalue(x) else vvalue(x),*
status(T) := ab,
pcvalue := $\forall$ *x : if x* ∈ *xlocks(T) then* ε *else pcvalue(x),*
Pxlocks(T) := ∅, *Pslocks(T) :=* ∅,
xlocks(T) := ∅, *slocks(T) :=* ∅.

7) ***Crash***

Предусловие: отсутствует.

Результат:

vvalue := $\forall$ *x :* ε,
xlocks := $\forall$ *T :* ∅, *slocks :=* $\forall$ *T :* ∅,
crashed := true.

8) ***Restart(Tset)***

Предусловие: $\forall$ *T : T* ∈ *Tset* ⇔ *status(T) = act.*

Результат:

vvalue := $\forall$ *x : if x* ∈ *updated then*
(if pcvalue(x) ≠ ε *then pcvalue(x) else cvalue(x))*
else ε,
xlocks := $\forall$ *T : Pxlocks(T),*
slocks := $\forall$ *T : Pslocks(T),*
status := $\forall$ *T : if status(T) = act then ab else status(T),*
crashed := false.

Имеется также дополнительное предусловие для всех операций относительно переменной *crashed*: операция перезапуска может быть выполнена, только если *crashed = true*, а для всех остальных операций мы требуем *crashed = false.*

Пояснение. Операция перезапуска имеет параметр *Tset*, обозначающий множество транзакций,

абортируемых во время перезапуска. Это позволяет нам более строго определить аборты транзакций таким образом, что это понятие покрывает как "добровольные", так и "вынужденные" аборты, т. е. аборты, выполняемые самой системой во время перезапуска. Мы теперь говорим, что T абортируется в узле st в списке S, представляя это как $Aborts(T, st, S)$, если S включает одну из операций $(Abort(T), st)$ или $(Recover(Tset), st)$, где $T \in Tset$.

Исполнение системы. Мы моделируем глобальное состояние системы как функцию, которая назначает каждому узлу его локальное состояние. В этом случае исполнение системы представляется бесконечной последовательностью вида

$$gs_0 \xrightarrow{(an_0, st_0)} \ldots \xrightarrow{(an_{i-1}, st_{i-1})} gs_i \xrightarrow{(an_i, st_i)} gs_{i+1} \xrightarrow{(an_{i+1}, st_{i+1})} \ldots,$$

где gs_i — это глобальные состояния и (an_i, st_i) представляет операцию с названием an_i, исполняемую в узле st_i, для всякого $i \in \mathbf{N}$. Такие исполнения должны быть семантически осмысленными, и поэтому если an_i имеет форму $Read(T, x, v)$, $Write(T, x, v)$ или $Flush(x)$, то мы требуем, чтобы x было расположено в st_i. Кроме этого, каждый $gs_i(st_i)$ должен удовлетворять предусловию (an_i, st_i), и каждая пара $(gs_i(st_i), gs_{i+1}(st_i))$ должна соответствовать результату (an_i, st_i).

Распределенное завершение. Атомарность и сериализуемость можно доказать лишь при обеспечении свойств AC1, AC2 и AC4 протокола 2ФФ (AC3 требуется лишь для свойств живости, которые не рассматриваются в данной работе). В нашей модели доказательство AC4 для каждого узла не представляет трудностей. Свойства AC1 и AC2 требуют обмена сообщениями между узлами. Мы здесь не рассматриваем детали механизма коммуникаций и поэтому специфицируем AC1 и AC2 глобальными предикатами на расписаниях.

Требуются некоторые дополнительные сокращения для определения этих предикатов. Мы говорим, что *T решает* в узле st в расписании S, обозначая это как $Decides(T, st, S)$, если выполняет либо $Commits(T, st, S)$, либо $Aborts(T, st, S)$. Мы говорим, что *T активна* в узле st в расписании S, обозначая это как $Active(T, st, S)$, если S включает одну из операций $(Read(T, x, v), st)$ или $(Write(T, x, v), st)$ для некоторых x и v. Мы говорим, что T фиксируется (префиксируется) в узле *st в момент i* в расписании S, представляя это как $Commits(T, st, i, S)$ $(Precommits(T, st, i, S))$, если $S[i] = (Commit(T), st)$ $(S[i] = (Precommit(T), st))$. Как объяснялось в разделе 2, операция предфиксации соответствует голосу ДА в исполнении 2ФФ. Если S— это расписание, соответствующее системному исполнению, то AC1 и AC2 определяются для S следующим образом (где fn — это функция из множества узлов во множество натуральных чисел):

$$AC1(S) = \forall\ st1, st2, T:$$
$$(Decides(T, st1, S) \wedge Decides(T, st2, S)) \Rightarrow$$
$$(Commits(T, st1, S) \Leftrightarrow Commits(T, st2, S))$$
$$AC2(S) = \forall\ st, T, i: Commits(T, st, i, S) \Rightarrow$$
$$\exists\ fn : \forall\ st1 : Active(T, st1, S) \Rightarrow$$
$$(fn(st1) < i \wedge Precommits(T, st1, fn(st1), S)).$$

4. Спецификация и верификация в PVS

Моделирование значений. В нашей спецификации в PVS мы внесли некоторые изменения в ранее описанную модель с целью облегчить верификацию. Доказательство атомарности обычно используют леммы следующего вида: "если транзакция T абортируется, то ни одна транзакция никогда не сможет прочитать значения, записанные T''. Такие вещи очень трудно доказать, если мы позволим некоторой фиксированной транзакции производить те же значения, что и T. Эту проблему можно решить, если мы потребуем уникальности всех значений. В нашей модели в PVS это легко реализовать, если использовать вместо "настоящих" значений идентификаторы транзакций, записавших эти значения (такой же подход используется в многоверсионном контроле параллелизма в главе 5 книги [4]). Таким образом, операция $Write(T1, x, v)$ заменяется на операцию $Write(T, x)$, назначающую x значение *"T1"*. Операция $Read(T2, x, v)$, где v записано $T1$, заменяется операцией $Read(T2, x, T1)$. В результате этих изменений определение немедленного обновления становится следующим: если $S[j] = Read(T2, x, T1)$ и $Commits(T2, S)$, то существует i такое, что $i < j$, $S[i] = Write(T1, x)$, $Commits(T1, S)$ и $NoWrites(x, i, j, S)$.

Верификация атомарности и долговечности. Консистентность решения легко доказывается с использованием предположения AC1. В доказательстве "немедленного обновления" большинство лемм являются довольно сложными инвариантами, доказываемыми по индукции. Чтобы представить типичный пример, мы вводим следующие обозначения: $sf(x)$ — это узел, на котором расположен x, $AS(se)$ — это последовательность операций, соответствующих системному исполнению se, и операция аборта имеет дополнительный параметр i, обозначающий момент, в который она произошла. Предположим, мы хотим доказать, что после аборта $T1$ на некотором узле ни одна транзакция не читает значения, записанные $T1$ в этом узле. Эта лемма представляется следующим образом (в математической форме, близкой к ее реализации в PVS):

$$\forall\ se, T1, x, i : Aborts(T1, sf(x), AS(se), i) \Rightarrow$$
$$\forall\ T2, j : j > i \Rightarrow$$
$$AS(se)[j] \neq (Read(T2, x, T1), sf(x)).$$

Доказательство этой леммы здесь не приводится.

Верификация сериализуемости. В работе [3] мы представили метод для верификации конфликтной сериализуемости, основанный на *конфликтных отношениях и сохраняющих конфликт в-штампов* (временны́х штампов — *timestamps*). В этом методе для всех расписаний (состоящих из фиксированных операций), имеющих сохраняющий конфликт в-штамп, доказана их конфликтная сериализуемость. Здесь мы расширяем этот метод для верификации отказоустойчивой конфликтной сериализуемости. Новое определение отношения конфликта для расписания S будет следующим: пара ($T1$, $T2$) принадлежит $Conflict(S)$, если $T1 \neq T2$, выполняются $Commits(T1, S)$, $Commits(T2, S)$ и S включает конфликтующие операции $a1$ транзакции $T1$ и $a2$ транзакции $T2$, такие что $a1$ предшествует $a2$.

В-штамп TS, т. е. функция из множества транзакций в множество натуральных чисел, называется *инъективным* и *сохраняющим конфликт* для S, если он назначает разные значения всем транзакциям, фиксирующимся в S, и выполняется следующее:

$\forall\ T1, T2 : Conflict(S)(T1, T2) \Rightarrow TS(T1) < TS(T2)$.

Мы говорим, что S упорядочено, если существует в-штамп TS, инъективный и сохраняющий конфликт по отношению к S. Нетрудно доказать, что если S упорядочено, то каждый префикс его фиксированной проекции также упорядочен согласно определению в [3] (тем же самым в-штампом). Поэтому мы смогли переиспользовать наши предыдущие результаты и доказать, что любое упорядоченное расписание также является отказоустойчиво конфликтно сериализуемым. Действительно, определим в-штамп следующим образом: для расписания S $TSp(S)$ назначает каждой транзакции, фиксированной в S, индекс ее первой операции фиксирования. Инъективность TSp очевидна. Доказательство того, что он сохраняет конфликт, мы здесь не приводим.

Заключение

Мы представили формальную спецификацию системы обработки транзакций, сочетающей несколько фундаментальных протоколов контроля параллелизма и восстановления. Мы также формализовали их свойства корректности (ACID) и изучили их отношения друг к другу. Наше моделирование включает такие интересные аспекты этих протоколов, как распределение и различные виды памяти. В то же время наш подход не приводит к очень сложной модели. В результате мы смогли полностью верифицировать свойства корректности с помощью интерактивного доказывателя теорем PVS. Верификация сериализуемости была значительно упрощена переиспользованием наших ранее выполненных спецификаций и доказательств в PVS.

Список литературы

1. **Gray J. N. and Reuter A.** Transaction Processing Concepts and Techniques. Morgan Kaufmann Publishers, Inc., 1993.
2. **Owre S., Rushby J. M. and Shankar N.** PVS: A Prototype Verification System // 11th International Conference on Automated Deduction (CADE), LNCS 607. Saratoga Springs. 1992. P. 748—752.
3. **Chkliaev D., Hooman J. and Van der Stok P.** Serializability Preserving Extensions of Concurrency Control Protocols. // Third International Conference "Perspectives of System Informatics", LNCS 1755. Novosibirsk, 1999. P. 180—193.
4. **Bernstein P. A., Hadzilacos V. and Goodman N.** Concurrency Control and Recovery in Database Systems. Addison-Wesley Publishing Comp., 1987.
5. **Kuo D.** Model and Verification of a Data Manager Based on ARIES. // ACM Transactions on Database Systems. 1996. Vol. 21(4). P. 427—479.
6. **Babaoglu O. and Toneg S.** Non-Blocking Atomic Commitment. Distributed Systems. Addison-Wesley Publishing Comp., 1993. P. 147—168.

УДК 004.652.4

Ю. И. Рогозов, д-р техн. наук, проф., зав. каф.,
А. С. Свиридов, канд. техн. наук, доц.,
С. А. Кучеров, ассистент,
Технологический институт
Южного федерального университета, г. Таганрог
e-mail: rogozov@tsure.ru,
e-mail: sviridov@tsure.ru,
e-mail: sergey.kutcherov@gmail.com

Метод построения структурно-независимых баз данных с использованием реляционных технологий

Предложен метод, позволяющий разрабатывать различные структурно-независимые базы данных (БД) с использованием реляционных технологий. Сформулированы требования, которым должна удовлетворять структурно-независимая БД, рассмотрены различные варианты реализации метода построения структурно-независимых БД.

Ключевые слова: *изменчивость данных, метод построения баз данных, модель данных, структурно-независимая база данных*

Введение

В настоящее время информационные системы работают с данными сложной структуры с высокой степенью изменчивости. Нередко приходится проектировать базы данных (БД) в условиях отсутствия априорного описания структуры хранимой информации. Применение реляционных технологий в такой ситуации становится затруднительным: изменчивость структуры хранимой информации затрагивает физический уровень реализации, что требует перепроектирования всей БД. В результате появляются дополнительные затраты как временные, так и финансовые. Чтобы снизить затраты и продолжить использование реляционных технологий, некоторые разработчики и исследователи создают модели БД с независимым от логической структуры хранимой информации физическим уровнем реализации [1—4].

В данной статье предлагается метод, позволяющий в зависимости от конкретной мотивации разработчика получать различные структурно-независимые БД с использованием реляционных технологий. Для формулировки метода проводится анализ физического уровня реализации реляционной модели и модели *Entity-Attribute-Value*, который позволяет выявить требования к структурно-независимым БД. Определяется последовательность шагов, которая приводит к соблюдению выявленных требований и получению физической структуры БД.

Анализ моделей данных

Проведем анализ реализаций моделей данных с точки зрения независимости физической структуры БД от логической структуры хранимых данных. Для наглядности особенностей каждую из реализаций моделей данных будем рассматривать с помощью трехмерных матричных представлений в базисе "сущность — экземпляр — атрибут".

Основу реализации *реляционной модели* [5] составляет хранение данных в виде таблиц (сущностей) с конечным набором полей (атрибутов) как на логическом, так и на физическом уровне. Состав таблиц и связи между ними специфицируются на этапе проектирования. Обозначим сплошной линией направления роста БД, влекущие изменения физической структуры, а штриховой линией — не влекущие никаких изменений.

Используя выбранное обозначение и базис "сущность — экземпляр — атрибут", реализацию реляционной модели можно пространственно представить в виде некоторой трехмерной фигуры произвольной формы (рис. 1). В ячейках такой фигуры хранятся значения атрибутов для экземпляров сущностей.

На основании рис. 1 можно сделать вывод, что в реляционной модели физическая структура БД не зависит только от числа экземпляров сущностей. Это обусловлено явным хранением метаданных, описывающих хранимые данные, в виде имен таблиц, названий полей, связей между таблицами. Реляционная модель не дает возможности изме-

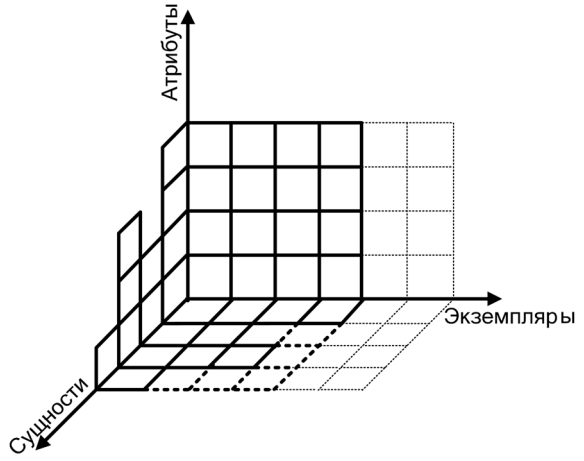


Рис. 1. Пространственное представление реализации реляционной модели данных в базисе "сущность—экземпляр—атрибут"

нения метаданных в любой момент использования БД без вмешательства на физическом уровне. Это объясняет ее неприменимость для хранения данных с изменчивой структурой.

В отличие от реляционной в модели данных *Entity-Attribute-Value* (EAV) [6—8] имеется возможность изменения части метаданных (атрибутов сущностей) во время использования БД. Реализация данной возможности заключается в непосредственном хранении изменяемых метаданных внутри самой БД. Для хранения атрибутов и их описаний создается специальная таблица, значения атрибутов хранятся отдельно [9, 10]. В то же время модель данных EAV позиционируется как узкоспециализированная и применяется в областях с заранее известным неизменным набором сущностей и частью их атрибутов: в медицинских информационных системах и системах электронной коммерции. Для определения направлений роста БД, реализованных по этой модели, воспользуемся трехмерным представлением в базисе "сущность — экземпляр — атрибут" (рис. 2).

На основании рис. 2 можно сделать вывод, что физическая структура БД не зависит от изменения числа атрибутов и экземпляров сущностей. При этом различные экземпляры одной сущности могут содержать различное число атрибутов. Иными словами, базис "экземпляр — атрибут" — это разреженная матрица, хранящая только действительные значения атрибутов. Снижение уровня зависимости физической структуры БД от структуры хранимых данных обусловливает целесообразность использования возможностей модели EAV для построения структурно-независимых БД.

Представление структуры данных в виде разреженных матриц позволяет обеспечить возможность ее динамического изменения, а непосредственное хранение метаданных внутри БД — сделать независимыми физический и логический уровни реализации. Использование этих утверждений для реализации структурно-независимой БД приводит к следующему ее трехмерному представлению в базисе "сущность — экземпляр — атрибут" (рис. 3).

Рис. 3 показывает, что для структурно-независимых БД отсутствует необходимость априорного описания предметной области — они готовы к использованию, даже если не содержат ни одной сущности и ни одного атрибута. Достигается полное разделение физического и логического уровня реализации, а именно:

- логическая структура, характеризующая пользовательские данные, описывается не физической структурой БД, а непосредственно хранимыми в ней метаданными;
- физическая структура БД является неизменной и для ее создания могут быть использованы реляционные технологии.

