ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

5(177) 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 004.912

А. Ю. Бородащенко, преподаватель,
e-mail: bdy55@mail.ru
В. А. Яковлев, слушатель,
e-mail: r1der88@mail.ru
Академия ФСО России, г. Орел

Алгоритм фильтрации текстовой информации на основе марковской модели

Предложен алгоритм определения семантического расстояния между текстами в различных информационных системах на основе марковской модели. Данный алгоритм позволяет не только повысить полноту и точность поиска информации в текстовых массивах, но и определить количественное значение меры близости между документами. Алгоритм реализует функцию поиска с заданной точностью. Для оценки эффективности предполагаемого алгоритма предложен оригинальный подход сопоставления результатов поиска.

Ключевые слова: *семантическое расстояние, марковская модель, мера близости, алгоритм фильтрации текстов*

Одной из важнейших задач, которую приходится ежедневно решать организациям, крупным компаниям, информационно-аналитическим подразделениям предприятий, является быстрый поиск документов в больших объемах данных. При этом, как правило, речь идет не о скорости работы самой системы-поисковика, а о времени поисковой сессии, т. е. времени, которое затрачивает оператор на получение нужной ему информации. Обычный фразовый поиск, реализованный во всех существующих информационно-поисковых системах, не представляет возможностей для снижения временных затрат при поиске. Основные проблемы: невозможность корректного задания поискового запроса с использованием ключевых слов и попадание в список документов, не относящихся к информационным потребностям пользователя. Повысить качество поиска можно путем использования функции семантической фильтрации, обеспечивающей отбор из массива документов, содержание которых подобно некоторому эталонному тексту [1]. Это позволяет сократить время поисковой сессии до минимума и дает требуемые пользователю результаты выполнения поискового запроса.

Одной из проблем, с которой также может столкнуться пользователь, является отсутствие количественного значения близости между найденными текстами. Информационные системы в результате поиска выводят множество документов, но у оператора может отсутствовать время на просмотр каждого из них. Расчет расстояния между эталонным текстом и найденным существенно уменьшает список просматриваемых документов и соответственно сокращает время поисковой сессии.

Вычисление близости между документами дает возможность отбросить наименее релевантные документы, а также осуществить поиск в заданном диапазоне, определяемом соответствующим коэффициентом.

Таким образом, в качестве одной из очевидных тенденций в развитии современных информационно-поисковых систем можно выделить внедрение методов фильтрации поисковых запросов на основе оценки семантического сходства (подобия) с эталоном [2]. В статье рассматривается один из возможных подходов к семантической обработке текстов, формализованный в виде алгоритма фильтрации текстовой информации на основе марковской модели. С помощью этого алгоритма определяется расстояние между текстовыми документами (мера подобия).

Определение семантического расстояния выполняется в два этапа:

- обработка входных текстовых данных;
- сравнение матриц марковских связей двух текстов.

На этапе обработки входных данных происходит создание матрицы марковских связей для каждого документа и текста-эталона. Имена колонок и строк являются словами из текста. Для каждого текста получается собственная матрица, содержащая информацию о вероятности повторений последовательности двух слов из текста, в которой первое слово — имя строки, а второе слово — имя колонки. В ходе обработки при появлении нового слова в тексте происходит добавление колонки и строки в матрице. Для исключения лишних словосочетаний в данной части обработки необходимо использовать фильтры, с помощью которых отбрасываются малозначащие слова, выполняется морфологический анализ, обеспечивающий приведение слов и словосочетаний к каноническому виду.

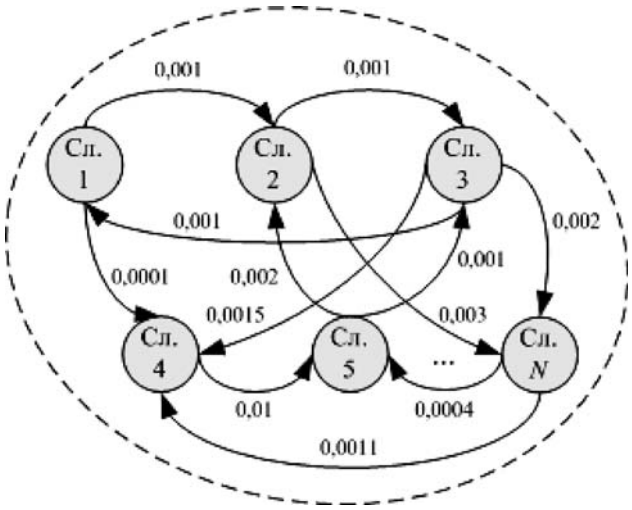


Рис. 1. Графическая иллюстрация марковской модели текста

Ниже представлен пример матрицы марковских связей внутренней структуры текста.

$$P_{kl} = \begin{vmatrix} 0 & 0{,}001 & 0 & 0{,}0001 & 0 & \dots & 0 \\ 0 & 0 & 0{,}001 & 0 & 0 & \dots & 0{,}003 \\ 0{,}001 & 0 & 0 & 0{,}0015 & 0 & \dots & 0{,}002 \\ 0 & 0 & 0 & 0 & 0{,}01 & \dots & 0 \\ 0 & 0{,}002 & 0{,}001 & 0 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0{,}0011 & 0{,}0004 & \dots & 0 \end{vmatrix}$$

Графическая иллюстрация марковской модели текста приведена на рис. 1.

Для выполнения этапа сравнения необходимо привести матрицы вероятностей документов и эталона к одному основанию. Для этого в матрицу каждого документа нужно добавить строки и столбцы, имена которых есть в матрице эталона, но нет в матрице документа и наоборот. Полученное множество ячеек заполняется нулями и на результат сравнения не влияет.

Этап сравнения формально выражен формулой, которая показывает расстояние между марковской моделью первого документа и марковской моделью второго документа (величина неподобия моделей) [3]:

$$D(\lambda_Y, \lambda_X) = \frac{1}{T}\Sigma_T[\log_a P(O_T^{(Y)}|\lambda_Y) - \log_a P(O_T^{(Y)}|\lambda_X)], \quad (1)$$

где $O_T^{(Y)} = O_1, O_2, O_3, \dots, O_T$ — последовательность наблюдений, порожденных моделью λ_Y, т. е. последовательность пар слов, из которых состоит текст Y.

Данная формула является мерой того, насколько марковская модель λ_X документа X согласуется с наблюдениями, порожденными марковской моделью λ_Y документа Y, в сравнении с тем, насколько хорошо модель λ_Y согласуется с наблюдениями, порожденными ей самой. Неудобством данной меры является то, что она несимметрична. Симметричным вариантом этой меры является величина

$$D_S(\lambda_Y, \lambda_X) = \frac{D(\lambda_Y, \lambda_X) + D(\lambda_X, \lambda_Y)}{2}, \quad (2)$$

где $D(\lambda_Y, \lambda_X)$ — расстояние между моделями λ_Y и λ_X; $D(\lambda_X, \lambda_Y)$ — расстояние между моделями λ_X и λ_Y.

В результате сравнения текстов получается нормированное число (от 0 до 1), показывающее сходство между двумя документами:

$$F = 1 - D_s(\lambda_Y, \lambda_X). \quad (3)$$

Формальная постановка задачи определения семантического расстояния имеет следующий вид.

Заданы: массив эталонных текстов X, отражающих семантику требуемых результатов поиска; массив произвольных текстов Y, которые требуется обработать. Каждый массив состоит минимум из одного документа.

Требуется: формализовать публикацию СМИ Y_j относительно X_i, т. е. определить значение семантического расстояния между текстами.

Необходимо найти:

T — число состояний модели текста;

P_{kl} — распределение вероятностей переходов между состояниями, где k, l — соответствующие слова текста.

Решающее правило оценки подобия: множество пороговых оценок семантического сходства $F(X_j, Y_i)$ для каждой пары текстов (Y_j, X_i):

$$F = 1 - D_s(\lambda_Y, \lambda_X).$$

На рис. 2 представлен алгоритм фильтрации текстовой информации на основе марковской модели.

В блоке 1 осуществляется загрузка текстов в программу. Установка параметров предварительного анализа текстов происходит в блоке 2. В блоке 3 выполняются процедуры предварительного анализа: удаление малозначащих слов с помощью словаря стоп-слов; морфологический анализ для слов документа; приведение слов и словосочетаний к каноническому виду. Построение матриц вероятностей марковских связей каждого текста реализуется в блоке 4. В блоке 5 осуществляется приведение матриц вероятностей выбранной пары текстов к общему основанию. В блоке 6 рассчитываются расстояния между марковскими моделями пары текстов (текста Y и текста X, текста X и текста Y) по формуле (1). Симметричное расстояние между парой текстов определяется в блоке 7 по формуле (2). Данное расстояние определяет степень удаленности текстов. В блоке 8 рассчитывается коэффициент близости текстов [выражение (3)]. В блоке 9 осуществляется визуализация результатов анализа.