Проведенный анализ моделей данных позволяет сформулировать ***требования***, которым должна удовлетворять структурно-независимая БД, реализуемая с использованием реляционных технологий:

1. *Метаданные, определяющие логическую структуру БД, хранятся непосредственно внутри БД как наборы данных в реляционных таблицах и составляют подсхему метаданных.* Отказ от явного определения метаданных в виде таблиц, полей и связей между таблицами обеспечивает независимость между логической и физической структурой.

2. *Данные группируются по типам в таблицы, хранящие значения атрибутов с указанием принадлежности в виде ссылок на таблицы метаданных, и составляют соответствующую подсхему.* Имя поля, в котором хранятся значения, не является именем атрибута.

3. *Структура данных в БД представляется с помощью разреженных матриц.* Логическая структура

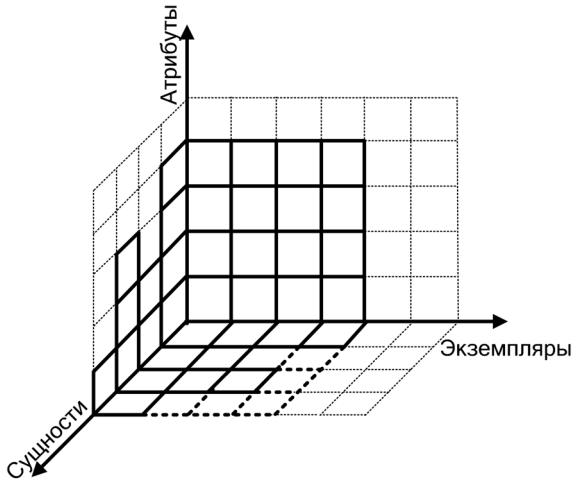


Рис. 2. Пространственное представление реализации модели данных EAV в базисе "сущность—экземпляр—атрибут"

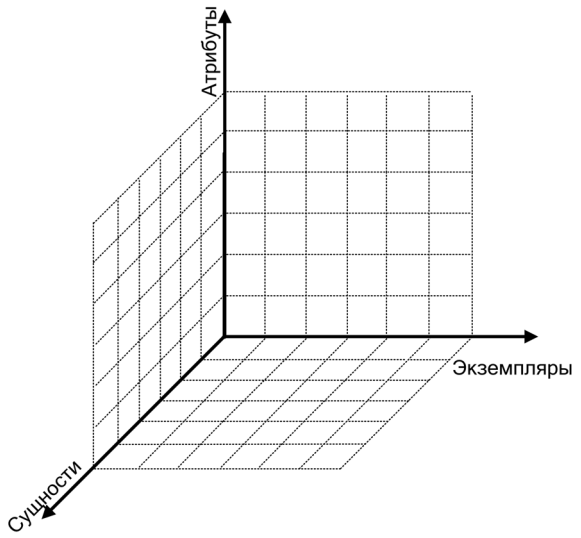


Рис. 3. Пространственное представление структурно-независимой БД в базисе "сущность—экземпляр—атрибут"

данных формируется "по факту", т. е. содержит только сущности и атрибуты, реально используемые в системе, и может быть дополнена или изменена в любой момент времени.

4. *Общее число таблиц базы данных на физическом уровне фиксированно и зависит только от числа таблиц метаданных и типов используемых данных и не зависит от логической структуры хранимых данных.* Ограничение числа таблиц на физическом уровне является ключом к повышению производительности и сокращению сложности запросов к БД и поддержанию полученных показателей во время ее эксплуатации.

Приведенные требования выполняются большинством известных моделей структурно-независимых БД с применением реляционных технологий [1—4], однако каждая из моделей соответствует конкретной мотивации разработчика.

Метод построения структурно-независимых баз данных

Независимо от мотивации разработчика имеет место конечная последовательность шагов, приводящая к получению необходимого набора связанных реляционных таблиц, отвечающего изложенным выше требованиям. Данный набор таблиц является физическим уровнем структурно-независимых БД, а последовательность шагов представляет собой ***метод построения структурно-независимых баз данных с использованием реляционных технологий.***

Для выполнения ***первого требования*** необходимо:

1. *Определить набор метаданных, необходимых для описания логической структуры БД.* Говоря языком реляционных БД, следует определить имена полей, в которых в виде значений будут храниться метаданные, описывающие пользовательские данные.

2. *Определить структуру метаданных и реализовать их в виде неизменного набора связанных реляционных таблиц, которые составляют подсхему метаданных.* Данный шаг включает в себя:

- разделение сформированного набора метаданных на группы по принадлежности (например, описывающие сущности и атрибуты, описывающие связи между сущностями);
- создание для каждой такой группы отдельной реляционной таблицы;
- составление спецификации связей между таблицами (например, между таблицей связей и таблицами описания сущностей и атрибутов).

Для выполнения ***второго требования*** необходимо:

3. *Определить типы данных, которые будут использоваться для хранения значений атрибутов.*

4. *Разработать подсхему данных в виде конечного набора однотипных не связанных между собой реляционных таблиц для хранения значений атрибутов.* Число таблиц соответствует числу используемых типов данных. Выделение отдельных таблиц на каждый тип необходимо для повышения производительности за счет упразднения процедур преобразования типов. Хранить значения целесообразно в виде троек "сущность—атрибут—значение", где первые два элемента — ссылки на записи в таблицах метаданных. Это позволяет представить структуру данных в виде разреженных матриц.

Для ***окончательного формирования структурно-независимой базы данных*** необходимо:

5. *Реализовать физическую структуру БД.* Следует определить и специфицировать связи между таблицами из подсхемы метаданных и подсхемы данных, реализовать полученную структуру в рамках конкретной реляционной СУБД.

Выполнение ***третьего*** и ***четвертого требований*** является следствием выполнения первых двух:

- представление хранимой структуры данных в виде разреженных матриц реализуется хранением значений в виде троек (разреженная матрица экземпляров) и хранением метаданных внутри БД (разреженные матрицы сущностей и атрибутов);
- независимость между физическим и логическим уровнем реализуется благодаря хранению метаданных внутри БД.

Применение изложенной выше последовательности шагов предваряет разработку пользовательских (логических) структур данных, выполняемую по определяемой разработчиком БД технологии.

Сформулированный метод позволяет получать различные эффективные структурно-независимые БД с использованием реляционных технологий в зависимости от конкретной мотивации разработчика. Для доказательства данного утверждения рассмотрим несколько примеров получения уже известных структур на основе данного метода, а также приведем собственный вариант.

Пример 1. Пусть *мотивацией* разработчиков является создание универсальной реляционной структуры для обработки и хранения слабоструктурированных данных в терминах объектно-ориентированной технологии — в виде классов и объектов. Согласно методу получение данной структуры выглядит следующим образом.

1. Выделяется набор метаданных:

- имена классов объектов и атрибутов;
- иерархии классов;
- связи между объектами;
- объекты классов.

2. Структура таблиц метаданных формируется согласно логической принадлежности метаданных:

- справочник классов объектов (уникальный id; имя класса объектов; id родительского класса,

который берется из этой же таблицы) — таблица "описания классов";
- справочник атрибутов (уникальный id; id класса; имя атрибута; тип атрибута; длина значения, хранимого атрибутом) — таблица "описания атрибутов";
- справочник связей между объектами (название связи; id первого связываемого объекта; id второго связываемого объекта) — таблица "связи";
- справочник объектов (уникальный id, он же id объекта; id класса объекта) — таблица "объекты".

3. Для представления модели БД на концептуальном уровне достаточно одного типа данных — строки.

4. Подсхема данных реализуется в виде одной таблицы — "значения атрибутов" со следующим набором полей: уникальный id, id объекта, id атрибута, имя атрибута, значение.

5. Связи между таблицами подсхемы метаданных и данных организуются по ключевым значениям уникальных id.

На основании сформулированного метода получена модель структуры БД, называемая *расширенной реляционной схемой для обработки квазиструктурированных данных*, которая описана в литературе [1]. Мотивация разработчиков моделей [2] и [3] совпадает с мотивацией разработчиков описанной выше модели. Основные отличия состоят в наборах метаданных и их структуре. Поэтому модели, представленные в работах [2, 3], можно описать аналогичным образом.

Пример 2. Пусть *мотивацией* разработчиков является создание универсальной реляционной структуры для хранения XML-данных в терминах объектно-ориентированной технологии. Согласно методу получение данной структуры выглядит следующим образом.

1. Выделяется набор метаданных:
- имена классов объектов и атрибутов;
- разрешенные связи между классами;
- связи между объектами.

2. Структура таблиц метаданных формируется согласно логической принадлежности метаданных:
- справочник классов объектов (уникальный id; имя класса объектов; xsd-схема объекта) — таблица "классы";
- справочник разрешенных связей между классами (уникальный id, название связи; id первого связываемого класса; id второго связываемого класса) — таблица "типы связей";
- справочник связей между объектами (id разрешенной связи; id первого связываемого объекта; id второго связываемого объекта) — таблица "связи объектов".

3. Использование XML-технологий позволяет хранить все данные в виде XML-объектов. Для этого выбирается большой двоичный тип данных.

4. Подсхема данных реализуется в виде одной таблицы — "объекты" следующим набором полей: уникальный id, id объекта, значение.

5. Связи между таблицами подсхемы метаданных и данных организуются по ключевым значениям уникальных id.

На основании сформулированного метода получена модель структуры БД, называемая *моделью хранения данных с применением XML-схемы*, которая описана в литературе [4].

Таким образом, метод построения структурно-независимых БД с использованием реляционных технологий позволяет получать различные эффективные модели БД в зависимости от конкретной мотивации разработчика.

Структурно-независимая база данных SIDB

Рассмотрим еще один пример использования предлагаемого метода для построения модели структурно-независимой БД. Эта модель носит название SIDB (*Structure-Independent Database*). *Мотивацией* в данном случае является хранение реляционных структур данных с высокой степенью изменчивости в БД со статической физической структурой. Согласно методу получение данной структуры выглядит следующим образом.

1. Состав метаданных, который необходим для SIDB, включает в себя:
- имена сущностей и атрибутов;
- связи сущность — атрибут;
- связи сущность — сущность;
- уникальные синонимы, позволяющие пользователю идентифицировать сущности и атрибуты (для исключения ошибок пользователя в случае совпадения имен сущностей или атрибутов, хранимых в БД);
- краткое описание атрибутов и сущностей;
- признак логического удаления;
- порядок, используемый при пользовательской сортировке;
- иерархии структур сущностей и атрибутов.

2. Метаданные можно подразделить на следующие группы с соответствующими таблицами:
- иерархический справочник, включающий в себя все метаданные, за исключением определяющих связи сущность — атрибуты и сущность — сущность. Метаданные, хранящиеся в этом справочнике, могут быть применены с одинаковой эффективностью как к сущностям, так и к атрибутам, поэтому могут быть представлены одной таблицей — tSprDicrionary. Данная таблица содержит следующие поля: уникальный id; имя сущности или атрибута; уникальный синоним; описание; идентификатор ближайшего предка, определяющий иерархию структур; идентификатор главного предка, определяющий

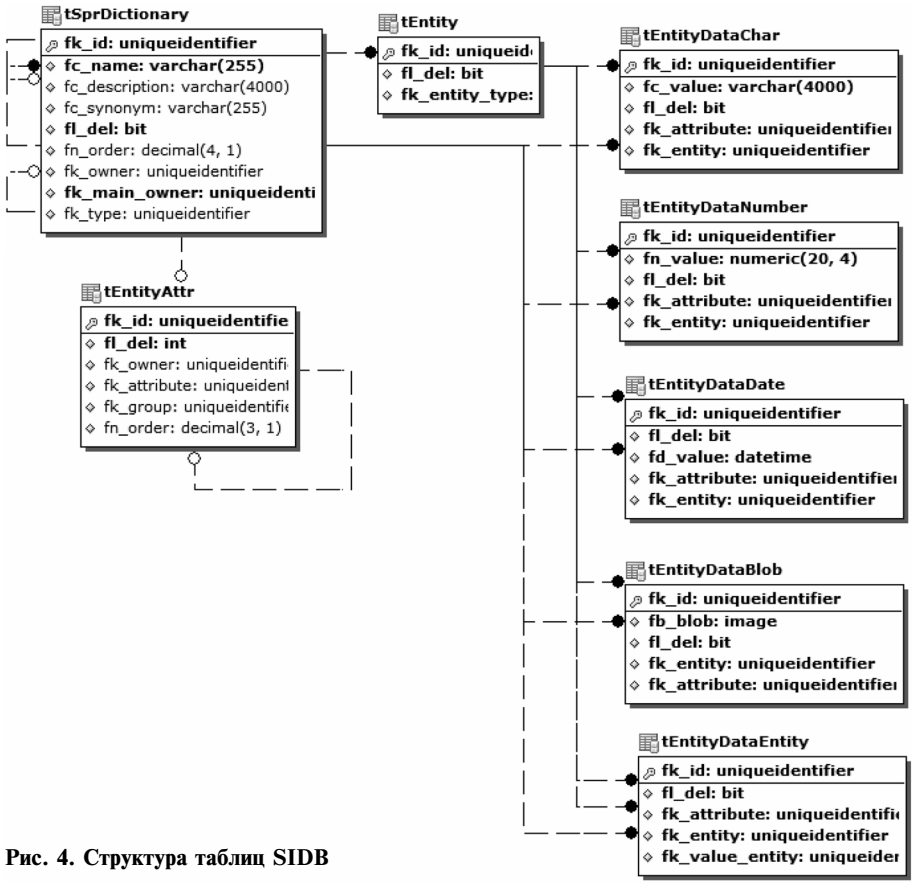


Рис. 4. Структура таблиц SIDB

принадлежность записи к группе логических справочников; идентификатор типа данных атрибута, признак логического удаления; пользовательский порядок сортировки;

- справочник атрибутов сущностей, записи в котором будут ссылаться на связываемые элементы иерархического справочника. Реализуется в виде таблицы tEntity Attr с полями: уникальный id, id сущности, которой принадлежит атрибут, id атрибута, пользовательский порядок сортировки, признак логического удаления;
- справочник экземпляров сущностей, ссылающийся на элементы иерархического справочника. Реализуется в виде таблицы tEntity с полями: уникальный id (он же идентификатор экземпляра), id сущности, признак логического удаления.

Специфика метаданных, описывающих связь сущность — сущность, заключается в их появлении в момент возникновения связи непосредственно между экземплярами сущностей. Целесообразно ввести соответствующий тип данных — ссылка на сущность — и хранить этот вид метаданных в подсхеме данных.

3. Набор типов данных, с которыми работает SIDB, включает в себя: строки, числа, даты, большие двоичные файлы, а также тип данных "Entity" — ссылка на сущность, используемый в качестве метаданных.

4. Для хранения данных целесообразно использовать тройки сущность — атрибут — значение (в работах [1—3] применяются тройки объект — атрибут — значение), причем сущности и атрибуты в таких тройках задаются неявно в виде ссылок на элементы иерархического справочника. Подсхема данных включает в себя следующие таблицы:

- tEntityDataChar — строковые данные;
- tEntityDataNumber — числовые данные;
- tEntityDataDate — дата и время;
- tEntityDataBlob — большие двоичные файлы;
- tEntityDataEntity — хранение связи сущность — сущность.

5. Связи между таблицами подсхемы метаданных и данных организуются по ключевым значениям уникальных id.

Полученная в результате применения метода модель SIDB (рис. 4) позволяет хранить реляционные структуры данных с высокой степенью изменчивости любой сложности.

Об эффективности использования SIDB можно судить на примере реализации программного средства "ПРИМИУС" [11].

Заключение

Метод, предложенный в данной статье, позволяет строить с использованием реляционных технологий структурно-независимые БД, обладающие следующими достоинствами:

- ограничение структурной сложности запросов к БД за счет фиксации числа реляционных таблиц на физическом уровне;
- повышение быстродействия операций с данными. Реляционные технологии изначально предназначены для работы со статическими структурами данных и обладают богатым набором средств повышения производительности;
- гибкость. Отсутствие зависимости между физическим и логическим уровнями БД, реализуемой с помощью предложенного метода, позволяет обеспечить максимальный уровень гибкости;
- низкие затраты на обслуживание. Отказ от привлечения высококвалифицированных специалистов, занимающихся поддержкой работоспособности физического уровня, позволяет экономить время и деньги.

В результате применения метода, сформулированного в данной статье, может быть получено множество различных структурно-независимых БД на основе реляционных технологий. Реализация каждого из шагов метода зависит от мотивации разработчика и не ограничивается рассмотренными в данной статье примерами.

Список литературы

1. **Палей Д.** Моделирование квазиструктурированных данных // Открытые системы. 2002. № 9. С. 57—64.
2. **Тенцер А.** База данных — хранилище объектов // Компьютер-Пресс. 2001. №8. С. 144—145.
3. **Банников Н.** Проект "Универсальная база данных". URL:www.stikriz.narod.ru
4. **Полищук Ю. В., Черных Т. А.** Моделирование подсистем хранения информации, ориентированных на хранение квазиструктурированных объектов // Информационные технологии. 2009. № 1. С. 66—71.
5. **Godd E. F.** A Relational Model of Data for Large Shared Data Banks // Communications of the ACM. 1970. Vol. 13. N 6. Русская версия статьи на www.citforum.ru
6. **Stead W. W., Hammond W. E., Straube M. J.** A Chartless Record — Is It Adequate? // Proceedings of the Annual Symposium on Computer Application in Medical Care. 1982. P. 89—94.
7. **Jacob Anhoj.** Generic Design of Web-Based Clinical Databases // Journal of Medical Internet Research 2004. URL: http://www.jmir.org/2003/4/e27/
8. **Nadkarni P.** An Introduction to Entity-Attribute-Value Design for Generic Clinical Study Data Management Systems // Center for Medical Informatics, Yale University Medical School. URL: http://med.yale.edu/
9. **Dinn V., Nadkarni P.** Guidelines for the effective use of entity—attribute—value modeling for biomedical databases // Elsevier, Ireland. International journal of medical informatics. 2007. N 76. P. 769—779.
10. **Brandt C., Morse R., Matthews K., Sun K., Deshpande A., Gadagkar R., Cohen D., Miller P., Nadkarni P.** Metadata — driven creation of data marts from an EAV-modeled clinical research database // Elsevier, Ireland. International journal of medical informatics. 2002. N 65. P. 225—241.
11. **Рогозов Ю. И., Бутенков С. А., Свиридов А. С.** Метод создания инструментальных средств разработки информационных систем // Информационно-измерительные и управляющие системы. 2008. Т. 6. № 3. С. 76—84.

УДК 004.9

Е. А. Рубцов, ст. преподаватель,
"МАТИ" — РГТУ имени К. Э. Циолковского,
e-mail: rea@inistek.ru

Модель данных для сбора, хранения и обработки информации о существующих во времени объектах различных классов

Описывается модель данных, объединяющая некоторые принципы объектно-ориентированного и темпорального подходов к разработке информационных систем.

Ключевые слова: *модель данных, объектно-ориентированный подход, темпоральный подход*

Введение

Основные принципы классификации объектов были заложены еще в аристотелевской логике в виде такой формы логического мышления как понятие. Любое понятие имеет содержание — мыслимые в понятии общие и существенные признаки предметов, и объем — охватываемые им предметы мысли [1]. Понятия могут находиться в различных отношениях друг с другом: равнозначность, подчинение, пересечение, соподчинение и др. В информационных технологиях эти идеи нашли отражение в объектно-ориентированном подходе [2], который в настоящее время является одним из наиболее распространенных подходов к разработке программного обеспечения. Однако ни в классической логике, ни в объектно-ориентированном подходе не учитывается такой важнейший фактор, как время.

Принципы учета времени существования объектов были отражены в таком направлении развития информационных систем, как темпоральные базы данных [3], которые по разным причинам не получили широкого распространения [4]. При этом, поскольку учет времени существования объектов крайне важен для информационных систем, он реализуется в настоящее время силами разработчиков и администраторов баз данных с использованием стандартных инструментов реляционного и объектно-ориентированного подходов. Можно сделать вывод о существующей необходимости стандартизации и реализации темпоральной составляющей в промышленных базах данных.

В статье предлагается модель данных, совмещающая свойства объектно-ориентированной и темпоральной моделей данных. В разделе 1 описывается структура предлагаемой модели, в разделе 2 определяется набор операций, которые могут быть использованы для обработки данных в рамках предлагаемой модели.

1. Структура модели

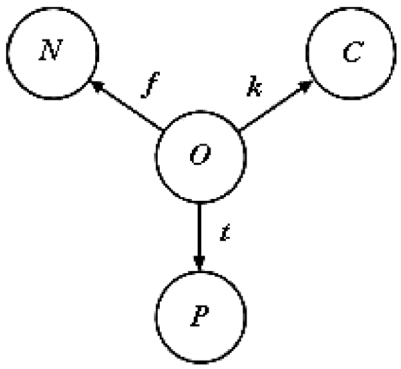
Множество всех объектов обозначим O. Перенумеруем объекты, задав инъективное отображение f из O в N, где N — множество натуральных чисел. Номер, связанный с объектом отображением f, будем называть *номером объекта*.

Помимо этого определим частично упорядоченное множество классов (частичная упорядоченность необходима для определения иерархии классов). Обозначим это множество C. Для классификации объектов зададим отображение g из множества O во множество C.

Для описания существования объектов во времени определим множество периодов времени P, представляющее собой множество упорядоченных пар целых чисел (включая символы $-\infty$, $+\infty$), в которых первое число меньше или равно второму. Произвольный элемент этого множества будем называть *периодом времени* или просто *периодом*. Для связывания объектов с периодами времени зададим отображение t из множества O во множество P. Период времени, связанный с определенным объектом с помощью отображения t, будем называть *периодом существования объекта*. Первое число из периода существования объекта будем называть *моментом появления объекта*, второе число — *моментом исчезновения объекта*. Период, у которого первое число равно второму, будем называть *элементарным периодом*. Схематично эта модель изображена на рисунке.

Для хранения данных в рамках предлагаемой модели необходимо создать два хранилища данных: *хранилище классов* и *хранилище объектов*. В упрощенном виде структуру хранилища классов можно представить следующим образом:

Код класса	**Код класса верхнего уровня (Код класса-предка)**



Структура модели данных

Структуру хранилища объектов можно представить так:

Номер объекта	**Код класса**	**Момент появления объекта**	**Момент исчезновения объекта**

2. Операции

Введем несколько обозначений и определим некоторые понятия:

o — произвольный элемент множества O; $<o>$ — вектор произвольных элементов множества O (вектор объектов); U — произвольное подмножество множества O; $<U>$ — вектор произвольных подмножеств множества O; k — произвольный элемент множества C (произвольный класс); $<k>$ — вектор произвольных элементов множества C (вектор классов); p — произвольный элемент множества P (произвольный период времени); $<p>$ — вектор произвольных элементов множества P (вектор периодов); Z — множество целых чисел, включающее в том числе символы $-\infty$, $+\infty$; $<z>$ — вектор произвольных элементов множества Z.

Будем говорить, что *объект принадлежит классу* k, если этому объекту с помощью отображения g поставлен в соответствие либо класс k, либо любой другой класс k' такой, что $k' \leqslant k$.

Будем говорить, что периоды $p1$ и $p2$ *пересекаются*, если первое число из периода $p1$ меньше или равно второму числу из периода $p2$, а второе число из периода $p1$ больше или равно первому числу из периода $p2$.

Будем говорить, что объект *принадлежит периоду* p, если период существования этого объекта пересекается с периодом p.

Операции над объектами можно разделить на три группы:

- операции для добавления, изменения и удаления классов;
- операции для добавления, изменения и удаления объектов;
- операции для выборки данных.

Первую группу операций реализуют следующие предлагаемые функции.

ДобавитьКласс(k, k_p) — добавляет в хранилище классов новый класс k, который является наследником существующего класса k_p.

ИзменитьПредка(k, k_p) — назначает классу k в качестве класса верхнего уровня класс k_p.

УдалитьКласс(k) — удаляет класс k из хранилища классов.

Вторую группу операций образуют следующие предлагаемые функции.

Добавить(k, p) — добавляет в хранилище объектов объект класса k с периодом p.

ИзменитьКласс(n, k) — заменяет у объекта с номером n его текущий класс на класс k.

ИзменитьПериод(n, p) — заменяет у объекта с номером n его текущий период существования на период p.

Удалить(n) — удаляет из хранилища объектов объект с номером n.

Третью группу образуют следующие функции.

ОбъектыКласса(k, U) — возвращает множество всех объектов, которые одновременно являются элементами множества U и принадлежат классу k.

ОбъектыКлассов($<k>$, U) — возвращает вектор $<U>$ такой, что U_i = *ОбъектыКласса*(k_i, U).

ОбъектыПериода(p, U) — возвращает множество всех объектов, которые одновременно являются элементами множества U и период существования которых пересекается с периодом p.

ОбъектыПериодов($<p>$, U) — возвращает вектор $<U>$ такой, что U_i = *ОбъектыПериода*(p_i, U).

Объем(U) — возвращает объем объектов в множестве U.

Объемы($<U>$) — возвращает вектор $<z>$ такой, что z_i = *Объем*(U_i).

ДлинаПериода(p) — возвращает длину периода, рассчитанную путем вычитания первого числа периода из второго с дальнейшим прибавлением к полученному результату единицы.

ДлиныПериодов($<p>$) — возвращает вектор $<z>$ такой, что z_i = *ДлинаПериода*(p_i).

Максимум($<z>$) — возвращает максимальное число из всех элементов вектора $<z>$.

Минимум($<z>$) — возвращает минимальное число из всех элементов вектора $<z>$.

Среднее($<z>$) — возвращает среднее значение всех элементов вектора $<z>$.

Сумма($<z>$) — возвращает сумму всех элементов вектора $<z>$.

ЭлементарныеПериоды(p) — возвращает вектор $<p>$, состоящий из всех элементарных периодов, пересекающихся с периодом p.

Упорядочить(U) — возвращает вектор объектов, состоящий из элементов множества U, упорядоченных по их номерам.

ПериодОбъекта($<o>$) — возвращает период существования заданного объекта.

ПериодыОбъектов($<o>$) — возвращает вектор периодов существования объектов, в котором p_i = *ПериодОбъекта*(o_i).

Используя эти функции, можно реализовать различные запросы к базе данных, хранящей информацию об объектах различных классов и периодах их существования. Ниже в качестве примера дается перечень некоторых запросов к базе данных. Сначала описывается требуемый результат, а затем идет текст запроса, созданного на базе предложенных функций и возвращающий затребованные данные.

Объем объектов класса k, существовавших в рамках заданного периода времени p:

Объем(ОбъектыПериода(p, ОбъектыКласса(k, U)))

Максимальный объем объектов класса k, существовавших одновременно в рамках заданного периода времени p:

Максимум(Объемы(ОбъектыПериодов(ЭлементарныеПериоды(p), ОбъектыКласса(k, U))))

Средняя длина периода существования объектов класса k, существовавших в рамках заданного периода времени p:

Среднее(ДлиныПериодов(ПериодыОбъектов(Упорядочить(ОбъектыПериода(p, ОбъектыКласса(k, U))))))

Заключение

Предложенная модель данных объединяет свойства темпоральных и объектно-ориентированных баз данных и может найти применение при разработке корпоративных и аналитических информационных систем. Основанный на этой модели подход был использован при создании информационной системы для производственного предприятия, выпускающего большой ассортимент одежды разных размеров и цветов [5]. Благодаря возможности указания даты появления и исчезновения объектов появилась возможность отслеживать и использовать в оперативном учете информацию о новых и устаревших моделях, цветах, размерах, а также о новых и устаревших связях между моделями, цветами и размерами и др.

Еще один вариант использования предлагаемого подхода — его применение для анализа активности пользователей корпоративных информационных систем на основании данных о подключениях различных групп пользователей к серверу.

Дальнейшим развитием изложенных в статье идей стало добавление к предложенной модели данных направленных бинарных отношений, которые также рассматриваются как существующие во времени объекты различных классов.

Список литературы

1. **Иванов Е. А.** Логика. Изд. 2-е, перераб и доп. М.: БЕК, 2001. 368 с.

2. **Буч Г.** Объектно-ориентированный анализ и проектирование. М.: Вильямс, 2008. 720 с.

3. **Christian S. J.** Temporal Database Management. URL: <http://www.cs.aau.dk/~csj/Thesis/> (16.10.2010).

4. **Костенко Б. Б., Кузнецов С. Д.** История и актуальные проблемы темпоральных баз данных. URL: <http://www.citforum.ru/database/articles/temporal> (16.10.2010)

5. **Павлючков С. Е., Рубцов Е. А., Шилов В. В.** Интегрированные информационные системы как инструмент решения производственных задач // Экономика и управление в машиностроении. 2009. № 6. С. 7—9.

УДК 004.932

А. О. Левашкина, канд. техн. наук, доц.,
С. В. Поршнев, д-р техн. наук, зав. каф.
Уральский государственный
технический университет (УГТУ—УПИ)
e-mail: iconismo@gmai

Исследование возможности использования ключевых точек в задаче поиска изображений с визуально похожими объектами

Описаны результаты исследования возможности использования ключевых точек в задаче поиска изображений по визуальному сходству. Наилучшие результаты с точки зрения точности и полноты на естественных изображениях показали детекторы Laplace (Affintpoints), Hessian (Affintpoints), Ridge (Affintpoints), Harris (Affintpoints). Среди всех детекторов, показавших наилучшие результаты поиска на естественных изображениях, следует выделить Hessian-детектор, поскольку положение ключевых точек, найденных с его помощью, в большей степени соответствует положению объекта на изображении, а показатели полноты и точности в наименьшей степени зависят от числа найденных ключевых точек.

Ключевые слова: *поиск изображений, ключевые точки, детекторы ключевых точек, визуально похожие объекты*

Введение

Ключевая точка — это точка изображения, структура окрестности которой ковариантна заданным преобразованиям изображения [1]. В литературе также используют термины "контрольная точка", "точка интереса", "характеристическая точка", "точечная особенность", *"interest point"*. Ключевые точки широко используют при решении классической задачи компьютерного зрения — задачи сопоставления изображений (*image matching*), состоящей в установлении соответствий между областями изображений.

В настоящее время ключевые точки активно применяют при решении задачи поиска нечетких дубликатов (см. например [1]). Напомним, что нечеткими дубликатами называются два изображения, которые могут быть переведены друг в друга путем элементарных преобразований, таких как поворот, сдвиг, изменения угла обзора, разрешения, масштаба, освещения. Примерами нечетких дубликатов являются фотоснимки и видеокадры одной сцены, сделанные с различных ракурсов, при различном освещении, в различные моменты времени с помощью различных регистрирующих устройств. Также нечеткие дубликаты возникают при редактировании изображений. Примерами нечетких дубликатов являются изображения объекта, повернутого на различные углы. Подобные серии изображений широко используются при тестировании детекторов ключевых точек.

Отметим, что задача поиска нечетких дубликатов по своей постановке оказывается достаточно близкой к задаче поиска изображений по визуальному сходству. (Их отличие состоит в том, что в случае поиска изображений по визуальному сходству искомые изображения относятся не к одной и той же сцене, а соответствуют одной семантической категории (например, "цветы", "лес", "дом" и пр.). При этом два изображения могут иметь глобальное сходство (например пейзажи) или локальное сходство объектов, интересующих пользователя (например "цветок", "книга").) Принимая во внимание известную близость постановки обсуждаемых задач, можно ожидать, что методы решения задачи поиска нечетких дубликатов окажутся полезными при решении задачи поиска изображений по визуальному сходству.

В статье описаны результаты исследования возможности использования ключевых точек в задаче поиска изображений с визуально похожими объектами.

Для выявления ключевых точек используют различные методы-детекторы. Разработка методов сопоставления изображений с использованием множества локальных ключевых точек началась в 1981 г. с работы *H. Moravec* по стереосопоставлению на основе детектора углов [2, 3]. *C. Harris* и *M. Stephens* в 1988 г. усовершенствовали *Moravec*-детектор с целью большей устойчивости к небольшим изменениям изображения [4]. В дальнейшем *Harris*-детектор получил широкое распространение при решении многих задач по сопоставлению изображений. Обычно *Harris*-детектор называют детектором углов, однако он позволяет выделять не только углы, но и любые другие точки, в которых наблюдаются большие значения градиента во всех направлениях. *Harris*-детектор очень чувствителен к изменениям масштаба изображения, поэтому он не дает хорошего результата при сопоставлении изображений разного размера [5]. *D. Low* предложил

SIFT-детектор, который оказался менее чувствительным к локальным искажениям изображения (например, изменение точки просмотра объекта в трехмерном пространстве). Таким образом, в настоящее время наиболее известны следующие детекторы: *Harris-Laplacian* [6], *Hessian-Laplacian* [6], *Difference of Gaussian* (*DoG*) [7], *Laplacian of Gaussian* (*LoG*) [8], *SIFT* [7], *PCA-SIFT* [9] и др.

Методика сравнения детекторов

При исследовании возможности использования точечных особенностей для описания свойств объекта на изображении мы использовали методику сравнения детекторов, реализованную следующей последовательностью действий.

1. Выбор коллекции изображений, на каждом из которых присутствует один явно выраженный объект $I = \{I^1, I^2, ..., I^i, ..., I^n\}$, где I — изображение; i — номер изображения в коллекции; n — число изображений в коллекции.

2. Формирование экспертами для каждого изображения бинарной маски объекта $M = \{M^1, M^2, ..., M^i, ..., M^n\}$, где M — бинарная маска объекта (пиксели, принадлежащие объекту, имеют значение 1, принадлежащие фону — значение 0).

3. Определение ключевых точек, принадлежащих объекту.

4. Вычисление критериев качества:

- полноты нахождения объекта — доли ключевых точек, входящих в состав объекта, от общего числа пикселей, принадлежащих объекту;
- точности нахождения объекта — доли ключевых точек, входящих в состав объекта, от общего числа найденных ключевых точек.