В качестве примера рассмотрим порядок обработки двух документов.

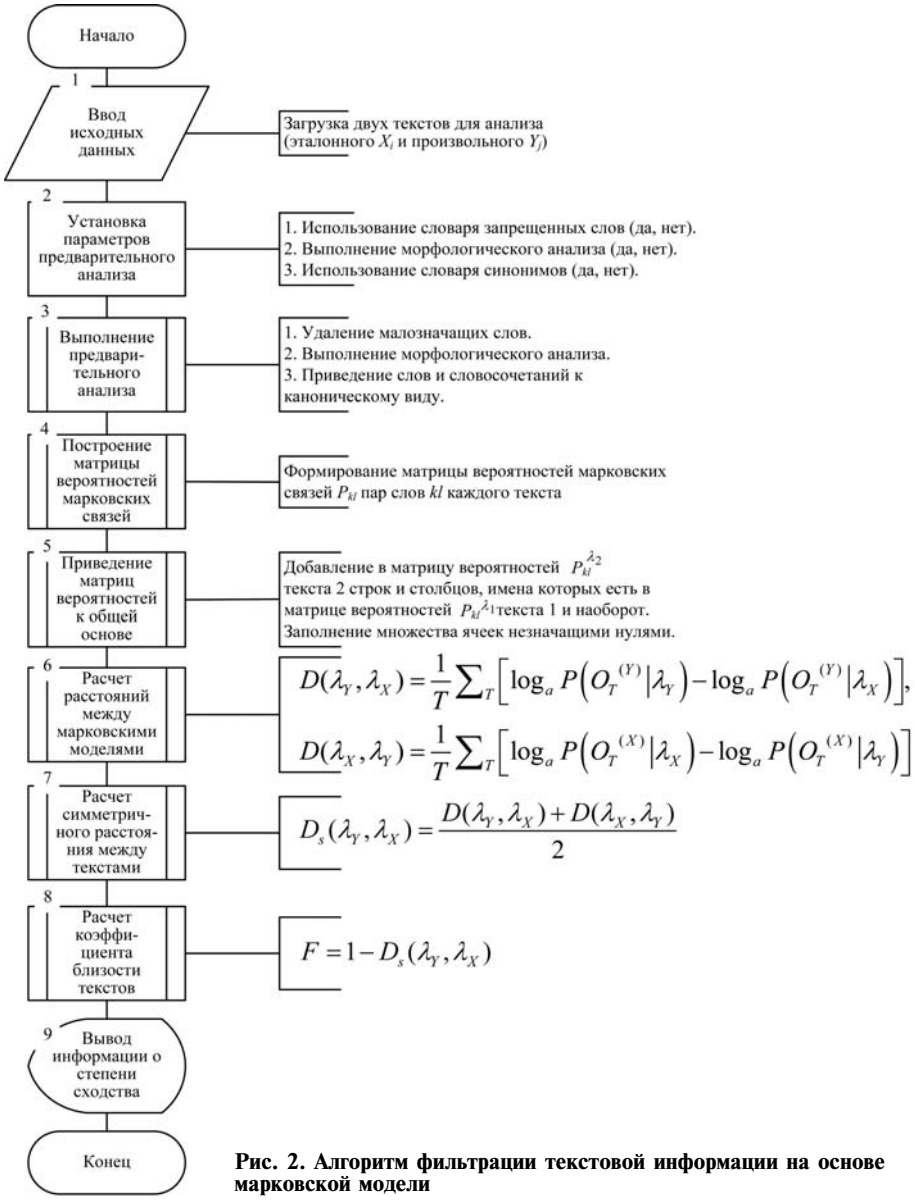


Рис. 2. Алгоритм фильтрации текстовой информации на основе марковской модели

Документ № 1:

"Ограничения импорта, закупок и производства.

В связи с необходимостью экономии электроэнергии и сокращения выброса углекислого газа в атмосферу, во многих странах введен или планируется ввод запрета на производство, закупку и импорт ламп накаливания с целью стимулирования замены их на энергосберегающие лампы (компактные люминесцентные лампы и др.).

С 1 сентября 2009 года в Евросоюзе в соответствии с директивой 2005/32/EG вступил в силу поэтапный запрет на производство, закупку магазинами и импорт ламп накаливания (за исключением специальных ламп). С 2009 г. запрет коснется ламп мощностью ⩾100 Вт, ламп с матовой колбой ⩾75 Вт и др.; ожидается, что к 2012 году будет запрещен импорт и производство ламп накаливания меньшей мощности.

В России правительство Москвы с 2011 года также планирует исключить из оборота и прекратить производство ламп накаливания мощностью ⩾100 Вт.

С 2005 г. на Кубе ограничено использование ламп накаливания мощностью более 15 Вт.

С 2009 г. ограничения коснутся также Новой Зеландии и Швейцарии, с 2010 г. — Австралии.

23 ноября 2009 года президент России подписал принятый ранее Госдумой закон "Об энергосбережении и повышении энергетической эффективности и о внесении изменений в отдельные законодательные акты Российской Федерации" [14]. Согласно документу, с 1 января 2011 года к обороту на территории страны не допускается продажа электрических ламп накаливания мощностью 100 Вт и более; с 1 января 2013 года — электроламп мощностью 75 Вт и более, а с 1 января 2014 года — ламп мощностью 25 Вт и более".

Документ № 2:

"23 ноября 2009 года вступил в действие закон "Об энергосбережении и повышении энергетической эффективности и о внесении изменений в отдельные законодательные акты Российской Федерации". Согласно документу, с 1 января 2011 г. к обороту на территории страны не допускаются электрические лампы накаливания мощностью 100 Вт и более. С 1 января 2013 г. может быть введен запрет на оборот на территории РФ электроламп мощностью 75 Вт и более, а с 1 января 2014 г. — ламп мощностью 25 Вт и более.

Федеральным законом регулируются отношения по энергосбережению и повышению энергетической эффективности, и в частности, вводятся ограничения на оборот ламп накаливания, устанавливаются требования по маркировке товаров с учетом их энергетической эффективности, по обязательному коммерческому учету энергетических ресурсов, по энергетической эффективности новых зданий (строений, сооружений), по сокращению бюджетных расходов на приобретение энергетических ресурсов, а также по содержанию общего имущества многоквартирных домов в отношении показателей энергетической эффективности. Кроме того, Федеральным законом предусматриваются введение для наиболее

энергоемких организаций обязанности по приведению энергетических обследований, утверждение программ по энергосбережению и повышению энергетической эффективности в бюджетной сфере, введение в гражданский оборот энергосервисных контрактов, переход к долгосрочному тарифному регулированию и создание единой (межведомственной) информационно-аналитической системы по энергетической эффективности.

По оценкам Минэкономразвития, общая стоимость замены ламп накаливания на энергосберегающие лампы составит до 2014 г. около 100 млрд руб. При этом строительство заводов по утилизации люминесцентных ламп входит в эту сумму".

На шаге 2 устанавливаются параметры предварительного анализа текстов:

- удаление малозначащих слов — вкл.;
- морфологический анализ — вкл.;
- приведение слов и словосочетаний к каноническому виду — вкл., реализуемые на шаге 3 работы алгоритма в специальном программном модуле.

На шаге 4 в два этапа осуществляется построение матрицы вероятностей марковских связей:

- формируется матрица A_{kl}, показывающая число повторений слова k после слова l текста;
- непосредственное формирование матрицы вероятностей P_{kl} путем пересчета значений матрицы A_{kl} по формуле

$$P_{kl} = \frac{A_{kl}}{\sum_{i=1}^{n} A_{k_i}},$$

где $\sum_{i=1}^{n} A_{k_i}$ — число пар слов, входящих в соответствующий текст с учетом шага 2 и 3 (документ № 1 — 95, документ № 2 — 105).

На шаге 5 в матрицы вероятностей марковских связей документа № 1 и документа № 2 осуществляется добавление недостающих строк и столбцов с заполнением новых ячеек незначащими нулями. Тем самым уравниваются основания матриц, длина которых для данных текстов равняется $T = 170$.

На шаге 6 рассчитываются значения

$$D(\lambda_Y, \lambda_X) \approx 0{,}21 \text{ и } D(\lambda_X, \lambda_Y) \approx 0{,}15.$$

На шаге 7 осуществляется расчет симметричного расстояния между документами:

$$D_s(\lambda_Y, \lambda_X) = \frac{0{,}21 + 0{,}15}{2} = 0{,}18.$$

На шаге 8 определяется значение коэффициента близости обрабатываемых документов: $F = 0{,}82$. Это значение говорит о высоком смысловом соответствии документа № 1 и документа № 2.

Рис. 3. Экранная форма матрицы вероятностей марковских связей текстового документа

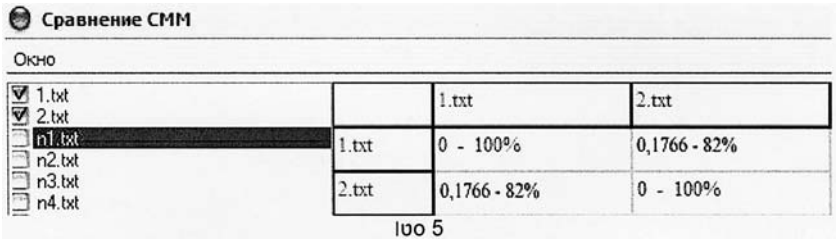


Рис. 5. Экранная форма результата сравнения массива текстовых документов с эталоном

Экранные формы пользовательского интерфейса прототипа программного модуля фильтрации текстовой информации представлены на рис. 3, рис. 4 (см. третью сторону обложки) и рис. 5.

Оценка алгоритма осуществлялась путем сравнения с программным продуктом SearchInform компании СофтИнформ [4], реализующим функцию поиска подобных документов, а также с принятой для таких систем практикой сравнения результатов с мнением экспертов. Разработанный макет обеспечивает определение семантических расстояний между текстами, соответствующих по значениям "эталонным" мнениям экспертов. Программа-аналог SearchInform выдает заниженные результаты степени подобия для рассматриваемых пар документов.

Таким образом, сравнение результатов тестирования разработанного авторами макета программы фильтрации текстов с существующим аналогом позволяет сделать вывод о том, что представленный в статье алгоритм оценки меры близости пары текстов с использованием скрытых марковских процессов является актуальным, а его применение в целом позволяет повысить качество соответствующих алгоритмов обработки текстовой информации.

Список литературы

1. **Бородащенко А. Ю., Бочков М. В., Салбиев А.Л.** Алгоритм оценки массива текстов на семантическое сходство с эталоном // Информационные технологии. 2008. № 12. С. 8—11.
2. **Ланде Д. В.** Глубинный анализ текстов. Технология эффективного анализа текстовых данных: — Персональный сайт Дмитрия Ландэ, 2009. URL: http://dwl.kiev.ua/art/dz/index.html, свободный.
3. **Рабинер Л. Р.** Скрытые марковские модели и их применение в избранных приложениях при распознавании речи: Обзор. // ТИИЭР. Т. 77, № 2, 1989. С. 86—120.
4. **SearchInform** Server. Функциональная спецификация. — М.: Компания "СофтИнформ", 2008. 26 с.

УДК 519.682

А. В. Мочалова, магистр,
Санкт-Петербургский государственный университет аэрокосмического приборостроения (ГУАП),
e-mail: itfru@mail.ru
В. А. Мочалов, аспирант,
Московский технический университет связи и информатики (МТУСИ),
e-mail: mvaproduct@mail.ru

Интеллектуальная вопросно-ответная система

Рассматриваются алгоритмы выявления логических связей между словами предложения и методы их хранения в базе данных. Предлагаются алгоритмы работы системы, отвечающей на вопросы, задаваемые на естественном языке к естественно-языковым текстам, а также алгоритмы ответа на вопрос с вопросительным словом и без такового. Изложенные в статье результаты могут быть использованы в информационных и аналитических системах, поисковых машинах, СУБД и интерпретаторах.

Ключевые слова: *вопросно-ответная система, диалоговая система, извлечение фактов из текста*

Введение

В конце 60-х годов в исследованиях по искусственному интеллекту сформировалось самостоятельное направление, получившее название "обработка естественного языка" (Natural Language Processing) [1]. Поиск знаний в тексте — это нетривиальный процесс обнаружения действительно новых, потенциально полезных и понятных шаблонов в неструктурированных текстовых данных [2, 3]. В настоящее время все больше внимания уделяется системам анализа текстов на естественном языке и извлечению из них фактов и знаний. Это связано со все возрастающим объемом информации, которым необходимо оперировать современным людям. Актуальным является создание системы, выдающей смысловой ответ на вопрос, задаваемый по тексту на естественном языке. Одной из важнейших задач анализа текстов является задача определения логических связей между словами предложений. В данной статье рассматриваются алгоритмы выявления подобных связей, а также структура базы данных, хранящей их. Также в статье предлагаются алгоритмы работы системы, отвечающей на вопросы, задаваемые на естественном языке к естественно-языковым текстам.

Используемые термины

Семантическая зависимость. Будем говорить, что два различных слова α и β из одного предложения связывает семантическая зависимость с именем R (обозначим $R(\alpha, \beta)$), если между α и β существует некая универсальная бинарная связь [4].

Для конкретных слов α, β и зависимости $R(\alpha, \beta)$ направление выбирается таким образом, чтобы формула $R(\alpha, \beta)$ была эквивалентна утверждению, что "α является R для β". Соответственно, формула $R(\beta, \alpha)$ должна быть эквивалентна утверждению "β является R для α". Например, для фразы "роман Толстого" будет построена формула АВТОР(Толстой, роман), а не наоборот, потому что верно утверждение "Толстой является АВТОРом романа", а не наоборот. Вследствие этого в формуле $R(\alpha, \beta)$ будем иногда отождествлять R с α [4]. В идеале множество семантических зависимостей, используемое при машинном анализе текста, должно покрывать все возможные связи между частями текста.

Каждой семантической зависимости $R(\alpha, \beta)$ можно поставить в соответствие множество вопросительных слов, которые можно задать от имени семантической зависимости R к β. Например, приведенной выше зависимости АВТОР(Толстой, роман) можно поставить в соответствие вопросительное слово "кто": ***Кто*** АВТОР романа? Соответствие некоторого вопросительного слова q семантической зависимости R обозначим $R{:}q$.

Ассоциативная связь. Будем говорить, что два различных слова α и β из анализируемого текста связаны ассоциативной связью, если существует некая закономерная связь между этими словами. Существуют специальные ассоциативные словари, в которых описаны подобные связи между словами. Как правило, подобные словари формируются на базе анализа опросов [5], а также посредством анализа различных текстов. Некоторые ассоциативные словари оперируют понятием *веса ассоциативной связи*, который определяется как некая целочисленная величина, характеризующая значимость ассоциативной связи.

Алгоритмы выявления логических связей в тексте

Виды связей предложений

В русском языке различают два типа связи предложений в тексте: цепную и параллельную.

Цепная связь последовательно связывает второе предложение с первым, третье со вторым и т. д. Она определяется повтором ключевого слова, заменой его синонимом, синонимическим оборотом, местоимением, наречием, повтором того или иного члена предложения. Цепная связь обусловлена че-

редованием "данного" и "нового", мысль автора развивается последовательно; то, что в первом предложении было "новым", во втором становится "данным" и т. д. Связь может осуществляться путем семантических соответствий или ассоциаций.

При *параллельной связи* происходит соподчинение второго, третьего и т. д. предложений первому, т. е. первое предложение внутренне скрепляет все предложения в единое целое.

В параллельной связи, образно говоря, первое предложение обычно задает картину в целом, а все последующие предложения описывают отдельные части этой картины. Таким образом, логические связи в первую очередь имеет смысл искать лишь в тексте, логическая связность которого достигается посредством цепной связи. Алгоритмы нахождения подобных логических связей описаны в следующих разделах.

Таблица эквивалентности

Определим задачу выявления логических связей в тексте как задачу нахождения множества всех различных слов или словосочетаний, которые в контексте данного текста обозначают один и тот же объект или являются различными определениями одного и того же объекта. Слова (или словосочетания) из такого множества будем называть *эквивалентными* относительно друг друга. Пара эквивалентных слов может встречаться как внутри одного предложения, так и в двух различных предложениях (не обязательно соседних), когда одно слово из пары находится в одном предложении, а другое — во втором.

Множество всех эквивалентных слов, обозначающих или характеризующих один и тот же объект в рамках анализируемого текста, назовем *классом эквивалентности* данного объекта.