В экспериментах были использованы следующие коллекции изображений:

1. Набор № 1 состоял из 50 цветных изображений. На каждом изображении присутствовал один хорошо выраженный объект, расположенный не по центру изображения. Данный набор использовался при исследовании влияния положения объекта относительно центра естественного изображения на точность и полноту нахождения объекта.

2. Набор № 2 состоял из 49 изображений размером 200 × 200 пикселей. В процентах от площади всего изображения средняя площадь объекта на изображении составляет $\overline{S} = 3{,}7 \pm 0{,}1$. Данный набор использовался при исследовании влияния положения объекта относительно центра искусственно сгенерированного цветного изображения на точность и полноту нахождения объекта.

3. Набор № 3 состоял из 72 изображений, на каждом из которых присутствует один выраженный объект, расположенный не по центру. В процентах от площади всего изображения средняя площадь объекта составляет $\overline{S} = 6{,}2 \pm 0{,}3$. Данный набор использовался при исследовании влияния небольшого размера объекта на естественном изображении на точность и полноту нахождения объекта.

4. Набор № 4 состоял из 50 изображений. В процентах от площади всего изображения средняя площадь объекта составляет $\overline{S} = 0{,}802 \pm 0{,}008$. Данный набор использовался при исследовании влияния небольшого размера объекта на искусственно сгенерированном цветном изображении на точность и полноту нахождения объекта.

5. Набор № 5 состоял из 49 изображений, на которые добавлен шум, и объекты расположены не по центру. В процентах от площади всего изображения средняя площадь объекта на изображении составляет $\overline{S} = 3{,}7 \pm 0{,}1$. Данный набор использовался при исследовании влияния наличия шумов на искусственно сгенерированном цветном изображении на точность и полноту его нахождения.

Описанные выше коллекции изображений использовались для исследования влияния следующих факторов на результаты локализации объекта:

- положения объекта относительно центра для цветных естественных и искусственно сгенерированных изображений (набор № 1, № 2);
- размера объекта на цветных естественных и искусственно сгенерированных изображениях (наборы № 3, № 4);
- наличия шумов на искусственно сгенерированных цветных изображениях (набор № 5).

Результаты экспериментов по сравнению детекторов

Результаты экспериментов, проведенных в соответствии с описанной выше методикой, представлены в табл. 1—4 и на рис. 1, 2. Из табл. 1—4

Таблица 1

Точность нахождения объекта (наборы данных № 1, № 2, № 3)

N	Детектор	№ 1	№ 2	№ 3
1	*SURF*	0,136 ± 0,03	0,034 ± 0,02	0,074 ± 0,02
2	*Harris (Affintpoints)*	0,32 ± 0,03	0,07 ± 0,01	0,22 ± 0,01
3	*Laplace (Affintpoints)*	0,36 ± 0,04	0,11 ± 0,02	0,38 ± 0,02
4	*Hessian (Affintpoints)*	0,39 ± 0,04	0,15 ± 0,04	0,38 ± 0,03
5	*Ridge (Affintpoints)*	0,35 ± 0,03	0,11 ± 0,03	0,39 ± 0,02
6	*SIFT (Low)*	0,30 ± 0,04	0,23 ± 0,05	0,23 ± 0,02
7	*SIFTfast*	0,09 ± 0,01	0,046 ± 0,004	0,077 ± 0,005
8	*Kadir-Brady*	0,24 ± 0,03	0,12 ± 0,03	0,15 ± 0,02

Таблица 2

Точность нахождения объекта (наборы данных № 4, № 5)

N	Детектор	№ 4	№ 5
1	*SURF*	0,0042 ± 0,002	0,048 ± 0,003
2	*Harris (Affintpoints)*	0,045 ± 0,005	0,029 ± 0,005
3	*Laplace (Affintpoints)*	0,071 ± 0,009	0,023 ± 0,004
4	*Hessian (Affintpoints)*	0,053 ± 0,007	0,024 ± 0,004
5	*Ridge (Affintpoints)*	0,066 ± 0,009	0,021 ± 0,005
6	*SIFT (Low)*	0,15 ± 0,04	0,030 ± 0,002
7	*SIFTfast*	0,0087 ± 0,0006	0,040 ± 0,002
8	*Kadir-Brady*	0,15 ± 0,04	0,057 ± 0,005

Таблица 3

Полнота нахождения объекта (наборы данных № 1, № 2, № 3)

N	Детектор	№ 1	№ 2	№ 3
1	*SURF*	0,0066 ± 0,0002	0,0049 ± 0,0001	0,0065 ± 0,0001
2	*Harris (Affintpoints)*	0,0025 ± 0,0002	0,0034 ± 0,0005	0,0028 ± 0,0002
3	*Laplace (Affintpoints)*	0,0033 ± 0,0004	0,006 ± 0,001	0,0059 ± 0,0005
4	*Hessian (Affintpoints)*	0,0036 ± 0,0004	0,0092 ± 0,002	0,0059 ± 0,0005
5	*Ridge (Affintpoints)*	0,0032 ± 0,0004	0,006 ± 0,001	0,0060 ± 0,0004
6	*SIFT (Low)*	0,0080 ± 0,0007	0,0075 ± 0,0008	0,0102 ± 0,0005
7	*SIFTfast*	0,0084 ± 0,0009	0,0147 ± 0,001	0,0136 ± 0,0008
8	*Kadir-Brady*	0,0224 ± 0,001	0,0199 ± 0,002	0,0242 ± 0,001

Таблица 4

Полнота нахождения объекта (наборы данных № 4, № 5)

N	Детектор	№ 4	№ 5
1	*SURF*	0,0028 ± 0,0001	0,0084 ± 0,0002
2	*Harris (Affintpoints)*	0,011 ± 0,001	0,0012 ± 0,0002
3	*Laplace (Affintpoints)*	0,020 ± 0,002	0,0013 ± 0,0002
4	*Hessian (Affintpoints)*	0,0152 ± 0,002	0,0013 ± 0,0002
5	*Ridge (Affintpoints)*	0,018 ± 0,003	0,0009 ± 0,0002
6	*SIFT (Low)*	0,0044 ± 0,0004	0,0100 ± 0,0005
7	*SIFTfast*	0,012 ± 0,001	0,0139 ± 0,0005
8	*Kadir-Brady*	0,050 ± 0,004	0,037 ± 0,002

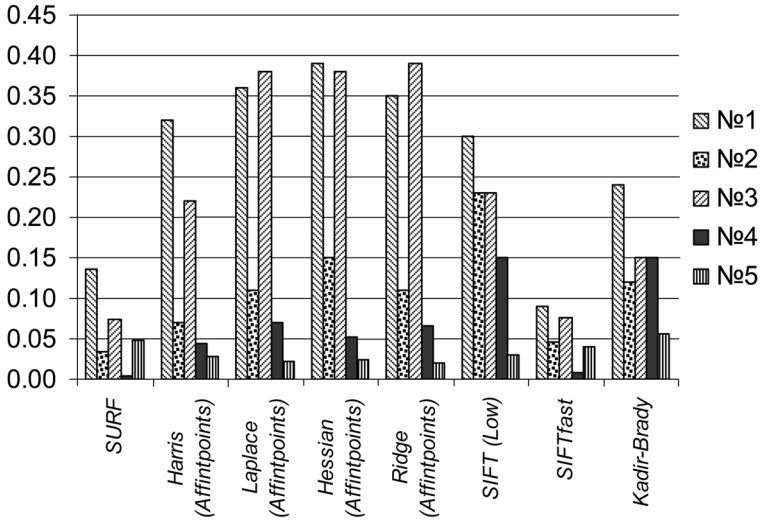


Рис. 1. Точность нахождения объекта

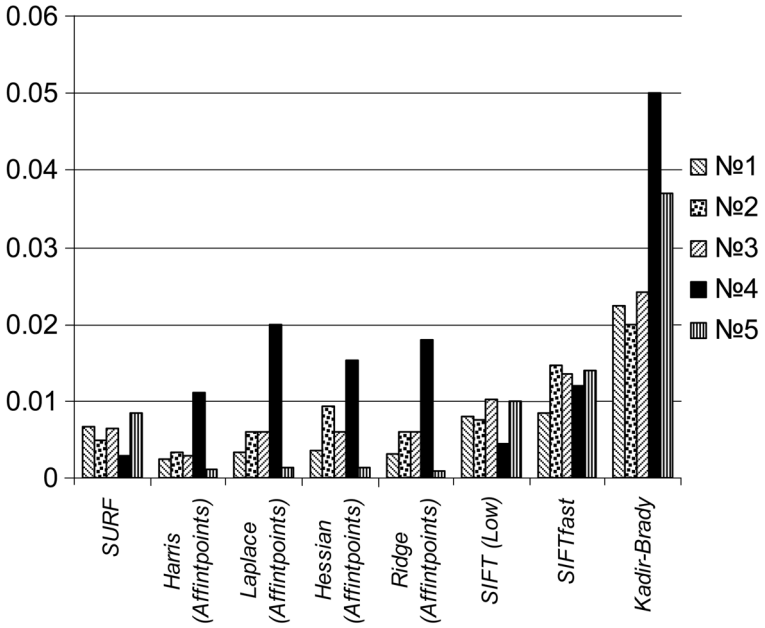


Рис. 2. Полнота нахождения объекта

видно, что на естественных изображениях (наборы данных № 1 и № 3) лучшие результаты показали детекторы: *Laplace* (*Affintpoints*), *Hessian* (*Affintpoints*), *Ridge* (*Affintpoints*), *Harris* (*Affintpoints*), которые были использованы в дальнейшем исследовании зависимости показателей полноты и точности от числа ключевых точек.

Отметим, что для выбранных детекторов необходимо задавать число ключевых точек, которые должны быть выделены на изображениях (в предыдущих экспериментах число точек равнялось 100). Поскольку возможное увеличение числа ключевых точек может привести к увеличению числа точек, которые будут зарегистрированы как принадлежащие объекту, потребовалось проведение дополнительного исследования, в ходе которого использовалась описанная выше методика сравнения детекторов. При этом число ключевых точек варьировалось от 100 до 1000 ($N = 100, 200, ..., 1000$). Результаты вычислений представлены в табл. 5, 6 и рис. 3, 4.

Из рис. 3 видно, что:

- для всех детекторов при увеличении числа вычисляемых ключевых точек снижается точность, т. е. доля найденных ключевых точек, которые входят в состав объекта;
- значения точности для *Hessian*-детектора выше, чем для других детекторов, при любом числе ключевых точек;
- для *Harris*-детектора при увеличении числа точек точность поиска резко снижается;
- *Laplace*- и *Ridge*-детекторы показали близкие результаты.

Из рис. 4 видно, что:

- для всех детекторов при увеличении числа вычисляемых ключевых точек увеличивается полнота, т. е. доля ключевых точек, входящих в состав объекта, от общего числа пикселей, принадлежащих объекту;

- значения полноты для *Hessian*-детектора выше, чем для других детекторов, при любом числе ключевых точек;
- для *Harris*-детектора при увеличении числа точек точность поиска увеличивается незначительно;
- *Laplace*- и *Ridge*-детекторы показали схожие результаты.

Таким образом, из всех рассмотренных детекторов выделяется *Hessian*-детектор, поскольку:

— положение ключевых точек, найденных с его помощью, в большей степени соответствует положению объекта на изображении;

— критерии полноты и точности для данного детектора в наименьшей степени зависят от числа найденных ключевых точек.

Таблица 5

Точность нахождения объекта

N	*Harris (Affintpoints)*	*Laplace (Affintpoints)*	*Hessian (Affintpoints)*	*Ridge (Affintpoints)*
100	0,32 ± 0,03	0,36 ± 0,04	0,39 ± 0,04	0,35 ± 0,03
200	0,20 ± 0,02	0,33 ± 0,03	0,38 ± 0,04	0,34 ± 0,03
300	0,15 ± 0,01	0,33 ± 0,03	0,37 ± 0,04	0,33 ± 0,03
400	0,12 ± 0,01	0,31 ± 0,03	0,36 ± 0,04	0,32 ± 0,03
500	0,11 ± 0,01	0,30 ± 0,03	0,34 ± 0,04	0,30 ± 0,03
600	0,108 ± 0,009	0,28 ± 0,03	0,33 ± 0,03	0,28 ± 0,03
700	0,108 ± 0,009	0,26 ± 0,02	0,31 ± 0,03	0,27 ± 0,03
800	0,108 ± 0,009	0,25 ± 0,02	0,30 ± 0,03	0,25 ± 0,02
900	0,108 ± 0,009	0,23 ± 0,02	0,29 ± 0,03	0,24 ± 0,02
1000	0,108 ± 0,009	0,22 ± 0,02	0,27 ± 0,03	0,23 ± 0,02

Использование ключевых точек в поиске изображений с визуально схожими объектами

Далее *Hessian*-детектор, обеспечивающий наилучшие результаты с точки зрения точности и полноты поиска, был использован в задаче поиска изображений с визуально схожими объектами. Здесь по ключевым точкам, находимым *Hessian*-детектором, вычислялся дескриптор *SIFT*, который, как следует из анализа литературных источников, наилучшим образом подходит для использования в поиске изображений [7].

Пусть изображению I соответствует множество дескрипторов найденных ключевых точек $\{FA\}_i^k$, где $i = \overline{1,f}$ — индекс признака ключевой точки; f — число признаков, которые вычисляются для одной ключевой

Таблица 6

Полнота нахождения объекта

N	*Harris (Affintpoints)*	*Laplace (Affintpoints)*	*Hessian (Affintpoints)*	*Ridge (Affintpoints)*
100	0,0025 ± 0,0002	0,0033 ± 0,0004	0,0036 ± 0,0004	0,0032 ± 0,0004
200	0,003 ± 0,0002	0,0058 ± 0,0006	0,0068 ± 0,0008	0,0058 ± 0,0006
300	0,0032 ± 0,0002	0,0082 ± 0,0008	0,0096 ± 0,001	0,0083 ± 0,0009
400	0,0034 ± 0,0002	0,01 ± 0,0009	0,012 ± 0,001	0,0103 ± 0,001
500	0,0037 ± 0,0001	0,0116 ± 0,001	0,014 ± 0,002	0,0118 ± 0,001
600	0,0039 ± 0,0001	0,0127 ± 0,001	0,016 ± 0,002	0,013 ± 0,001
700	0,0039 ± 0,0001	0,014 ± 0,001	0,017 ± 0,002	0,014 ± 0,001
800	0,0039 ± 0,0001	0,015 ± 0,001	0,019 ± 0,002	0,015 ± 0,001
900	0,0039 ± 0,0001	0,016 ± 0,001	0,020 ± 0,002	0,016 ± 0,001
1000	0,0039 ± 0,0001	0,016 ± 0,001	0,021 ± 0,002	0,016 ± 0,001

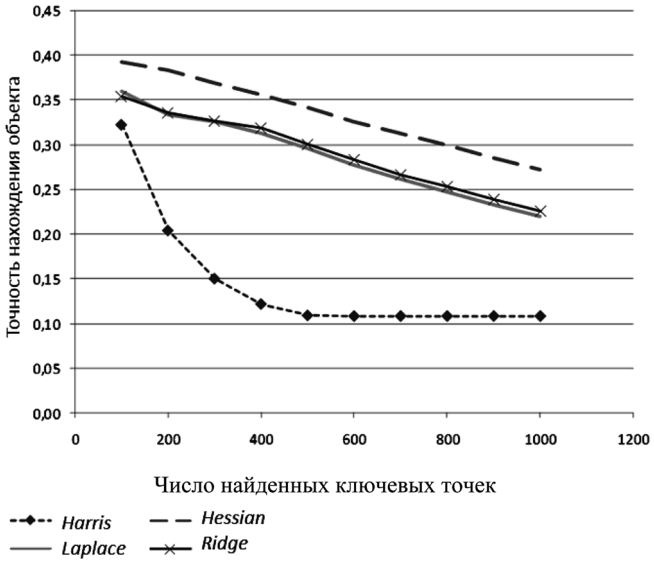


Рис. 3. График зависимости точности нахождения объекта от числа найденных ключевых точек

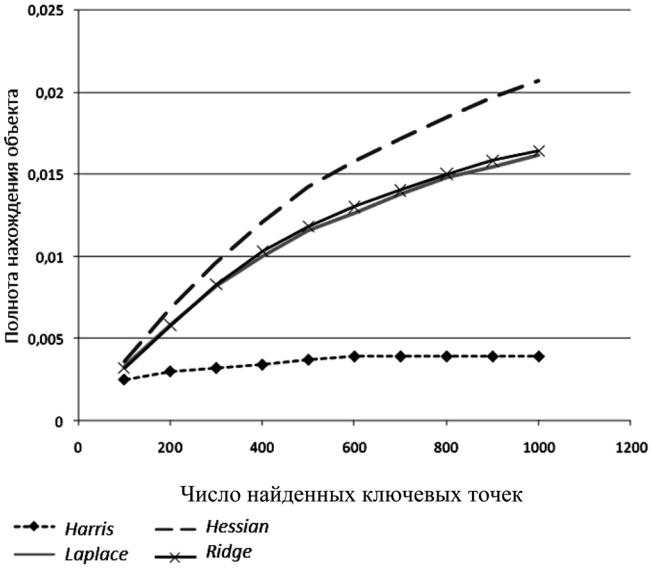


Рис. 4. График зависимости полноты нахождения объекта от числа найденных ключевых точек

точки ($f = 128$); $k = \overline{1, F}$ — индекс ключевых точек; F — число ключевых точек, которые были найдены для изображения I (для каждого изображения F разное и вычисляется автоматически).