Для хранения классов эквивалентности, построенных по заданному тексту, удобно использовать таблицу (будем называть ее таблицей эквивалентности и обозначим буквой E), в каждой ячейке $e[i, j]$ которой находится слово, принадлежащее одному из классов эквивалентности (ячейка $e[i, j]$ находится в i-й строке, j-м столбце таблицы).

Помимо самой таблицы E необходимо хранить две вспомогательные таблицы, которые будут содержать полную информацию о местоположении эквивалентных слов в анализируемом тексте:

- таблица N, каждая ячейка $n[i, j] = N_{ij}$ которой содержит порядковый номер предложения из анализируемого текста, которому принадлежит слово из $e[i, j]$;
- таблица M, каждая ячейка $m[i, j] = M_{ij}$ которой содержит порядковый номер слова из $e[i, j]$ в предложении, которому принадлежит это слово.

Слова в таблице эквивалентности упорядочены по строкам в порядке их расположения в исходном тексте, каждая строка в таблице эквивалентности содержит отдельный класс эквивалентности.

Введем следующие обозначения:

- E^i — класс эквивалентности с порядковым номером i (т. е. все слова из i-й строки таблицы E);
- $e[i, j] = E^i[M_{i,j}, N_{i,j}]$ — слово из таблицы эквивалентности E, находящееся в i-й строке (т. е. принадлежащее E^i), j-м столбце; предложение, содержащее данное слово в анализируемом тексте, имеет порядковый номер $M_{i,j}$, а позиция слова $e[i, j]$ в этом предложении есть $N_{i,j}$.

Таблица E будет иметь столько строк, сколько в анализируемом тексте найдено классов эквивалентности (обозначим k — число строк в таблице E), а число столбцов будет равно наибольшей из мощностей всех классов эквивалентности (обозначим l — число столбцов в таблице E).

В заданных обозначениях общий вид ***таблицы эквивалентности*** будет следующим:

$E^1[M_{1,1}, N_{1,2}]$	...	$E^1[M_{1,l}, N_{1,l}]$
...	...	...
$E^k[M_{k,1}, N_{k,2}]$	...	$E^k[M_{k,l}, N_{k,l}]$

Очевидно, что при такой организации таблицы эквивалентности таблица E (а также M и N по аналогии) может содержать пустые ячейки: в каждой i-й строке будет столько пустых ячеек, на сколько мощность $|E^i|$ меньше максимальной из всех мощностей классов эквивалентности из таблицы E. Для уменьшения объема памяти, занятой для хранения таблиц E, M и N, строки этих таблиц можно хранить, например, в динамических массивах или использовать таблицы реляционной базы данных.

Средства связи частей текста

В русском языке выделяют следующие основные виды связей предложений в тексте: синонимический и лексический повторы, однокоренные и местоименные слова.

Выявить связей первого и второго типа — задача достаточно простая: для нахождения лексического повтора достаточно лишь найти совпадающие слова, выраженные самостоятельными частями речи, однако проблема может возникнуть в том случае, если найдены совпадающие слова, но фактически они обозначают или характеризуют разные объекты, что, например, может произойти при употреблении омонимов; связи, построенные посредством употребления однокоренных слов находятся с помощью словаря словоформ, а синонимические повторы можно найти с помощью словаря синонимов.

Наибольший интерес представляет задача нахождения связей, образуемых с помощью местоимений. Местоимения указывают на отдельные

слова или часть предшествующего текста, вводя тем самым их в следующее предложение. Местоименные слова позволяют избежать пагубного для общего слога текста повтора слов. Ниже будет представлен алгоритм нахождения подобных связей.

Алгоритм выявления логических связей в тексте

Местоимения, используемые для связи частей текста, как правило, заменяют слово или часть текста, указывая тем самым на них в последующем тексте.

Чаще всего местоимения связаны логически с существительным из предшествующей части текста (т. е. они обозначают один и тот же объект).

Рассмотрим наиболее часто встречающийся случай, когда требуется выявить логическую связь существительного из одного предложения и местоимения из другого. Для определенности рассмотрим два произвольных соседних предложения из некоторого связного текста. Необходимо установить связь между существительным из первого предложения (обозначим его T_p, где p — порядковый номер предложения T в анализируемом тексте) и местоимением из предложения T_{p+1}. Заметим, что как существительных в T_p, так и местоимений в T_{p+1}, может быть несколько. Пусть для определенности существительных в T_p будет s (обозначим их $C_1, C_2, ..., C_s$), а местоимений в T_{p+1} будет r (обозначим их $M_1, M_2, ..., M_r$):

$$\underbrace{C_1,\ C_2,\ ...,\ C_s}_{T_p}\ \ \underbrace{M_1,\ M_2,\ ...,\ M_r}_{T_{p+1}}.$$

Мы хотим выявить логические связи между существительными из T_p и местоимениями из T_{p+1}.

Введем следующие вспомогательные множества:

X — множество слов $\xi \in T_p$:$\exists R = R(C_i, \xi)$ для некоторого C_i, $i = 1, ..., s$;

Y — множество слов $\eta \in T_{p+1}$:$\exists R = R(M_i, \eta)$ для некоторого M_i, $i = 1, ..., r$;

$R(\alpha, \beta)$ — семантическая зависимость, связывающая два слова: α и β.

Найдем всевозможные пары слов (C_i, η):

$$\begin{cases} C_i \in T_p; \\ \eta \in Y; \\ \exists A(C_i, \eta), \end{cases} \quad (1)$$

где $A(C_i, \eta)$ — ассоциативная связь слов C_i и η. Пусть число таких пар равно P_1.

Найдем все возможные пары слов (ξ, η):

$$\begin{cases} \xi \in X; \\ \eta \in Y; \\ \exists A(\xi, \eta), \end{cases} \quad (2)$$

где $A(\xi, \eta)$ — ассоциативная связь слов ξ и η. Пусть число таких пар равно P_2.

В случае $P_1 = 0$ будем рассматривать лишь пары, определенные системой (2). Если $P = 1$, значит, существует единственная пара (C_i, η), для которой выполняется (1), и есть семантическая зависимость $R(M_j, \eta)$, связывающая M_j и η, из чего следует, что M_j и η принадлежат одному и тому же классу эквивалентности. Если $P_1 > 1$, значит, существует несколько пар вида (C_i, η), удовлетворяющих условиям системы (1); из них выбираем пару (или пары — их может быть несколько) с максимальным весом ассоциативной связи $A(\xi, \eta)$ (где все η из вышеописанных пар (C_i, η), а $\xi \in X$) и, найдя семантическую зависимость $R(M_j, \eta)$ (такая зависимость обязательно найдется, так как $\eta \in Y$), M_j и η относим к одному и тому же классу эквивалентности. Если нашлось несколько пар, удовлетворяющих системе (1), для которых выполняется условие максимального веса ассоциативной связи, то каждую соответствующую им пару $(C_i \in T_p, M_j \in T_{p+1})$ отнесем к одному и тому же классу эквивалентности.

Если одно и то же слово принадлежит нескольким различным классам эквивалентности, то очевидно, что истинным является только один из этих классов, и поэтому система при поиске ответа на вопрос должна выдать несколько вариантов ответа, последовательно предполагая истинность лишь одного из вышеописанных классов, после чего человек сможет проанализировать ответы системы и данные, на основе которых были построены эти ответы (в том числе и различные таблицы эквивалентности), и на базе этих данных выбрать правильный ответ.

Рассмотрим случай $P_1 = 0$. Если пар, удовлетворяющих системе (1) не найдено, то вывод о принадлежности пары $(C_i \in T_p, M_j \in T_{p+1})$ одному классу эквивалентности делается на основе анализа пар, определяемых системой (2).

Прежде чем приступить к анализу пар, определенных системой (2), из рассмотрения исключим местоимения и существительные, между которыми выявлена логическая связь, по результатам анализа пар, определенных системой (1). Структура данного анализа аналогична вышеописанному анализу пар, определенных системой (1).

Общая структура представления текста

Таким образом, анализируемый текст может быть представлен в виде "многослойного графа" (рис. 1), вершины которого — слова из анализируемого текста; ребра, нарисованные сплошными линиями, — семантические зависимости между словами внутри каждого отдельного предложения; ребра, нарисованные штриховыми линиями, — логические связи, хранящиеся в таблице эквивалентности; каждый порядковый номер "слоя" графа соответствует порядковому номеру предложе-

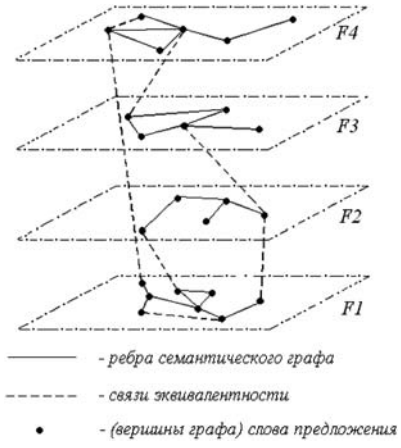


Рис. 1. Структура хранения текста

ния в тексте (слой $F1$ соответствует предложению T_1, слой $F2$ — предложению T_2 и т. д.).

Алгоритмы ответа на вопрос по тексту на естественном языке

Все вопросительные предложения русского языка условно можно разделить на два типа: предложения, содержащие вопросительное слово, и предложения, не содержащие такового. В зависимости от того, к какому из этих типов относится анализируемое вопросительное предложение, подаваемое на вход системы ответа на вопрос, мы предлагаем различные алгоритмы работы такой системы.

Данные алгоритмы базируются на семантическом анализе предложений, анализе семантических и ассоциативных зависимостей анализируемого текста; в ходе анализа также используются словари синонимов и таблица эквивалентности, построенная по анализируемому тексту, к которому задается вопрос. Для сокращения записи введем следующие обозначения:

- Q — вопросительное предложение;
- q — вопросительное слово из Q;
- Г — главное слово из Q (то, к которому задан вопрос);
- T — анализируемый текст;
- R — семантическая зависимость;
- $R(\alpha, \beta)$ — семантическая зависимость, связывающая два слова: α и β;
- RT — все семантические зависимости, построенные по анализируемому тексту T;
- RQ — все семантические зависимости, построенные по вопросительному предложению Q;
- $R{:}q$ — все семантические зависимости, которым соответствует вопросительное слово q;
- W — число предложений в анализируемом тексте T;
- $T = \{T_1, T_2, ..., T_W\}$, T_i — предложение с порядковым номером i из T;
- $T_{i,j}$ — слово с порядковым номером j из предложения $T_i \in T$;
- $Q = \{Q_1, ..., Q_N\}$, Q_i — слово с порядковым номером i из Q;
- $RT = \{RT_1, ..., RT_W\}$, RT_i — все семантические предложения, построенные по T_i;
- AQ — множество всех ассоциативных слов, построенных для Q;
- $a(x)$ — множество слов, ассоциативно связанных с x;
- $wa(x, y)$ — вес ассоциативной связи, связывающей два слова: x и y;
- SQ — множество всех синонимичных и эквивалентных слов для всех слов из Q;
- $s(x)$ — множество всех синонимов и все слова из класса эквивалентности для слова x.

Алгоритм ответа на вопрос без вопросительного слова

На вход системе подаются:

- вопросительное предложение Q на естественном языке, на которое система должна дать ответ;
- текст T на естественном языке, к которому задается вопрос Q;
- a, b, ζ, μ — некоторые настраиваемые параметры алгоритма, оптимальное значение которых устанавливается экспериментально путем анализа достаточного количества тестов.

По вопросительному предложению Q и по каждому невопросительному предложению анализируемого текста T_i, $i = 1, ..., W$, строятся все семантические зависимости: RQ и RT соответственно.

Множество семантических зависимостей RQ максимально дополняется уникальными семантическими зависимостями по следующей схеме: если некоторая семантическая зависимость $R(x, y) \in RQ$, то

$$s_i(x) \in T \Rightarrow RQ := RQ + R(s_i(x), y);$$

$$s_j(y) \in T \Rightarrow RQ := RQ + R(x, s_j(y));$$

$$s_i(x) \in T \text{ и } s_j(y) \in T \Rightarrow RQ := RQ + R(s_i(x), s_j(y)),$$

где $i = 1, ..., |s(x)|$, $j = 1, ..., |s(y)|$. Очевидно, что проверку на принадлежность $s_i(x) \in T$ и $s_j(y) \in T$ необходимо проводить лишь для $s_i(x)$, $s_j(y)$, не являющихся словами из классов эквивалентности слов x и y соответственно.

Далее для всех предложений $T_i \in T$, $i = 1, ..., W$, вычисляется значение величины $p_i = \dfrac{|RT_i \cap RQ|}{|RQ|}$. Если оказывается, что $p_i \geqslant \zeta$ (т. е. достаточное число семантических зависимостей, построенных по вопросительному предложению, совпало с семантическими зависимостями, построенными по i-му предложению анализируемого текста T, где достаточность определяется величиной ζ), очевидно, что чем ближе значение ζ к единице, тем досто-

вернее ответ "ДА", который должна выдать система в случае $p_i \geqslant \zeta$.

В случае $p_i < \zeta$ вычисляется величина

$$t_i := t_i + \sum_{j=1}^{N} \sum_{k=1}^{|T_i|} wa(Q_j, T_{i,k}), \qquad (3)$$

характеризующая число и веса ассоциативных связей слов из вопросительного предложения Q и слов из предложения T_i анализируемого текста T. Далее

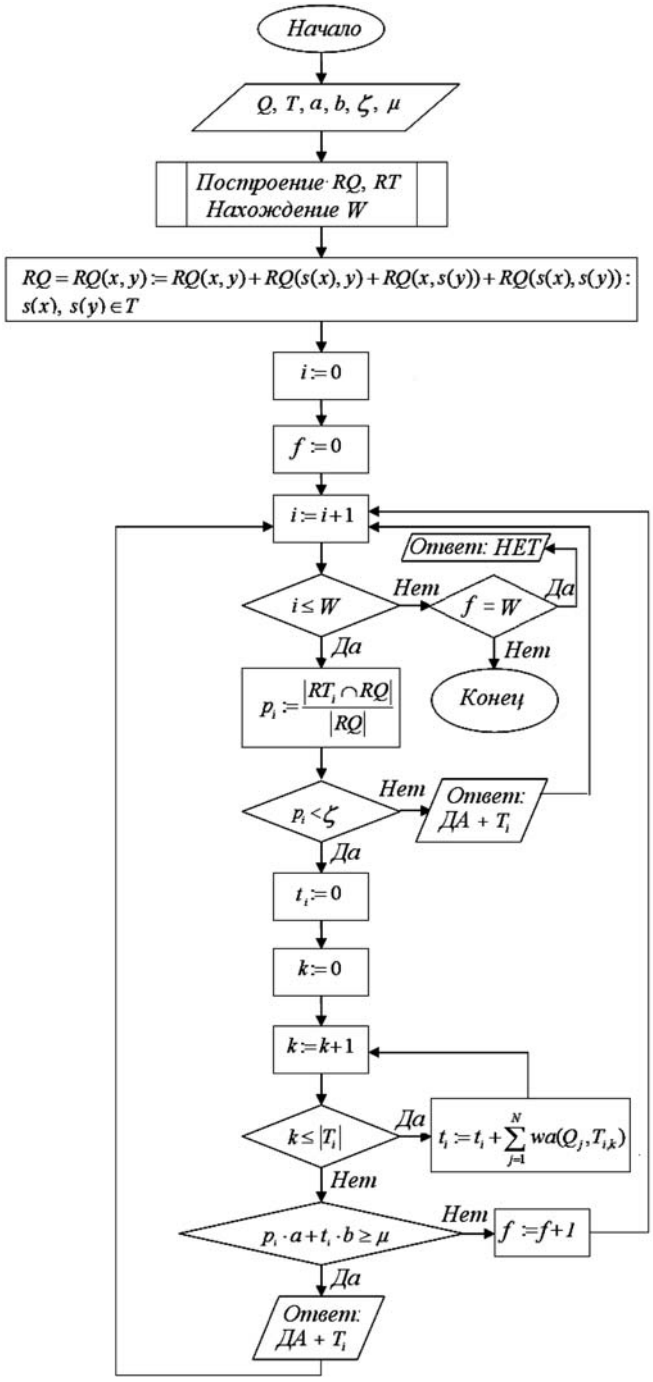


Рис. 2. Блок-схема алгоритма ответа на вопрос без вопросительного слова

подсчитываем значение $ap_i + bt_i$, где значимость параметров p_i и t_i определяется параметрами a и b (оптимальное значение параметров следует выяснить в ходе тестирования системы). В случае, если сумма $ap_i + bt_i \geqslant \mu$, система дает ответ "ДА", в противном случае система не дает никакого ответа на базе рассматриваемого i-го предложения.

Если в результате выполнения данного алгоритма система не дала положительного ответа ни для одного из предложений текста T, то система дает общий ответ "НЕТ".

Как видно из блок-схемы данного алгоритма (рис. 2), помимо самого положительного ответа, всякий раз выводится предложение, на базе которого был получен данный ответ.