Для вычисления сходства между двумя изображениями *I_query* и *I_from_dataset* использовался подход, который реализуется следующей последовательностью действий.

1. Вычисление множества дескрипторов для каждого изображения:

$$\{FA\_query\}_i^{k1},\ k1 = \overline{1, F1},$$

$$\{FA\_from\_dataset\}_i^{k2},\ k2 = \overline{1, F2}.$$

2. Вычисление расстояния между всеми комбинациями пар дескрипторов из множеств $\{FA\_query\}_i^{k1}$ и $\{FA\_from\_dataset\}_i^{k2}$. Результатом вычислений является матрица расстояний $D_{p,t}$, $p = \overline{1, F1}$, $t = \overline{1, F2}$.

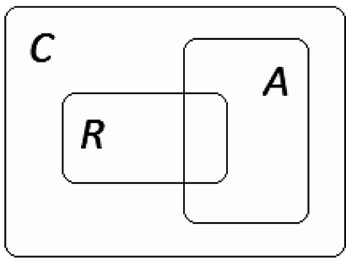


Рис. 5. Иллюстрация к вычислению критериев

Таблица 7

Критерии эффективности

Критерии	*dist*_1	*dist*_2
SR (50)	0,0723	0,1033
SP (5)	0,0381	0,3075
SP (10)	0,0367	0,2163
SP (20)	0,0415	0,1422
SP (50)	0,0478	0,0854

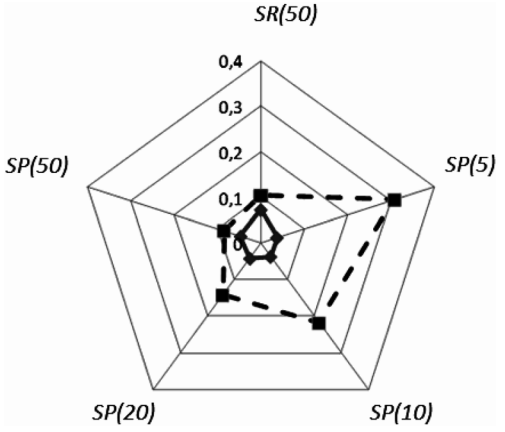


Рис. 6. Критерии эффективности (—— *dist*_1, ... *dist*_2)

3. Вычисление среднего расстояния между ключевыми точками изображений *dist*_1 и моды расстояния между ключевыми точками *dist*_2.

Тестовое множество изображений содержало 2233 цветных изображений, сжатых методом JPEG. Каждое изображение относилось к одной из 49 категорий. Выбор коллекции с разным числом изображений по категориям позволяет приблизиться к реальным базам изображений, составы которых имеют в целом случайный характер. Изображение тестовой коллекции имеет обозначение, состоящее из двух частей — номер категории и номер изображения в категории. Используя данные обозначения, можно автоматически определить принадлежность изображения категории запроса.

Обозначим C — множество изображений, по которому ведется поиск; A — множество найденных изображений; R — множество истинно релевантных изображений (рис. 5).

Критерием оценки точности поиска является доля истинно релевантных изображений в общем числе найденных:

$$SP = \frac{|R \cap A|}{|A|}.$$

Точность на уровне N изображений определяется как число релевантных изображений среди первых N выданных изображений, деленное на N. Обозначим ее $SP(N)$.

Критерием полноты поиска является доля обнаруженных истинно релевантных изображений:

$$SR = \frac{|R \cap A|}{|R|}.$$

В работе полнота рассчитывалась на уровне первых 50 найденных изображений, обозначим ее $SR(50)$.

Значения критериев, вычисленных для двух описанных вариантов поиска по ключевым точкам, представлены в табл. 7 и на рис. 6.

Из табл. 7 видно, что значения критериев для поиска *dist*_2 (по среднему расстоянию между ключевыми точками изображений) оказались выше, чем соответствующие значения для поиска *dist*_1 (по моде расстояний между ключевыми точками изображений):

- $SR(50)$ — в 1,43 раза;
- $SP(5)$ — в 8,07 раз;
- $SP(10)$ — в 5,89 раза;
- $SP(20)$ — в 3,43 раза;
- $SP(50)$ — в 1,79 раза.

Заключение

Таким образом, результаты проведенного экспериментального исследования возможности использования методов регистрации ключевых точек изображения в решении задачи поиска изображе-

ний с визуально похожими объектами, позволяют сделать следующие выводы.

1. Наилучшие результаты с точки зрения точности и полноты поиска на естественных изображениях показали детекторы *Laplace* (*Affintpoints*), *Hessian* (*Affintpoints*), *Ridge* (*Affintpoints*), *Harris* (*Affintpoints*) (наборы данных № 1 и № 3).

2. Среди всех детекторов, показавших наилучшие результаты поиска на естественных изображениях, следует выделить *Hessian*-детектор, поскольку положение ключевых точек, найденных с его помощью, в большей степени соответствует положению объекта на изображении, а показатели полноты и точности в наименьшей степени зависят от числа найденных ключевых точек.

В заключение отметим, что сравнительный анализ результатов использования метода ключевых точек в задаче поиска изображений с другими известными методами решения данной задачи, в том числе, методом поиска на основе свойств протообъекта, разработанном авторами работы [10], является предметом отдельной статьи.

Список литературы

1. **Пименов В. Ю.** Вычислительно-эффективный метод поиска нечетких дубликатов в коллекции изображений // Труды РОМИП 2009. СПб.: НУ ЦСИ, 2009. 198 с.
2. **Moravec H. P.** Towards Automatic Visual Obstacle Avoidance // Proc. 5th International Joint Conference on Artificial Intelligence. 1977. P. 584.
3. **Moravec H. P.** Visual Mapping by a Robot Rover // Proc. International Joint Conference on Artificial Intelligence. 1979. P. 598—600.
4. **Harris C., Stephens M.** A Combined Corner and Edge Detector // Proc. Alvey Vision Conf. Univ. Manchester. 1988. P. 147—151.
5. **Harris C.** Geometry from visual motion // Active Vision. MTT Press, 1992. P. 212—216.
6. **Schmid C., Mikolajczyk M. K.** Scale and affine invariant interest point detectors // International Journal of Computer Vision. 2004. Vol. 60. P. 63—86.
7. **Low D.** Distinctive image features from scale-invariant keypoints // International Journal on Computer Vision. 2004. Vol. 60, N 2. P. 91—110.
8. **Linderberg T.** Feature detection with automatic scale detection // International Journal of Computer Vision. 1998. V. 30, N 2. P. 79—116.
9. **Ke Y., Sukthankar R.** PCA-SIFT: A more distinctive representation for local image descriptors // Computer Vision and Pattern Recognition. 2004. V. 2. P. 506—513.
10. **Левашкина А. О.** Поиск изображений с учетом присутствующих на них объектов // Труды Третьей российской конференции молодых ученых по информационному поиску. Петрозаводск: Изд-во ПетрГУ, 2009. С. 76.

УДК 004.9

О. П. Архипов, канд. техн. наук, директор,
З. П. Зыкова, канд. физ.-мат. наук, зав. лаб.,
Орловский филиал
Института проблем информатики РАН
e-mail: ofran@orel.ru

Многокритериальный выбор тестового множества при исследовании цветовосприятия

Вывод на периферийные устройства ПЭВМ, основанный на оцифровке и согласовании цветовых пространств и адекватно воспринимаемый большинством пользователей, цветовосприятие которых близко к стандартному, для значительной части наблюдателей, имеющих аномалии цветного зрения, не является корректным. Некоторые цвета, различаемые стандартным наблюдателем, воспринимаются ими как одинаковые. Рассматривается задача выбора тестового множества при исследовании цветовосприятия и оцифровке цветового пространства произвольного пользователя ПЭВМ, решение которой необходимо для предотвращения искажения восприятия выводимой на периферийные устройства цветовой информации.

Ключевые слова: *цветная периферия, стандартное цветовосприятие, аномалии цветного зрения, искажение восприятия цветовой информации*

Введение

Одно из основных средств коммуникации человека и окружающего мира — зрительное восприятие информации, при этом наибольшей информативностью обладает цветная информация. Современные периферийные устройства делают цветную информацию доступной любому пользователю ПЭВМ.

Известно, что видимое наблюдателем цветовое пространство шире цветового пространства монитора, которое, в свою очередь, шире цветового пространства принтера. Технологии вывода на периферийные устройства, основанные на оцифровке и согласовании цветовых пространств, ориентируются на восприятие цвета "средним стандартным колориметрическим наблюдателем" (СН) [1]. Большинству пользователей, цветовые пространства которых близки к цветовому пространству СН, они обеспечивают адекватное восприятие цветной информации, включенной в оборот в инфокоммуникационных средах.

Как известно, значительная часть наблюдателей имеет различные аномалии цветного зрения (АН) и поэтому может воспринимать цветную информацию в искаженном (по сравнению с СН) виде.

В рамках данной работы рассматривается цветовосприятие таких АН, которые как одинаковые воспринимают некоторые цвета, различаемые

СН. Чтобы обеспечить АН качественную коммуникацию в системе пользователь/вычислительная среда на основе зрительного восприятия пользователями цветной графической информации, выводимой периферийными устройствами ПЭВМ (мониторами и принтерами), необходимо создать цифровое описание его цветового пространства.

Цветовое восприятие некоторых групп АН в общем виде описывается следующим стандартным образом [1]:

- дейтеранопия — цветовая слепота, при которой не различаются зеленый и красный цвета с нормальной функцией спектральной световой эффективности;
- протанопия — цветовая слепота, при которой не различаются зеленый и красный цвета с ненормально низкой функцией спектральной световой эффективности на длинноволновом участке спектра;
- тританопия — цветовая слепота, при которой не различаются желтые и синие цвета.

В медицине для исследования цветового зрения и обнаружения указанных аномалий используются так называемые пороговые таблицы Юстовой — отпечатки специальных двуцветных изображений, выполненных на полиграфическом оборудовании. Понятно, что результатов такого исследования достаточно для постановки диагноза, но недостаточно для цифрового описания цветовосприятия указанных групп АН, поскольку неизвестно соотношение цветов пороговых таблиц Юстовой, цветов на мониторе и в отпечатках на принтере.

Существующие программы тестирования, после которого испытуемые наблюдатели могут выяснить, являются ли они типичными АН, опираются на различающиеся между собой цифровые модели.

Чтобы проиллюстрировать это, сравним результаты предсказания цветовосприятия дейтеранопов программами ColorOracle [2] и Vischeck [3] на примере изображения $Image_0$ в соответствии с рис. 1 (см. четвертую сторону обложки), построенного из пикселей множества M:

$$M = \{(r_i, g_i, b_i)\} = \{(j \cdot 17, k \cdot 17, n \cdot 17)\},$$

$$j, k, n = 0, 1, ..., 15,$$
$$i = j + k \cdot 16 + n \cdot 16 \cdot 16 = 0, 1, ..., 4095.$$

В соответствии с используемой моделью цветовосприятия дейтеранопов *ColorOracle* преобразует изображение $Image_0$ в изображение $Image_1$ в соответствии с рис. 2 (см. четвертую сторону обложки), а *Vischeck* — в изображение $Image_2$ в соответствии с рис. 3 (см. четвертую сторону обложки).

Таким образом, предсказания *ColorOracle* и *Vischeck* восприятия дейтеранопом изображения $Image_0$ различны. Изображения $Image_1$ и $Image_2$ различаются не только по цветовому решению, но и по структуре, как видно из рис. 4 и 5 (см. четвертую сторону обложки), где пиксели изображений $Image_1$ и $Image_2$ представлены в декартовой системе координат (r, g, b).

Если в первом случае (рис. 4) практически все пиксели лежат в одной плоскости, то во втором (рис. 5) для описания структуры потребуется более сложная нелинейная поверхность.

В связи с этим актуальной является задача цифрового описания цветового пространства произвольного пользователя (ПП). В работах [4—10] нами рассматривались более простые задачи, для решения которых было необходимо отыскать как можно больше пикселей, отпечатки которых различаются ПП. Не требовалось цифровое описание всего пространства цветовосприятия. Было достаточно рассмотрения отдельных его фрагментов — некоторых зон толерантности (совокупностей пикселей, которые не различаются ПП).

Для описания всего пространства цветовосприятия необходимо доработать предложенные в работах [4—10] методы, в частности, необходимо разработать метод выбора пикселей, одновременно являющихся узлами интерполяции при аппроксимации цветового пространства и составляющих тестовое множество при исследовании цветовосприятия ПП.

Постановка задачи

Рассматриваются цветные *RGB*-изображения, состоящие из произвольных пикселей исходного пространства $C_и$ (координатное *RGB*-пространство): $\boldsymbol{x} = \{x\}$, $x \in C_и$, которым после представления их на мониторе соответствуют пиксели из цветового пространства монитора $C_м$: $\boldsymbol{u} = \{u\}$, $u \in C_м$. После визуализации ПП-представления пикселей на мониторе им соответствуют пиксели из цветового пространства ПП $C_{пп}$: $\boldsymbol{y} = \{y\}$, $y \in C_{пп}$.

После печати пикселей $\{x\}$ им соответствуют отпечатки — пиксели цветового пространства применяемого принтера $C_п$: $\boldsymbol{z} = \{z\}$, $z \in C_п$. После сканирования отпечатки $\{z\}$ пикселей $\{x\}$ отображаются в соответствии с [5, 6] в пиксели из координатного *RGB*-пространства применяемого сканера $C_с$: $\boldsymbol{v} = \{v\}$, $v \in C_с$, а после визуализации ПП отпечатков пикселей в пиксели из цветового пространства ПП $C_{пп}$: $\boldsymbol{w} = \{w\}$, $w \in C_{пп}$.

Рассматриваются отображения цветовых пространств периферийных устройств и ПП, которые можно охарактеризовать с помощью функций цветопередачи f_0 ($C_и \Rightarrow C_м$), f_1 ($C_м \Rightarrow C_{пп}$), f_2 ($C_и \Rightarrow C_п$), f_3 ($C_п \Rightarrow C_с$) и f_4 ($C_п \Rightarrow C_{пп}$):

$$u = f_0(x),\ x \in C_и,\ u \in C_м;\ y = f_1(u),\ u \in C_м,\ y \in C_{пп};$$
$$z = f_2(x),\ x \in C_и,\ z \in C_п;\ v = f_3(z),\ z \in C_п,\ v \in C_с;$$
$$w = f_4(z),\ x \in C_п,\ w \in C_{пп}.$$

Рассмотрим также функцию цветопередачи из $C_и$ в $C_м$, а затем из $C_м$ в $C_{пп}$ (обозначим ее F_0):

$$F_0 = f_1 f_0,\ y = F_0(x) = f_1(u) = f_1(f_0(x)),$$
$$x \in C_и,\ y \in C_{пп},\ u \in C_м.$$

Поскольку исходным пространством $C_и$ является *RGB*-пространство, то составляющие его пиксели, а также пиксели исходных изображений характеризуются *RGB*-координатами: $x = (r, g, b)$, $0 \leqslant r, g, b \leqslant 255$. Также пикселям из $C_с$ — значениям цветовой характеристики отпечатков — соответствуют некоторые *RGB*-координаты: $v = (r', g', b')$, $0 \leqslant r', g', b' \leqslant 255$.

Пусть отпечаток пикселя (r_i, g_i, b_i) имеет значение цветовой характеристики (r_i', g_i', b_i'), а пикселя (r_j, g_j, b_j) — (r_j', g_j', b_j'). Выполнение равенства $(r_i, g_i, b_i) = (r_j', g_j', b_j')$ не означает совпадение цветов пикселя (r_i, g_i, b_i) и отпечатка пикселя (r_j, g_j, b_j). Хотя используются координаты одного типа, но цветовые пространства разные. В общем случае одному и тому же цвету в них соответствуют пиксели с разными *RGB*-координатами, поэтому они могут быть согласованы лишь в том смысле, что между пикселями устанавливается связь с помощью функции цветопередачи (обозначим ее F_1) из $C_и$ в $C_с$, являющейся суперпозицией функций f_3 и f_2:

$$F_1 = f_3 f_2,\ v = F_1(x) = f_3(z) = f_3(f_2(x)),$$
$$v = (r', g', b') \in C_с,\ z \in C_п,\ x = (r, g, b) \in C_и.$$

Рассмотрим также функцию цветопередачи из $C_и$ в $C_п$, а затем из $C_п$ в $C_{пп}$ (обозначим ее F_2):

$$F_2 = f_4 f_2,\ w = F_2(x) = f_4(z) = f_4(f_2(x)),$$
$$w \in C_{пп},\ z \in C_п,\ x = (r, g, b) \in C_и.$$

Заметим, что в отличие от функции F_1, имеющей в качестве и аргументов, и значений *RGB*-пиксели, значения функций F_0 и F_2 не являются *RGB*-пикселями.