Алгоритм ответа на вопрос с вопросительным словом

На вход системе подаются:

- вопросительное предложение Q на естественном языке, на которое система должна дать ответ;
- текст T на естественном языке, к которому задается вопрос Q;
- d_1, d_2, z, h, g, μ — некоторые настраиваемые параметры алгоритма, оптимальное значение которых устанавливается экспериментально путем анализа достаточного количества тестов.

Среди слов предложения Q с помощью специальных алгоритмов, которые в этой статье мы не приводим, находится главное слово Г, т. е. слово, к которому задается вопрос, выраженный вопросительным словом q.

По вопросительному предложению Q и по каждому невопросительному предложению анализируемого текста T_i, $i = 1, ..., W$, строятся все семантические зависимости: RQ и RT соответственно.

Далее для всех семантических зависимостей $RT_j(x_j, y_j)$, $j = 1, ..., |RT|$, построенных по исходному тексту T, проверяется выполнение условия:

$$[RT_j{:}q] \;\&\; [(x_j \in Q) \| (s(x_j) \in Q)] \;\&\; [y_j \in T].$$

Все $RT_j(x_j, y_j)$, удовлетворяющие этому условию, помещаются в некоторое множество RT_L.

Если $RT_L \neq \varnothing$ и среди семантических зависимостей есть зависимости, построенные по нескольким разным предложениям из анализируемого текста T, то все эти предложения для удобства дальнейшего анализа помещаются во множество F. Из F выбирается не более d_1 предложений (из которых формируется множество $\Pi = \{\Pi_1, ..., \Pi_{d_1}\}$), для которых сумма

$$|RF_i(x, y) \cap RQ(x, y)| + |RF_i(x, y) \cap RQ(s(x), y)| + |RF_i(x, y) \cap RQ(x, s(y))| + |RF_i(x, y) \cap RQ(s(x), s(y))| \qquad (4)$$

максимальна (где RF — все семантические зависимости, построенные по предложениям из множе-

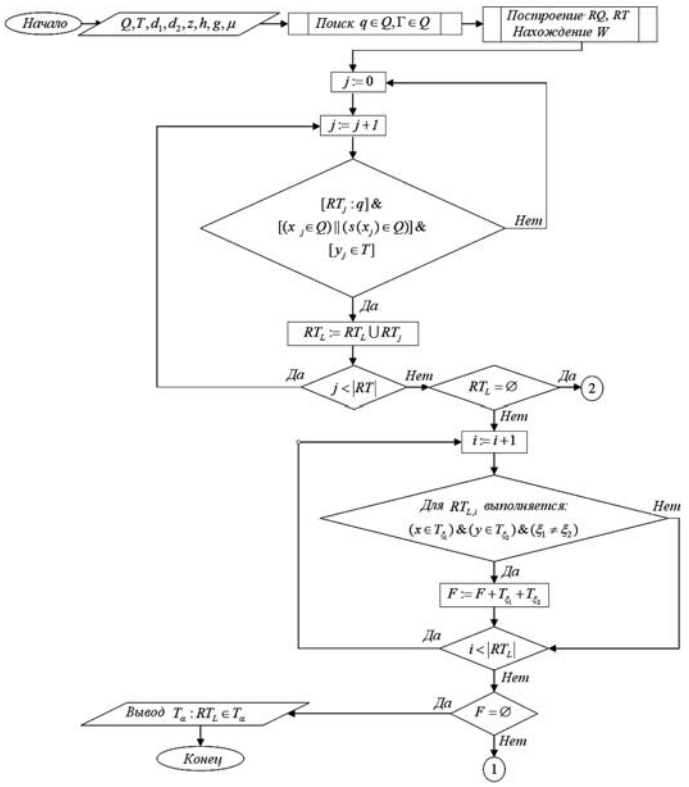


Рис. 3. Блок-схема алгоритма ответа на вопрос с вопросительным словом (часть 1)

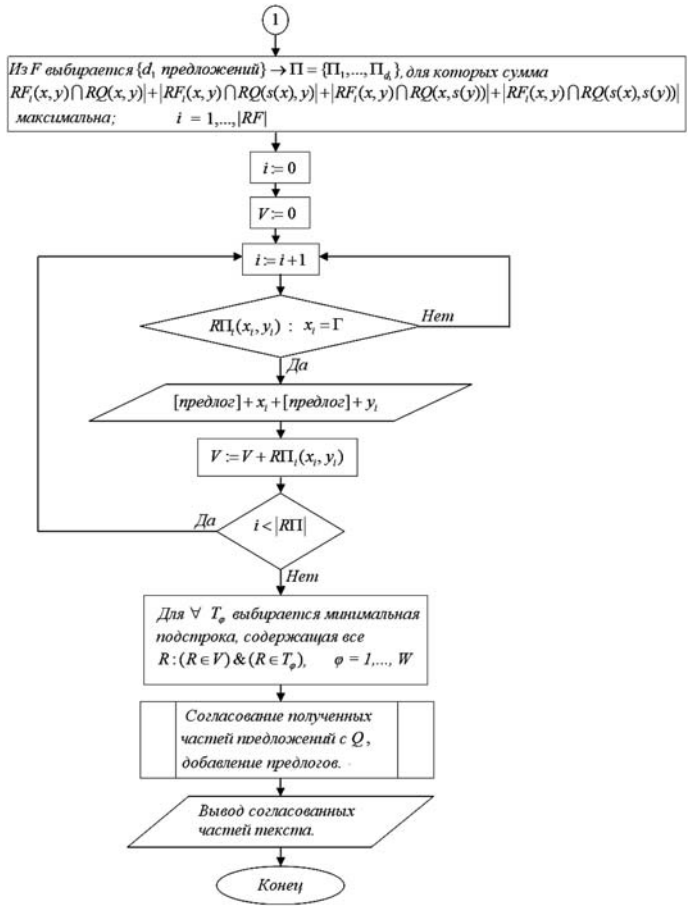


Рис. 4. Блок-схема алгоритма ответа на вопрос с вопросительным словом (часть 2)

ства F; RF_i — все семантические зависимости, построенные по i-му предложению из RF, $i = 1, ..., |RF|$; $|RF|$ — число предложений во множестве RF), после чего последовательно рассматриваются предложения из множества П.

Если среди $R\Pi$ ($R\Pi$ — множество семантических зависимостей, построенных по предложениям из П; $R\Pi_i$ — множество семантических зависимостей, построенных для i-го предложения из П) найдется $R\Pi_i(x_i, y_i):x_i = \Gamma$, то в качестве краткого ответа выводится словосочетание (x_i, y_i) с соответствующими предлогами из анализируемого текста T (если они есть). Из каждого предложения Π_i выбирается набор из минимального числа подряд идущих слов, содержащих все зависимости $R{:}q$, определяемые вопросительным словом $q \in Q$. Полученный набор слов из Π_i дополняется соответствующими предлогами из T (если они есть) и согла-

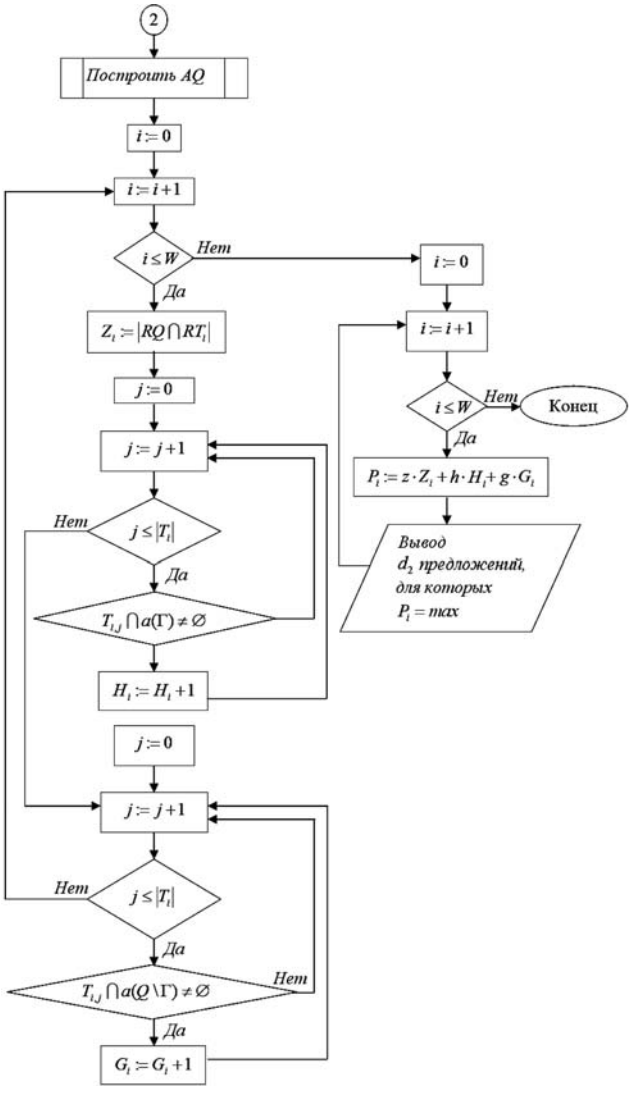


Рис. 5. Блок-схема алгоритма ответа на вопрос с вопросительным словом (часть 3)

совывается с вопросительным предложением Q, после чего выводится в качестве ответа.