Пусть протестировано восприятие ПП-представления на мониторе последовательности пикселей $M' = \{x_i\} \subset C_и$. Им соответствуют цвета $\{y_i\} \subset C_{пп}$ восприятия ПП: $y_i = F_0(x_i)$, причем про каждую пару $(x_{i'}, x_{i''})$ пикселей из $\{x_i\}$ после тестирования становится известно, различаются они протестированным ПП $(y_{i'} \neq y_{i''})$ или нет $(y_{i'} = y_{i''})$.

В этом случае одномерную последовательность $\{y_i\}$ можно представить в виде двумерной последовательности:

$$\{Y_{i,j}\},\ i = 1, 2, ..., I,\ j = 1, 2, ..., J_i, \qquad (1)$$

для компонентов которой выполнены соотношения

$$Y_{i,j'} = Y_{i,j''},\ 1 \leqslant i \leqslant I,\ 1 \leqslant j', j'' \leqslant J_i,\ Y_{i',j''} \neq Y_{i''j'''},$$
$$1 \leqslant i', i'' \leqslant I,\ i' \neq i'',\ 1 \leqslant j' \leqslant J_{i'},\ 1 \leqslant j'' \leqslant J_{i'''}. \qquad (2)$$

Компоненты последовательности $\{X_{i,j}\}$, определенной соответствующим образом из пикселей последовательностей $\{x_i\}$, связаны следующими соотношениями:

$$Y_{i,j} = F_0(X_{i,j}),\ 1 \leqslant i \leqslant I,\ 1 \leqslant j \leqslant J_i.$$

Из (1) и (2) следует:

$$F_0(X_{i,j}) = Y_{i,k},\ 1 \leqslant i \leqslant I,\ 1 \leqslant j, k \leqslant J_i,$$

откуда, в частности,

$$F_0(X_{i,j}) = Y_{i,1},\ 1 \leqslant i \leqslant I,\ 1 \leqslant j \leqslant J_i. \qquad (3)$$

Поскольку пикселям $Y_{i,k}$ не могут быть присвоены числовые координаты, то функция F_0 не может быть использована для оцифровки цветового пространства ПП.

Для цифрового описания цветового пространства $C_{пп}$ предлагается использовать функцию F_0' цветопередачи из $C_и$ в $C_и$, обладающую на множестве M' свойствами, аналогичными свойствам (3) функции F_0:

$$F_0'(X_{i,j}, f_1, f_0, M') = X_{i,1},\ 1 \leqslant i \leqslant I,\ 1 \leqslant j \leqslant J_i.$$

За приближенные значения функции F_0' при произвольных значениях аргумента x могут быть приняты значения функции Ψ_1:

$$F_0'(x, f_1, f_0, M') \approx \Psi_1(x, f_1, f_0, M', \omega),$$

вычисленные при интерполировании (обозначим интерполяционную формулу ω) по узлам $X_{i,j}$, в которых значения функций F_0' и Ψ_1 совпадают:

$$\Psi_1(X_{i,j}, f_1, f_0, M', \omega) = F_0'(X_{i,j}, f_1, f_0, M') = X_{i,1},$$
$$1 \leqslant i \leqslant I,\ 1 \leqslant j \leqslant J_i.$$

Пусть далее протестировано восприятие ПП-последовательности отпечатков $\{z_i\} \subset C_п$ пикселей из $M'' = \{x_i\} \subset C_и : z_i = f_2(x_i)$, которым соответствует последовательность значений цветовой характеристики $\{v_i\} \subset C_с$: $v_i = f_3(z_i)$, а также цвета $\{w_i\} \subset C_{пп}$ восприятия ПП: $w_i = f_4(z_i)$.

Если про каждую пару $(z_{i'}, z_{i''})$ отпечатков из $\{z_i\}$ известно, различаются они протестированным ПП $(w_{i'} \neq w_{i''})$ или нет $(w_{i'} = w_{i''})$, то одномерную последовательность $\{w_i\}$ можно представить в виде двумерной последовательности

$$\{W_{i,j}\},\ i = 1, 2, ..., I,\ j = 1, 2, ..., J_i, \qquad (4)$$

для компонентов которой выполнены соотношения

$$W_{i,j'} = W_{i,j''},\ 1 \leqslant i \leqslant I,\ 1 \leqslant j', j'' \leqslant J_i,$$
$$W_{i',j'} \neq W_{i'',j''},\ 1 \leqslant i', i'' \leqslant I,\ i' \neq i'',$$
$$1 \leqslant j' \leqslant J_{i'},\ 1 \leqslant j'' \leqslant J_{i''}. \quad (5)$$

Компоненты последовательностей $\{X_{i,j}\}$, $\{Z_{i,j}\}$, $\{V_{i,j}\}$, определенных соответствующим образом из пикселей последовательностей $\{x_i\}$, $\{z_i\}$, $\{v_i\}$, связаны следующими соотношениями:

$$Z_{i,j} = f_2(X_{i,j}),\ V_{i,j} = f_3(Z_{i,j}),\ W_{i,j} = f_4(Z_{i,j}),$$
$$1 \leqslant i \leqslant I,\ 1 \leqslant j \leqslant J_i.$$

Из (4) и (5) имеем:

$$f_4(Z_{i,j}) = W_{i,k},\ 1 \leqslant j \leqslant I,\ 1 \leqslant j,\ k \leqslant J_i,$$

и, в частности,

$$f_4(Z_{i,j}) = W_{i,1},\ 1 \leqslant i \leqslant I,\ 1 \leqslant j \leqslant J_i.$$

Следовательно, определены некоторые значения функции F_2:

$$F_2(X_{i,j}) = f_4(f_2(X_{i,j})) = f_4(Z_{i,j}) = W_{i,1},$$
$$1 \leqslant i \leqslant I,\ 1 \leqslant j \leqslant J_i. \quad (6)$$

Для цифрового описания цветового пространства восприятия $C_{\text{пп}}$ используется функция F_2' цветопередачи из $C_{\text{и}}$ в $C_{\text{с}}$, обладающая на множестве M'' свойствами, аналогичными свойствам (6) функции F_2:

$$F_2'(X_{i,j}, f_4, f_3, f_2, M'') = V_{i,1},\ 1 \leqslant i \leqslant I,\ 1 \leqslant j \leqslant J_i.$$

За приближенные значения функции F_2' при произвольных значениях аргумента могут быть приняты значения функции Ψ_2:

$$F_2'(x, f_4, f_3, f_2, M'') \approx \Psi_2(x, f_4, f_3, f_2, M'', \omega),$$

вычисленные при интерполировании по узлам $X_{i,j}$, в которых значения функций F_2' и Ψ_2 совпадают:

$$\Psi_2(X_{i,j}, f_4, f_3, f_2, M'', \omega) = F_2'(X_{i,j}, f_4, f_3, f_2, M'') = V_{i,1},$$
$$1 \leqslant i \leqslant I,\ 1 \leqslant j \leqslant J_i.$$

Пусть с помощью применения какой-либо интерполяционной формулы ω для всех пикселей $x = (R, G, B)$ множества M при применении тестового множества M' определены значения функции Ψ_1:

$$(R', G', B') = \Psi_1((R, G, B), f_1, f_0, M', \omega),$$

а при M'' — значения функции Ψ_2:

$$(R'', G'', B'') = \Psi_2((R, G, B), f_4, f_3, f_2, M'', \omega).$$

Обозначим как (r', g', b') и (r'', g'', b'') точные значения функции F_0':

$$(r', g', b') = F_0'((R, G, B), f_1, f_0, M')$$

и соответственно F_2':

$$(r'', g'', b'') = F_2'((R, G, B), f_4, f_3, f_2, M'').$$

Если точные значения известны, то погрешности интерполяции выражаются следующим образом:

$$\varepsilon(f_1, f_0, M', \omega) =$$
$$= \max_{x \in M} \rho(\Psi_1(x, f_1, f_0, M', \omega),\ F_0'(x, f_1, f_0, M')),$$

где $\rho(\Psi_1(x, f_1, f_0, M', \omega),\ F_0'(x, f_1, f_0, M')) = \max(|R' - r'|, |G' - g'|, |B' - b'|)$,
а также

$$\varepsilon(f_4, f_3, f_2, M'', \omega) =$$
$$= \max_{x \in M} \rho(\Psi_2(x, f_4, f_3, f_2, M'', \omega),\ F_2'(x, f_4, f_3, f_2, M'')),$$

где $\rho(\Psi_2(x, f_4, f_3, f_2, M'', \omega),\ F_2'(x, f_4, f_3, f_2, M'')) = \max(|R'' - r''|, |G'' - g''|, |B'' - b''|)$.

Обозначим ε_0 — допустимое значение погрешности. Заметим, что максимальное значение, которое можно принять в качестве ε_0, в соответствии с [11] равно шестнадцати: $\varepsilon_0 \leqslant 16$.

Задача. Выбрать интерполяционную формулу ω и тестовые множества M' и M'' такие, чтобы

- погрешности $\varepsilon_1(f_1, f_0, M', \omega)$ и $\varepsilon_2(f_4, f_3, f_2, M'', \omega)$ при интерполяции функций цветопередачи из [2, 3] были допустимыми, т. е. были выполнены неравенства

$$\varepsilon_1(f_1, f_0, M', \omega) \leqslant \varepsilon_0,\ \varepsilon_2(f_4, f_3, f_2, M'', \omega) \leqslant \varepsilon_0; \quad (7)$$

- тестовые множества M' и M'' включали в себя вершины *RGB*-куба;
- тестовое множество M'' включало в себя *RGB*-пиксели, которые при цветопередаче в пользовательской компьютерной системе являются прообразами реперных цветов пороговых таблиц Юстовой.

Замечание. Решение поставленной задачи позволяет, ориентируясь на функции цветопередачи моделей [2, 3], на цифровой анализ пороговых таблиц Юстовой, априори получить приближенное представление о погрешностях интерполяции в общем случае.

Алгоритм решения задачи

Рассматривая примеры моделей из [2, 3], будем использовать следующие обозначения для соответствующих им функций цветопередачи между *RGB*-пространствами: φ_1 — для дейтеранопов [2]; φ_2 — для протанопов [2]; φ_3 — для тританопов [2]; φ_4 — для дейтеранопов [3]; φ_5 — для протанопов [3]; φ_6 — для тританопов [3].

Чтобы вершины *RGB*-куба входили в состав тестовых множеств, достаточно выбрать одно из следующих небольших по размеру множеств *RGB*-пикселей, равномерно распределенных в *RGB*-кубе:

$$T_1 = \{(r_i, g_i, b_i)\} = \{j \cdot 255, k \cdot 255, n \cdot 255)\},$$
$$j, k, n = 0, 1, i = j + k \cdot 2 + n \cdot 2 \cdot 2 = 0, 1, ..., 7;$$

$$T_2 = \{(r_i, g_i, b_i)\} = \{j \cdot 85, k \cdot 85, n \cdot 85)\},$$
$$j, k, n = 0, 1, ..., 3, i = j + k \cdot 4 + n \cdot 4 \cdot 4 = 0, 1, ..., 65;$$

$$T_3 = \{(r_i, g_i, b_i)\} = \{j \cdot 51, k \cdot 57, n \cdot 51)\},$$
$$j, k, n = 0, 1, ..., 5, i = j + k \cdot 6 + n \cdot 6 \cdot 6 = 0, 1, ..., 215.$$

Чтобы множество M'' удовлетворяло условиям задачи, необходимо к выбранному множеству также добавить *RGB*-прообразы $T_4 = \{x_{i,j}\}$, $i = 0, 1, ..., 11$, $j = 0, 1$, реперных пикселей пороговых таблиц Юстовой.

Для вычисления T_4 необходимо проделать следующую работу:

- создать и напечатать изображение из пикселей $x \in M$;
- отсканировать полученный отпечаток, обработать полученный скан и вычислить значения цветовой характеристики $\{v\}$ отпечатков $\{z\}$ пикселей $\{x\}$: $v = F_1(x)$;
- отсканировать пороговые таблицы, обработать полученный скан и вычислить значения $\{v'_{i,j}\}$, $i = 0, 1, ..., 11, j = 0, 1$, цветовой характеристики отпечатков — реперные пиксели пороговых таблиц;
- для каждого реперного пикселя $\{v'_{i,j}\}$, $i = 0, 1, ..., 11, j = 0, 1$, среди пикселей $\{v\}$, являющихся значениями цветовой характеристики отпечатков пикселей из множества M, найти ближайший (обозначим его $v''_{i,j}$), т. е. такой, что

$$\rho(v'_{i,j}, v''_{i,j}) = \min_{\{v\}} \rho(v'_{i,j}, v);$$

- в качестве прообраза реперного пикселя $v'_{i,j}$ необходимо выбрать пиксель $x'_{i,j}$ такой, что $v''_{i,j} = F_1(x'_{i,j})$.

Пусть множества M' и M'' определены одним из указанных ранее способов. Значения функций цветопередачи $\{\varphi_k\}$, $k = 1, 2, ..., 6$, заданные неявно, могут быть определены как координаты соответствующих пикселей изображений, модифицированных с помощью ColorOracle или Vischeck. В связи с этим,для вычисления погрешности $\varepsilon_1(f_1(\varphi_k), f_0, M', \omega)$ потребуется:

- создать изображения Img_1 из пикселей $x \in M$;
- с помощью ColorOracle или Vischeck создать из Img_1 изображение $Img_{2,k}$, при этом пиксели k-го изображения равны значениям функции φ_k от аргументов, которые равны соответствующим пикселям из Img_1:
- определить значения $\Psi_1(x, f_1(\varphi_k), f_0, M', \omega) = \varphi_k(x)$, $x \in M'$, $1 \leqslant k \leqslant 6$, отыскивая пиксель $x \in M'$ в Img_1 и в качестве $\Psi_1(x, f_1(\varphi_k), f_0, M', \omega)$ выбирая пиксель с соответствующим местоположением из $Img_{2,k}$;
- создать из Img_1 изображения $Img_{3,k}$ путем замены пикселей $(R, G, B) \in M$ значениями (R'_k, G'_k, B'_k), вычисленными по интерполяционной формуле ω по узлам $x \in M'$ и значениям в них $\varphi_k(x)$;
- вычислить погрешность $\varepsilon_1(f_1(\varphi_k), f_0, M', \omega)$ как максимум из расстояний между соответствующими пикселями изображений $Img_{3,k}$ и $Img_{2,k}$.

Для вычисления погрешности $\varepsilon_2(f_4(\varphi_k), f_3, f_2, M'', \omega)$ потребуется:

- создать и напечатать изображение Img_1 из пикселей $x \in M$;
- отсканировать полученный отпечаток, обработать полученный скан и вычислить цветовые характеристики $\{v\}$ отпечатков $\{y\}$ пикселей $\{x\}$: $v = F_1(x)$;
- создать из Img_1 изображение Img_4 путем замены пикселей x соответствующими пикселями v;
- с помощью ColorOracle или Vischeck создать из Img_4 изображение $Img_{5,k}$, при этом координаты пикселей k-го изображения равны значениям функции φ_k от аргументов, которые равны соответствующим пикселям из Img_4;
- определить значения $\Psi_2(x, f_4(\varphi_k), f_3, f_2, M'', \omega) = \varphi_k(v)$, $x \in M''$, $v = F_1(x)$, $1 \leqslant k \leqslant 6$, отыскивая пиксель $x \in M''$ в Img_1 и в качестве $\varphi_k(v)$ выбирая пиксель с соответствующим местоположением из $Img_{5,k}$;
- создать из Img_1 изображения $Img_{6,k}$ путем замены пикселей $(R, G, B) \in M$ значениями (R'_k, G'_k, B'_k), вычисленными по интерполяционной формуле ω по узлам $x \in M''$ и значениям в них $\varphi_k(F_1(x))$;
- вычислить погрешность $\varepsilon_2(f_4(\varphi_k), f_3, f_2, M'', \omega)$ как максимум из расстояний между соответствующими пикселями изображений $Img_{5,k}$ и $Img_{6,k}$.

Цветовосприятие дейтеранопов, протанопов и тританопов можно считать наибольшими отклонениями от цветовосприятия СН. В связи с этим можно надеяться, что погрешности $\varepsilon_1(f_1(\varphi_k), f_0, M', \omega)$ и $\varepsilon_2(f_4(\varphi_k), f_3, f_2, M'', \omega)$, характеризующие аппроксимацию функций цветопередачи в рассмотренных случаях, будут того же порядка, что и погрешность аппроксимации произвольной функции цветопередачи, занимающей некоторое промежуточное положение между указанными случаями.