Если $RT_L = \varnothing$, то для вопросительного предложения Q строится множество всех ассоциативных слов AQ. Далее для всех предложений $T_i \in T$, $i = 1, ..., W$, находятся значения величин

$$Z_i = |RQ \cap RT_i|,$$
$$H_i = |T_i \cap a(\Gamma)|,\ G_i = |T_i \cap a(Q\backslash\Gamma)|. \qquad (5)$$

Для каждого предложения T_i исходного текста T вычисляется величина

$$p_i = zZ_i + hH_i + gG_i, \qquad (6)$$

где Z_i, H_i и G_i определены в формуле (5), a, z, h и g — настраиваемые параметры алгоритма, оптимальное значение которых необходимо найти в результате анализа работы алгоритма при различных значениях этих величин.

В качестве ответа выводится не более d_2 предложений, с наибольшими значениями p_i. Блок-схема данного алгоритма представлена на рис. 3, 4, 5.

Заключение

Изложенные в статье результаты представляют значительный теоретический и практический интерес и могут быть использованы в информационных и аналитических системах, поисковых машинах, СУБД, интерпретаторах и интеллектуальном агенте-менеджере [6]. Информацию о текущем состоянии реализации интеллектуальной вопросно-ответной системы можно получить на авторском портале искусственного интеллекта [7].

Список литературы

1. **Попов Э. В.** Экспертные системы: Решение неформализованных задач в диалоге с ЭВМ. Сер. Проблемы искусственного интеллекта. М.: Наука, 1987. 288 с.

2. **Fayyad U., Piatetsky-Shapiro G.** From Data Mining to knowledge discovery: An overview // Advances in Knowledge Discovery and Data Mining, AAAI/MIT Press, 1996.

3. **Барсегян А. А.** Технологии анализа данных: Data Mining Visual Mining, Text Mining, OLAP / А. А. Барсегян, М. С. Куприянов, В. В. Степаненко, И. И. Холод. 2-е изд., перераб. и доп. СПб.: БХВ-Петербург, 2007. 384 с.

4. **Автоматическая** обработка текста. URL: http://www.aot.ru

5. **Добровольский Д. О., Караулов Ю. Н.** Ассоциативный фразеологический словарь русского языка. М.: Помовский и партнеры, 1994.

6. **Мочалов В. А., Старкова А. В.** Интеллектуальный агент-менеджер // Тр. конгресса по интеллектуальным системам и информационным технологиям "AIS-IT'09". Научное изд. в 4-х томах. М.: Физматлит, 2009. Т. 3. С. 261—268.

7. **Портал** искусственного интеллекта. URL: http://www.it-fru.ru/

УДК 004.3

Д. А. Назаров, аспирант,
Уфимский государственный
авиационный технический университет
e-mail: denis.nsc@gmail.com

Задача двухмерной упаковки в полосу: точный алгоритм с разбиением на лестничные частичные упаковки

Рассматривается задача двухмерной упаковки в полубесконечную полосу, которая принадлежит к классу NP-трудных задач. Предлагается точный алгоритм, который позволяет сократить время на доказательство оптимальности ранее найденного решения по сравнению с другими точными алгоритмами. Алгоритм основывается на возможности разбиения произвольной прямоугольной упаковки на две лестничные частичные упаковки.

Ключевые слова: *NP-трудные задачи, двухмерная упаковка в полосу, метод ветвей и границ*

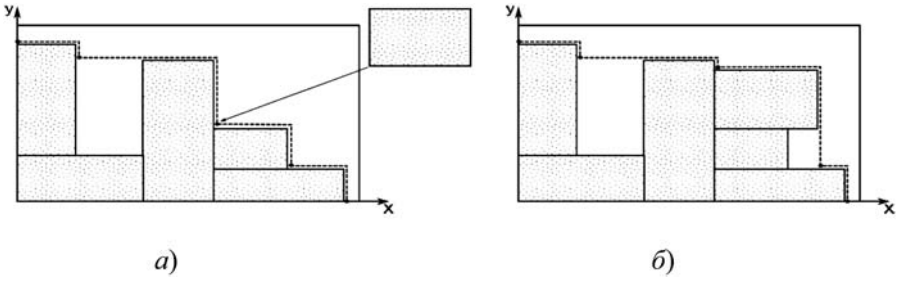
Введение

Задача двухмерной упаковки в полубесконечную полосу (*2-dimensional strip packing problem*, **2SPP**) заключается в размещении множества из n прямоугольников заданной ширины w_i и высоты h_i в полосу ширины W и неограниченной высоты так, чтобы прямоугольники не перекрывались между собой и не выходили за границы полосы. Размещение прямоугольников в полосе, удовлетворяющее этим требованиям, называют допустимой прямоугольной упаковкой (*rectangular pocking*, RP). Необходимо найти допустимую упаковку с минимальной занятой высотой H полосы. Задача рассматривается при условиях целочисленности размеров всех объектов и фиксированной ориентации прямоугольников в полосе. Задача **2SPP** является NP-трудной задачей. M. Garey и D. Johnson доказали NP-трудность для случая одномерной упаковки [1]. Существующие точные алгоритмы в общем случае находят оптимум для задач с небольшим числом заготовок. Зачастую оптимальное решение находится довольно быстро, и основное время уходит на доказательство его оптимальности. Предложенный алгоритм позволяет существенно сократить это время.

Лестничные частичные упаковки

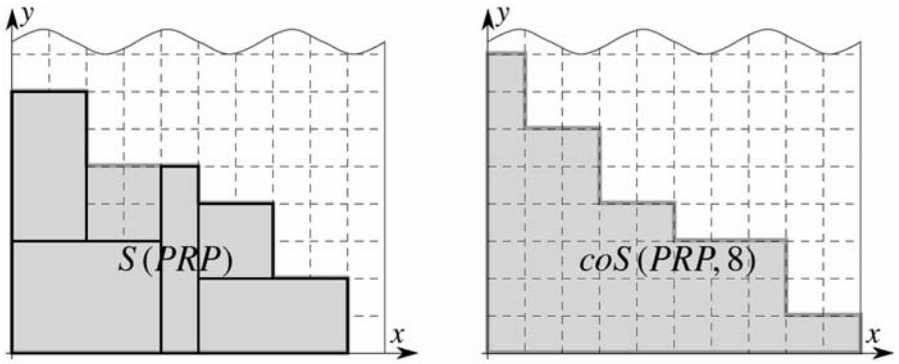
При решении **2SPP** входной информацией является $\langle W, n, w, h\rangle$. Выходом служит $<x, y>$, $x = (x_1, ..., x_n)$, $y = (y_1, ..., y_n)$, где (x_i, y_i) — координаты левого нижнего угла i-го прямоугольника. Занятая высота полосы определяется как $H = \max\limits_{(i = \overline{1, n})} (y_i + h_i)$.

Пусть $I = \{P_1, ..., P_n\}$ — множество всех прямоугольников задачи T:$\langle W, n, w, h\rangle$. Рассмотрим разбиение I на два подмножества I^+ и I^-. Пусть T^*:$\langle W, n^*, w^*, h^*\rangle$ — задача, которая получается из задачи T удалением прямоугольников из I^-. Тогда допустимая прямоугольная упаковка для задачи T^* является прямоугольной частичной упаковкой (*partial rectangular packing*, PRP) для задачи T. Прямоугольники из множества I^* в дальнейшем будут называться размещенными, а из I^- — свободными.