Если значения погрешностей $\varepsilon_1(f_1(\varphi_k), f_0, M', \omega)$ и $\varepsilon_2(f_4(\varphi_k), f_3, f_2, M'', \omega)$ допустимы, то задача решена. Иначе необходимо попытаться изменить

входные параметры (M', M'' и ω), чтобы добиться выполнения условия (7).

Результаты применения другой интерполяционной формулы априори не очевидны, тем более что может быть использован итерационный метод расширения M', M'' путем добавления новых пикселей из M, гарантирующий на конечной итерации выполнение (7) при любой применяемой итерационной формуле. Действительно, в вырожденном предельном случае, когда $M' = M$, $M'' = M$, погрешность равна нулю, (7) очевидно выполнено. В связи с этим искать более подходящую интерполяционную формулу следует только, если число пикселей в M' или M'' слишком велико.

Рассмотрим один из алгоритмов итерационного расширения тестовых множеств на примере множества M'. Будем обозначать тестовое множество на i-й итерации как M'_i. На нулевой итерации положим $M'_0 = M'$.

Пусть выполнена i-я итерация и определено тестовое множество M'_i, на котором $\varepsilon_1(f_1(\varphi_k), f_0, M'_i, \omega) > 16$, тогда существует хотя бы один пиксель $x' \in M$, для которого выполнены неравенства

$$\rho(\Psi_1(x', f_1(\varphi_k), f_0, M'_i, \omega), F'_0(x', f_1(\varphi_k), f_0, M'_i)) > 16;$$

$$\rho(\Psi_1(x', f_1(\varphi_k), f_0, M'_i, \omega), F'_0(x', f_1(\varphi_k), f_0, M'_i)) \geqslant$$
$$\geqslant \rho(\Psi_1(x, f_1(\varphi_k), f_0, M'_i, \omega), F'_0(x, f_1(\varphi_k), f_0, M'_i)),$$
$$x \in M.$$

После определения множества M'_{i+1} как объединения множества M'_i и пикселя x': $M'_{i+1} = M'_i \cup x'$, для него вычисляется значение погрешности. Если $\varepsilon_1(f_1(\varphi_k), f_0, M'_{i+1}, \omega) > 16$, то итерационная процедура продолжается до тех пор, пока погрешность не станет допустимой.

Аналогичная процедура применима и для уменьшения погрешности $\varepsilon_2(f_4(\varphi_k), f_3, f_2, M'', \omega)$. По завершении процедур необходимо оценить число пикселей в тестовом множестве. Если число пикселей не является чрезмерно обременительным для проведения тестирования цветовосприятия ПП пикселей и их отпечатков, то начальные значения параметров для проведения тестирования можно считать найденными, а задачу решенной. Иначе следует пересмотреть подход к аппроксимации и попробовать применить другую интерполяционную формулу.

Пример реализации алгоритма решения задачи

Для реализации алгоритма было создано специальное программное обеспечение, функционирующее на базе одного компьютера типа PC IBM с оболочкой Windows XP. Использовалась периферия: цветной лазерный принтер HP Color LaserJet 4700n и цветной сканер FUJITSU fi-60F.

Интерполяция проводилась в несколько этапов:

- для пикселей M, принадлежащих ребрам *RGB*-куба;
- для пикселей M, принадлежащих поверхности *RGB*-куба;
- для пикселей M, являющихся внутренними точками *RGB*-куба.

Результаты выполнения каждого промежуточного этапа использовались в качестве начальных данных следующего этапа.

Заметим, что при аппроксимации на ребрах *RGB*-куба меняется только одна из координат (пусть, для примера, r). Значение функции $\Psi_1((r, g, b), f_1(\varphi_k), f_0, M', \omega)$ во внутренней точке (r_0, g, b) отрезка $[(r_1, g, b), (r_2, g, b)]$, $r_1 < r_0 < r_2$, по известным значениям функции на концах отрезка вычислялось при линейной интерполяции

$$\Psi_1((r_0, g, b), f_1(\varphi_k), f_0, M', \omega) \approx \frac{(r_2 - r_0)\Psi_1((r_1, g, b), f_1(\varphi_k), f_0, M', \omega) + (r_0 - r_1)\Psi_1((r_2, g, b), f_1(\varphi_k), f_0, M', \omega)}{(r_2 - r_1)}.$$

При аппроксимации на поверхности и во внутренних точках *RGB*-куба использовались обобщения приведенной формулы на двумерный и соответственно на трехмерный случаи. При реализации итерационной процедуры тестовое множество доопределялось постепенно при рассмотрении функций цветопередачи $\{\varphi_k\}$, $k = 1, 2, ..., 6$.

Результаты проведения экспериментального исследования с помощью созданного программного обеспечения показали, что размеры допустимого тестового множества зависят от порядка, в котором рассматривались функции цветопередачи, а также от размеров и структуры начального тестового множества.

Число пикселей тестовых множеств

№ дополнения	Начальное тестовое множество		
	T_1	T_2	T_3
1	27	76	227
2	31	81	232
3	76	121	262
4	165	208	341
5	226	270	392
6	281	323	436
7	282	324	443
8	284	326	444
9	285	327	445

Например, при последовательном переборе функций цветопередачи $\{\varphi_k\}$, $k = 1, 2, ..., 6$, чтобы найти допустимое тестовое множество, начиная с одного из множеств T_1, T_2, T_3, понадобилось 18 шагов, причем на девяти из них тестовые множества дополнялись новыми пикселями. Число пикселей при каждом из дополнений приведено в таблице.

Как видно из приведенной таблицы, наилучшие результаты получены при выборе в качестве начального приближения множества T_1. Найденное при этом допустимое тестовое множество не потребует чрезмерных затрат ресурсов при тестировании.

Заключение

Рассмотрена задача вычисления тестового множества, допустимого при исследовании цветовосприятия вывода на периферийные устройства и оцифровке цветового пространства произвольного пользователя ПЭВМ.

Предложен метод ее решения, адаптированный к пользовательским условиям и позволяющий на основе применения специального программного обеспечения вычислять тестовые множества, удовлетворяющие определенным критериям и допустимые при проведении тестирования цветовосприятия ПП.

Результаты работы имеют важное практическое значение, поскольку применимы при решении задач управления цветопередачей в компьютерных системах с цветной периферией не только в интересах пользователей, чье цветовосприятие близко к цветовосприятию СН, но и в интересах пользователей, имеющих аномалии цветного зрения.

Список литературы

1. **Джадд Д., Вышецки Г.** Цвет в науке и технике. М.: Мир, 1978. 592 с.
2. **Color** Oracle / Institute of Cartography, ETH Zurich, 2008. URL: http://colororacle.cartography.ch
3. **Vischeck**. — Электрон. граф. дан. и прогр. URL: http://www.vischeck.com
4. **Архипов О. П., Архипов П. О., Зыкова З. П.** Цветной штриховой код на лазерных принтерах // Информационные технологии. 2007. № 2. С. 11—15.
5. **Архипов О. П.** Неустранимые погрешности цветовоспроизведения на лазерных принтерах // Информационные технологии. 2007. № 4. С. 24—27.
6. **Архипов О. П., Архипов П. О., Захаров В. Н., Зыкова З. П.** Аппроксимация цветных изображений для получения машиночитаемых объектов // Информационные технологии. 2007. № 7. С. 53—57.
7. **Архипов О. П., Зыкова З. П.** Допечатное тестирование индивидуального зрительного восприятия // Вестник компьютерных и информационных технологий. 2008. № 12. С. 2—8.
8. **Архипов О. П., Захаров В. Н., Зыкова З. П.** Тестирование подсистем цветного вывода в распределенных системах // Информационные технологии и вычислительные системы. 2008. № 3. С. 78—84.
9. **Архипов О. П., Бородина Л. Н., Зыков Р. В.** и др. Технология оцифровки цветовосприятия отпечатков. М.: ИПИ РАН, 2009. 115 с.
10. **Архипов О. П., Зыкова З. П.** Обеспечение идентичности результатов печати на различных цветных принтерах // Информационные технологии. 2009. № 3. С. 37—42.
11. **Городецкий В. И., Самойлов В. В.** Стеганография на основе цифровых изображений // Информационные технологии и вычислительные системы. 2001. № 2/3. С. 51—64.

УДК 004.89

М. А. Масюк, аспирант,
Сибирский государственный
технологический университет, г. Красноярск,
e-mail: masyuk@legis.krsn.ru

Система анализа и визуализации связей нормативно-правовых документов

Рассматривается сложившаяся в Российской Федерации ситуация, связанная со стремительным ростом числа принимаемых документов законотворческого характера. Значительная часть принимаемых законов носит поправочный характер, т. е. содержит в себе ссылки на другие акты. Анализ множества документов с их взаимосвязями является сложным рутинным занятием, требующим наличия высококвалифицированных специалистов. Предлагается комплексный подход к усовершенствованию справочно-правовых систем и электронных баз данных путем интеграции в них системы, реализующей визуальное отображение взаимосвязей документов и анализ взаимосвязей на предмет их соответствия нормам законотворчества. Приводится пример графического построения "окрестности" одного из реальных законотворческих документов.

Ключевые слова: *нормативно-правовой документ, анализ, визуализация*

В последние годы в Российской Федерации и ее субъектах наблюдается стремительный рост законотворческой деятельности, который отнюдь не свидетельствует о качестве правого регулирования [1]. С развитием законодательной базы существенно возросло число производных нормативно-правовых актов — законов, постановлений, указов, значительная часть которых носит поправочный характер, т. е. содержит в себе ссылки на другие документы с описанием вносимых поправок в текст или отменой ранее действующих документов. Такие ссылки одних документов на другие можно представить в виде единой связанной структуры — ориентированного графа, который можно рассматривать на множестве документов какой-либо электронной базы данных или справочно-правовой системы, в рамках законодательства Российской Федерации или отдельного субъекта. Представление общей картины путем анализа текстов является трудоемкой процедурой. Кроме того, существует вероятность ошибки при разработке нового законодательного акта, если не учесть все связанные с ним (ранее принятые) документы, которые, в свою очередь, имеют аналогичные взаимосвязи с другими документами.

Автором предлагается комплексный подход к усовершенствованию справочно-правовых систем и электронных баз данных путем интеграции в них системы, реализующей визуальное отображение взаимосвязей документов и их анализ на предмет соответствия нормам законотворчества, в целях повышения эффективности работы экспертов и юристов и, как следствие, повышения качества принимаемых законов.

Математическая модель взаимосвязей документов

Для формализации предметной области предлагается следующая математическая модель взаимосвязей документов.

Задан массив документов (база данных документов) $S = \{s_1, s_2, ..., s_N\}$, где s_i — i-й документ массива, $i = 1, ..., N$, N — общее число документов в массиве.

Структура документа s_i представляет собой упорядоченный набор атрибутов $\langle a_{i1}, a_{i2}, ..., a_{ik_i}, Ri\rangle$, где a_{ik} — k-й информационный атрибут i-го документа (название, номер, дата принятия), а Ri — специальный атрибут для связи с другими документами из S.

Документы множества S упорядочены по дате принятия — одному из своих атрибутов.

Между документами S существует система связей $L = \{l_{ij}, i, j = 1, ..., N, i \neq j\}$, где l_{ij} — связь документа s_i с документом s_j.

Связи документов l_{ij} принимают значения из множества Λ типов связей $l_{ij} \in \Lambda = \{\lambda_0, \lambda_1, \lambda_2, \lambda_3\}$ по следующим правилам:

- $l_{ij} = \lambda_0$ тогда и только тогда, когда документ s_i не имеет в своем тексте ссылки на документ s_j, т. е. λ_0 — пустая связь;
- $l_{ij} = \lambda_1$ тогда и только тогда, когда в тексте документа s_i содержатся указания о внесении изменений в документ s_j, т. е. λ_1 — изменяющаяся связь;
- $l_{ij} = \lambda_2$ тогда и только тогда, когда в тексте документа s_i содержатся указания о признании документа s_j утратившим силу, т. е. λ_2 — отменяющая связь;

- $l_{ij} = \lambda_3$ тогда и только тогда, когда документ s_i имеет в своем тексте упоминание о документе s_j, но семантическое значение этого упоминания не соответствует ни λ_1, ни λ_2, т. е. λ_3 — связь произвольного типа.

Определение окрестности документа

Окрестностью первого уровня K_i^1 документа s_i называется множество всех документов, имеющих ссылку на документ s_i (стоковая часть окрестности), в объединении с множеством документов, на которые имеется ссылка в документе s_i (истоковая часть окрестности):

$$K_i^1 = K_i^{1+} \cup K_i^{1-},$$

где $K_i^{1+} = K^{1+}(s_i) = \bigcup_{j=1}^{N} s_j | l_{ji} \neq \lambda_0$; $K_i^{1-} = K^{1-}(s_i) = \bigcup_{j=1}^{N} s_j | l_{ij} \neq \lambda_0$; $i = 1, ..., N$.

Окрестностью второго уровня K_i^2 документа s_i называется множество

$$K_i^2 = K_i^1 \cup \bigcup_{j=1}^{N} K_j^1 | s_j \in K_i^1 = K_i^1 \cup$$

$$\cup \bigcup_{j=1}^{N} \left(\bigcup_{k=1}^{N} s_k | \lambda_{kj} \neq \lambda_0 \right) | s_j \in K_i^1 \cup \bigcup_{j=1}^{N} \left(\bigcup_{k=1}^{N} s_k | \lambda_{jk} \neq \lambda_0 \right) | s_j \in$$

$$\in K_i^1 = K_i^1 \cup K_i^{2+} \cup K_i^{2-}, \; i = 1, ..., N.$$

Окрестностью n-го уровня по аналогии называется множество

$$K_i^n = K_i^1 \cup K_i^2 \cup ... \cup K_i^{n-1} \cup \bigcup_{j=1}^{N} K_j^1 | s_j \in K_i^{n-1} =$$

$$= K_i^{1+} \cup K_i^{1-} \cup K_i^{2+} \cup K_i^{2-} \cup ... \cup K_i^{n+} \cup K_i^{n-},$$

$$i = 1, ..., N.$$

Практическое применение

В процессе ведения законотворческой деятельности при необходимости внесения поправок в действующие законодательные акты и в процессе написания новых существует риск возникновения нарушений формальных правил и норм законодательного процесса, закрепленных в виде специальных документов [2], [3]. В указанных методических рекомендациях приведены требования к оформлению законопроектов и к законодательной технике по внесению поправок и отмене действующих законов с учетом предыдущих изменений и связанных с ними документов.

Предлагаемые автором методика и реализующая ее система предназначены для усовершенствования произвольных электронных баз данных нормативно-правовых документов или так называемых, справочных правовых систем. Применяемые в системе наглядное графическое построение "окрестности" документа и автоматический анализ связей нормативно-правовых документов с цветовым указанием документов и связей, потенциально-опасных с точки зрения несоблюдения норм законотворчества, способствуют повышению эффективности поиска и разрешению проблем такого рода, а также повышают качество инвентаризации и мониторинга законодательства.

В Законодательном собрании Красноярского края в режиме опытной эксплуатации функционирует прототип системы анализа и визуализации связей нормативно-правовых документов, интегрированный в автоматизированную систему обеспечения законодательной деятельности (АСОЗД) и содержащий в себе следующие подсистемы:

1) подсистема расстановки гиперссылок [4] документов;

2) подсистема автоматического анализа связанной структуры на наличие связей, некорректных с точки зрения норм законотворчества;

3) подсистема визуализации "окрестностей" нормативно-правовых документов.

В основе реализации лежит продукт корпорации IBM — Lotus Notes/Domino [5], представляющий собой документно-ориентированную платформу типа клиент—сервер, служащую для разработки, размещения и использования прикладных программ группового обеспечения.

Определение потенциально опасных с точки зрения несоблюдения норм законотворчества связей и документов

Выделим L^1, L^2, ..., L^A — подмножества множества L и $C = \{C_1, C_2, ..., C_A\}$ — систему условий (критериев). Связь l_{ij} является потенциально опасной и принадлежит L^a, $a = 1, ..., A$, тогда и только тогда, когда она удовлетворяет критерию C_a:

$$l_{ij} \in L^a \Leftrightarrow C_a(l_{ij}) = 1,$$

где $L^a \subseteq L$, $C_a \subseteq C$, $a = 1, ..., A$.

Аналогично для документов:

S^1, S^2, ..., S^B — подмножества множества S и $C' = \{C_1', C_2', ..., C_B'\}$ — система условий (критериев). Документ s_i является потенциально опасным и принадлежит S^b, $b = 1, ..., B$, тогда и только тогда, когда он удовлетворяет критерию C_b':

$$s_i \in S^b \Leftrightarrow C_b'(s_i) = 1,$$

где $S^b \subseteq S$, $C_b' \subseteq C'$, $b = 1, .., B$.

Подсистема расстановки гиперссылок документов

В основе подсистемы расстановки гиперссылок лежит лингвистический анализатор, выполняющий автоматическую расстановку гиперссылок одних документов на другие в рамках электронной базы данных. Способ реализации подсистемы гиперссылок может варьироваться в зависимости от особенностей электронной базы данных и клиентского приложения. Можно выделить два принципиально разных подхода в расстановке гиперссылок в документы:

- предварительная обработка документов с добавлением к ним метаданных, содержащих информацию об имеющихся ссылках на другие документы "окрестности";
- потоковая обработка документов по запросу пользователя или других подсистем без редактирования документов.

Первый описанный способ является более предпочтительным, так как не требует излишних затрат времени на повторяющиеся вычисления, но при этом необходима возможность редактирования данных родительской СУБД. Еще одно преимущество первого подхода — возможность улучшения качества взаимодействия системы с пользователем путем организации перехода от документа к документу посредством использования гиперссылки, что может быть реализовано при условии доработки клиентского приложения.

Для организации гиперссылок в системе используется лингвистический анализатор, выполняющий следующие функции:

- корректное распознавание наличия ссылок в тексте документа;
- автоматическое распознавание типа ссылки (отмена документа, внесение поправок в текст и др.);
- анализ наличия документов, на которые выявлена ссылка, в рассматриваемой электронной базе данных;
- добавление в документ метаданных о ссылках (в случае способа предварительной обработки документа).

Подсистема автоматического анализа связанной структуры

Множество нормативно-правовых документов, взаимосвязанных между собой, образуют единую связанную структуру или взвешенный ориентированный граф, ребра которого имеют три возможных значения веса λ_1, λ_2, λ_3. Вершины графа раскрашены в зависимости от типа документа и его функциональной направленности (о внесении изменений, о признании утратившими силу законов).

Назначение подсистемы — автоматический анализ "окрестностей" исследуемого документа в единой связанной структуре и обнаружение потенциально опасных с точки зрения норм законотворчества ситуаций. На данном этапе развития системы практический интерес представляет анализ окрестностей не более чем второго уровня, так как визуальное восприятие более широких окрестностей затруднительно, а используемые в данный момент критерии не оперируют узлами, расположенными от исследуемого на большем расстоянии.

Для решения задачи обнаружения потенциально опасных связей нормативно-правовых документов в подсистеме автоматического анализа применена технология интеллектуальных агентов [6]. По запросу пользователя либо при выполнении автоматического фонового анализа законодательной базы на вход агенту последовательно передаются исследуемые документы. Интеллектуальный агент, используя доступную базу знаний — критериев потенциальной опасности, анализирует окрестность входного документа и принимает решение о наличии или отсутствии возможных противоречий в его входящих и исходящих связях. Результатом работы программного агента служат два массива специальных объектов, каждый из которых может сигнализировать о потенциальной опасности соответствующей ему связи и содержит в себе пояснение, на основании какого критерия связь идентифицирована как опасная. Такие выходные данные можно считать универсальными и пригодными для использования в любых подсистемах в дальнейшем.

Используемый подход включает в себя логический аппарат метаправил, формально описывающих критерии потенциально опасных ситуаций. Система реализована таким образом, что эволюция норм и правил законотворчества влечет добавление или изменение метаправил и, как следствие, расширение возможностей агента, а не изменение его алгоритма работы.

Примерами связей, потенциально опасных с точки зрения противоречия нормам законотворчества, могут служить:

1) ссылки любого из трех типов на документ, который был отменен ранее; соответствующий критерий:

$$C_1(l_{xi}) = \lfloor \exists l_{yi} | l_{yi} = \lambda_2,\ x > y,\ i,\ x,\ y = 1,\ ...,\ N \rfloor;$$

2) ссылки типа "внесение изменений" на документ, от которого исходит ссылка того же типа; в реальной ситуации это означает, что вносятся изменения в закон, который сам вносит изменения в другой закон, что недопустимо ([2], пункт 57); соответствующий критерий:

$$C_2(l_{xi}) = \lfloor l_{xi} = \lambda_2,\ \exists l_{iy} = \lambda_2,\ i,\ x,\ y = 1,\ ...,\ N \rfloor.$$

Практическое значение может иметь как анализ уже существующей законодательной базы субъектов РФ (кодификация законодательства), так и постоянный анализ всех принимаемых законодательных актов.

Подсистема визуализации взаимосвязей нормативно-правовых документов

Подсистема визуализации выполняет функции графического отображения части общей связанной структуры, включающей рассматриваемый и связанные с ним на произвольную глубину документы. Для решения этой задачи используется рекурсивный алгоритм, базовым методом которого является графическое построение следующего дерева:

1) основной рассматриваемый документ отображается в виде узла в центре области изображения;

2) документы, ссылки на которые содержатся в основном рассматриваемом документе, отображаются в виде узлов и дуг на нижнем уровне относительно основного узла — истоковая часть;

3) документы из рассматриваемой предметной области, которые содержат в себе ссылки на основной рассматриваемый документ, отображаются в виде узлов и дуг на верхнем уровне относительно основного узла — стоковая часть.

Применяя описанный выше метод на все вновь создаваемые узлы, клиентское приложение выполняет графическое построение окрестности рассматриваемого документа в виде графа (см. рисунок). Для каждого строящегося узла в режиме реального времени осуществляется проверка связей соответствующего ему документа с использованием описанного выше программного агента. Части графа, удовлетворяющие критериям потенциальной опасности, окрашиваются в красный цвет.

Используемый в подсистеме алгоритм построения дополнительно решает сопутствующие задачи: обработка ситуаций с повторяющимися узлами — построение циклов; расположение и масштабирование структуры на плоскости.

Подсистема графического отображения обладает следующими особенностями, повышающими ее гибкость, наглядность и удобство при использовании:

- подмножество узлов графа, представляющих документы, и подмножество дуг графа, представляющих связи документов, обладающие особыми свойствами, выделяются цветом, отличающимся от основной части узлов и дуг. Такое цветовое выделение позволяет точнее проанализировать специфику юридической ситуации;
- документы визуально расположены в порядке, определяемом временем вступления нормативно-правового документа в силу, для повышения наглядности и информативности;
- интерфейс пользователя содержит фильтры, ограничивающие множество отображаемых документов по их типам; при этом возможно построение графа из документов определенного типа;
- интерфейс пользователя содержит инструменты, позволяющие масштабировать изображение графа на экране компьютера по вертикали и горизонтали для улучшения наглядности и печати;
- интерфейс пользователя содержит вывод всплывающих подсказок, содержащих более подробную информацию о документе, соответствующем узлу, а также о том, по какому именно критерию элемент был идентифицирован агентом как потенциально опасный с точки зрения нарушения норм законотворчества.

Подсистема визуализации взаимосвязей нормативно-правовых документов реализована с использованием VML (*Vector Markup Language*) [7], отображение осуществляется в браузере Microsoft Internet Explorer. Возможна реализация любыми другими подобными средствами.

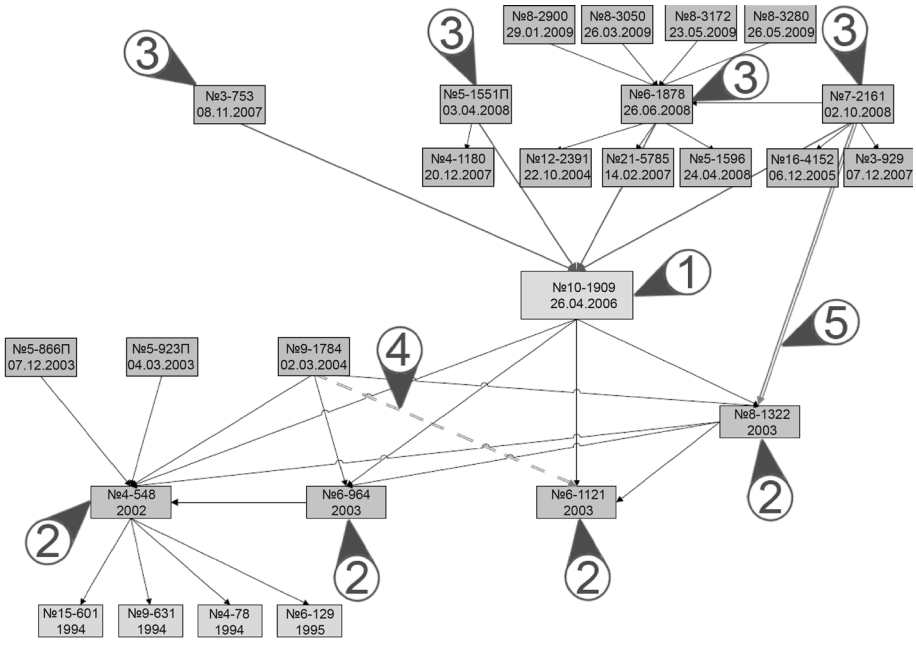
На рисунке представлен пример визуализации "окрестности" нормативно-правового документа, сгенерированный подсистемой, где введены следующие обозначения (ситуация искусственно имитирована):

1 — основной рассматриваемый документ № 10-1909;

2 — документы истоковой части графа;

3 — документы стоковой части графа;

4 — отсутствие связи между документами, которое система распознала как потенциально опасное с точки зрения норм законотворчества;



Пример визуализации окрестности нормативно-правового документа

5 — связь, которую система распознала как потенциально опасную с точки зрения норм законотворчества.

Заключение

В результате проделанной работы автором разработаны методика и реализующая ее система, позволяющие повысить эффективность создания нормативно-правовых актов и анализа существующей законодательной базы со следующими особенностями:

- использование элементов искусственного интеллекта при поиске потенциально опасных с точки зрения норм законотворчества ситуаций;
- графическое представление окрестности нормативно-правового документа, позволяющее пользователю идентифицировать такие ситуации и лучше понять их специфику.

Применение технологий автоматического анализа и графической интерпретации в исследуемой области — взаимосвязанной структуре нормативно-правовых документов — не имеет аналогов в известных нам программных продуктах и открытых публикациях, что подтверждено соответствующими патентными исследованиями [8].

Прототип системы анализа и визуализации связей нормативно-правовых документов вызвал заинтересованность со стороны специалистов и руководителей Законодательного собрания Красноярского края. В настоящее время планируется дальнейшее усовершенствование системы в следующих направлениях:

- наращивание функциональных возможностей и доработка интерфейса пользователя;
- расширение возможностей интеллектуального агента в поиске потенциально опасных ситуаций путем применения онтологии соответствующей предметной области.

Список литературы

1. **Доклад** Совета Федерации Федерального Собрания Российской Федерации 2008 года — Москва, 2008.
2. **Методические** рекомендации по юридико-техническому оформлению законопроектов: Письмо Аппарата ГД ФС РФ от 18.11.2003 N вн2-18/490. — Москва, 2003 г.
3. **Постановление** Законодательного собрания Красноярского края № 12-2575П "О методических рекомендациях по юридико-техническому оформлению краевых законопроектов". 2004.
4. **Holm N. T.** Geeks bearning gifts. How the computer world Got this way. Mindful Press, 2009.
5. http://www-01.ibm.com/software/lotus/notesanddomino/
6. **Рассел С., Норвиг П.** Искусственный интеллект: современный подход. — М.: Вильямс, 2007. 2-е изд. 1408 с.
7. **Vector** Markup Language (VML) // World Wide Web Consortium (W3C). — 13 5 1998 г. URL: http://www.w3.org/TR/NOTE-VML
8. **Отчет** о патентном исследовании относительно конструкторско-технологического решения "Визуализация взаимосвязей нормативно-правовых документов в виде графа". Новосибирск, ЗАО "Крепость Технопарк". 2009.

CONTENTS

In this paper, I will discuss the problem of aggregating the results of a distributed computation, in a multi processor network. As we know, data aggregation provides a challenge, especially when we deal with large sets of data points, in a noisy background. As a solution, I will represent a new parallel clustering algorithm, which employs a minimum spanning tree (MST) of the graph and solves data aggregation problem for GRID-resources. The computational bottleneck of my algorithm is the construction of an MST of a graph, for which a parallel algorithm is employed.

Keywords: GRID-computing, cluster, parallel clustering algorithm, hierarchical clustering, minimum spanning tree

The computer service in Grid-systems and multiprocessor computer systems demands an allocation of lengthy horizontal set of rectangular coordinate resource elements with variable sections of measurements along coordinate axes of integer-valued plane into a square frame of resource field of computer system. The length of an array of service demands mentioned above exceeds the number of measurements of resource field. Here arises the task of symmetric localization of linearly lengthy set of resource elements into resource shell as a subset of the frame mentioned above. The task of localization complies with the paired goal criteria of the resource shell measurements asymmetry minimization and the maximization of coefficient of shell filling of given resource rectangles of demands array. It is considered the localization algorithms which depend on quadratic type of demands array. It is introduced the definitions of circle, hyperbolic, parabolic arrays and values of localization indicators are set up.

Keywords: Grid-system, multiprocessor computer system, supervision, local schedule, optimal schedule, quadratic type of users demands array, the minimum of measurements asymmetry principle of comprehensive resource rectangle

The technique of definition of limiting length of a reclaiming site of xDSL-modems of a network of user's access depending on a total handicap and from transitive attenuations is developed at parallel work on one cable of two polytypic modems, each of which works on separate pair in an one-strip duplex mode. The received technique is suitable also for variants of construction of a network with any number of in parallel working modems with digital linear signals which have any speeds of transfer and use one-strip duplex transfer on one or more cable pairs.

Keywords: the xDSL-modem, the user's access symmetric and asymmetric access, the reclaiming site, directing system

This article offers a perfect threshold secret sharing scheme for an elliptic curve over Z_q, where $q = \prod_{i=1}^{s} p_i$, p_i — distinct primes and $p_i > 3$ for all $i = 1...s$.

Keywords: elliptic curve, elliptic curve cryptography, threshold secret sharing schemes on the elliptic curve

We consider the formal specification and automated verification of transaction processing systems used in distributed databases. In such systems, a standard set of ACID properties must be ensured by a combination of concurrency control and recovery protocols. In the existing literature, these protocols are usually studied in isolation, making a lot of assumptions about each other, and the problem of interaction among them is largely ignored. To study the formal verification of a set of combined protocols, we specify a transaction processing system, integrating strict two-phase locking, undo/redo recovery protocol and two-phase atomic commitment. We proved with the interactive theorem prover PVS that our system satisfies atomicity, durability and serializability properties.

Keywords: databases, concurrency control protocols, recovery protocols, fault-tolerance, formal specification, automated verification, interactive theorem prover

The method, which allows to develop a variety of structure-independent databases with using relational technologies are proposed, the requirements that must be satisfied by structure-independent databases are formulated, considered various options for implementation of the method.

Keywords: variability of data, database constructing method, data model, structure-independent database

This article tells about model of data which combine some principles of object-oriented and temporal approaches to developing of information systems.

Keywords: data model, object-oriented approach, temporal approach

We examine performance of local image features for retrieval of images with similar objects. Our image collection consists of images with visually similar objects, and duplicate images are excluded from this dataset. Objects are considered similar if they belong to the same semantic category. Several keypoint detectors are used to compute local features — SURF, Harris, Laplace, Hessian, Ridge, SIFT, Kadir-Brady detectors. Hessian detector show the best result since keypoint positions are more consistent with object position. Precision and recall for object localization slightly depend on amount of keypoints detected by Hessian detector.

Keywords: content-based image retrieval, cbir, interest points, keypoint detector, Hessian, Ridge, Laplace, Harris, difference of Gaussian, Laplacian of Gaussian, SIFT, PCA-SIFT

The output on the peripherals PC, based on digitizing and consistency of color spaces and adequately perceived by the most users, the color perception of which is close to standard, for a considerable part of users, having the anomalies of color vision, is not correct. Some colors, distinguished by the standard user, are taken by them as equal. The calculating problem of test for research of color perception and of digitizing color space of arbitrary PC users is being considered. Solution it is necessary to avoid perceptual aberration of color information.

Keywords: color periphery, standard color perception, anomalies of color vision, perceptual aberration of color information

The situation which has been developed in the Russian Federation, connected with the rapid growth of quantity of accepted documents of legislative character is considered. The significant part of the accepted laws is the nature of correction; it means that it contains references to other certificates. The analysis of a great number of documents with their interrelations is a difficult routine work, requiring the presence of highly skilled specialists. The analysis and visualization system of the relations of the normative legal documents suggests a complex approach to improvement of the legal-reference systems and electronic databases by integration into them of the system, realizing visual display of documents and the analysis correlation for the purpose of their conformity to the norms of lawmaking. The example of graphic construction of "vicinity" of one of the real legislative documents is resulted.

Keywords: normative legal document, analysis, visualization

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 *Т. В. Пчелкина*.
Сдано в набор 10.12.2010. Подписано в печать 21.01.2011. Формат 60×88 1/8. Бумага офсетная. Печать офсетная.
Усл. печ. л. 9,8. Уч.-изд. л. 11,00. Заказ 64. Цена договорная.

Журнал зарегистрирован в Министерстве Российской Федерации по делам печати, телерадиовещания и средств массовых коммуникаций.
Свидетельство о регистрации ПИ № 77-15565 от 02 июня 2003 г.

Отпечатано в ООО "Подольская Периодика"
142110, Московская обл., г. Подольск, ул. Кирова, 15