а) *б*)

Рис. 1. Частичные упаковки:
а — до вставки прямоугольника; *б* — после вставки прямоугольного предмета (черные точки соответствуют угловым точкам, а штриховая линия — лестничному контуру)



а) *б*)

Рис. 2. Пример для лестничной частичной упаковки PRP из пяти прямоугольников в полосу ширины $W = 10$:
а — область под лестничным контуром PRP; *б* — дополнение области под лестничным контуром

Рассмотрим некоторую допустимую прямоугольную упаковку RP. Если в упаковке существует прямоугольник, который можно сдвинуть влево или вниз, то сдвинем его в выбранном направлении (если сдвиг возможен в обоих направлениях, то одно из них выбирается случайно). В результате таких операций будет получена прямоугольная упаковка RP_{BL}, в которой не существует прямоугольников, сдвиг которых влево или вниз возможен. Такая прямоугольная упаковка называется удовлетворяющей BL условию [2]. При преобразовании RP в RP_{BL} занятая высота полосы может только уменьшиться. Следовательно, среди допустимых прямоугольных упаковок, удовлетворяющих BL условию, обязательно будет оптимальная.

Определение 1. *Угловыми точками* для прямоугольной частичной упаковки PRP называются точки последовательности $\{(x_1^c, y_1^c), ..., (x_k^c, y_k^c)\}$, состоящей из всех таких точек, что:

- для каждой точки (x_j^c, y_j^c) последовательности найдется такой прямоугольник $i \in I^+$, что $x_j^c = x_i + w_i$ и $y_j^c < y_i + h_i$, или $x_j^c = 0$;
- для каждой точки (x_j^c, y_j^c) последовательности найдется такой прямоугольник $i \in I^+$, что $y_j^c = y_i + h_i$ и $x_j^c < x_i + w_i$, или $y_j^c = 0$;
- все точки (x, y) такие, что $x > x_j^c$ и $y > y_j^c$, не принадлежат ни одному прямоугольнику в PRP;
- каждая последующая точка в последовательности находится правее и ниже предыдущей, т. е. $x_{j+1}^c > x_j^c$ и $y_{j+1}^c < y_j^c$.

Угловые точки задают "лестничный" контур $[(x_1^c, y_1^c) - (x_2^c, y_1^c) - (x_2^c, y_2^c) - ... - (x_k^c, y_k^c)]$, который состоит только из вертикальных и горизонтальных отрезков (рис. 1). Если прямоугольники добавлять в прямоугольную частичную упаковку PRP так, чтобы их нижние левые вершины совпадали с угловыми точками, то в результате можно получить любую допустимую прямоугольную упаковку RP, удовлетворяющую BL условию [2]. Частичные упаковки, полученные таким способом, называются *лестничными*. Впервые лестничные упаковки были введены под названием "метод зон" [3].

Для описания нового свойства лестничных частичных упаковок понадобятся следующие определения (рис. 2).

Определение 2. *Областью под лестничным контуром* прямоугольной частичной упаковки PRP называется многоугольник $S(PRP)$, заданный верши-

нами (0, 0), (x_1^c, y_1^c), (x_2^c, y_1^c), (x_2^c, y_2^c), ..., (x_k^c, y_k^c) в порядке обхода по часовой стрелке.

Определение 3. *Дополнением S(PRP) по высоте H* называется многоугольник *coS*(*PRP*, *H*), заданный вершинами (0, 0), (0, *H*), $(W - x_k^c, H - y_k^c)$, $(W - x_k^c, H - y_{k-1}^c)$, ..., $(W - x_1^c, H - y_1^c)$ в порядке обхода по часовой стрелке.

Кроме этих определений введем следующее вспомогательное определение.

Определение 4. Лестничная прямоугольная частичная упаковка PRP_A в полосу ширины *W* и с занятой высотой полосы *H* центрально симметрична лестничной прямоугольной упаковке PRP_B из того же набора прямоугольников, если координаты прямоугольников в PRP_B получаются из координат соответствующих прямоугольников в PRP_A так: $x_i' = W - x_i - w_i$, $y_i' = H - y_i - h_i$ и, возможно, последующим сдвигом вниз и влево.

Тогда свойство лестничных частичных упаковок, на котором основывается предлагаемый алгоритм, можно сформулировать в виде следующей теоремы.

Теорема 1. Любую допустимую прямоугольную упаковку RP в полосу ширины *W* и с занятой высотой полосы *H* можно представить в виде двух лестничных частичных упаковок PRP_B и PRP_T, состоящих из попарно различных прямоугольников так, что:

- PRP_B состоит из $\lceil n/2 \rceil$ прямоугольников, а PRP_T — из $\lfloor n/2 \rfloor$;
- координаты прямоугольников в PRP_B такие же, как и у соответствующих прямоугольников в RP;
- координаты прямоугольников в PRP_T такие же, как и у соответствующих прямоугольников в упаковке центрально симметричной RP;
- $S(PRP_T) \in coS(PRP_B, H)$.

Основная схема точного алгоритма

Перед началом работы точного алгоритма находятся значения нижней LB и верхней UB границ целевой функции. Запуск точного алгоритма необходим, только если LB < UB.

В процессе работы алгоритма обходится дерево поиска, которое однозначно задается операцией ветвления BRANCH. Такое дерево поиска в дальнейшем будем обозначать как T(BRANCH). Функция отсечения неперспективных ветвей BOUNDED и способ обхода дерева поиска определяет множество вершин дерева поиска T(BRANCH), которые будут посещены. В случае предлагаемого алгоритма на BRANCH и BOUNDED накладывается ряд ограничений.

Определение 5. Операция ветвления BRANCH называется лестничной, если:

- все вершины в T(BRANCH) соответствуют лестничным частичным упаковкам, среди которых существует хотя бы одна оптимальная;
- корень дерева соответствует пустой полосе;
- для любой не корневой вершины A число прямоугольников в соответствующей частичной лестничной упаковке ровно на единицу превосходит число прямоугольников в упаковке, соответствующей предку A.

Очевидно, что высота дерева поиска, порожденного лестничной операцией ветвления, составляет *n*. В дальнейшем речь пойдет только о лестничных операциях ветвления. Поэтому слово лестничная перед BRANCH будет опускаться.

Определение 6. Пусть RP_A — оптимальная лестничная упаковка, которая соответствует некоторой вершине в T(BRANCH), а RP_B — упаковка, центрально-симметричная RP_A. Операция ветвления BRANCH называется четной, если для каждой RP_A существует вершина в T(BRANCH), соответствующая такой лестничной упаковке RP_C, что $S(RP_C) = S(RP_B)$.

Таким образом, четная операция ветвления BRANCH не должна исключать центрально-симметричные упаковки при ветвлении. Примером не четной операции ветвления BRANCH может послужить операция, которая при ветвлении проверяет, что первый прямоугольник находится левее второго.

Определение 7. Пусть RP_A — оптимальная лестничная упаковка, которая соответствует некоторой вершине в T(BRANCH), а PRP_B — некоторая лестничная частичная упаковка, из которой может быть получена RP_A путем размещения свободных прямоугольников в угловых точках. Операция ветвления BRANCH называется полной, если для каждой PRP_B существует вершина в T(BRANCH), соответствующая такой лестничной частичной упаковке PRP_C, что PRP_C и PRP_B состоят из одного и того же набора прямоугольников и $S(PRP_C) = S(PRP_B)$.

Таким образом, полной операции ветвления BRANCH разрешается исключать идентичные с точностью до лестничного контура упаковки при ветвлении. Простейшим примером таких упаковок может послужить упаковка двух прямоугольников с одинаковой шириной и высотой — вне зависимости от того, размещен первый прямоугольник до или после второго, результирующая упаковка будет иметь один и тот же лестничный контур. Однако если из двух лестничных частичных упаковок с различным лестничным контуром можно получить оптимальную упаковку (которая имеет соответствующую вершину в T(BRANCH)), то тогда ни одна из частичных упаковок не должна быть исключена при ветвлении. Заметим, что полная операция ветвления BRANCH не гарантирует,

что в T(BRANCH) будут содержаться вершины, соответствующие всем возможным лестничным оптимальным упаковкам.

Примером четной полной операции ветвления может послужить простейшая операция ветвления BRANCH-NAIVE, которая порождает все возможные лестничные частичные упаковки в полосу заданной ширины. Дочерние вершины для некоторой вершины, соответствующей лестничной частичной упаковке PRP, получаются размещением всевозможных свободных прямоугольников в каждой из угловых точек PRP согласно BRANCH-NAIVE.

Более того, операции ветвления, которые используются в известных точных алгоритмах [2, 4], также являются четными полными лестничными операциями ветвления.

Определение 8. Функция отсечения неперспективных ветвей BOUNDED называется полной, если ни одна из вершин $V \in T(BRANCH)$ таких, что существует путь от V до некоторой вершины, соответствующей лестничной упаковке с высотой занятой части полосы меньшей, чем UB, не будет отсечена.

В качестве примера полной функции отсечения неперспективных ветвей можно предложить любую функцию, которая основывается на вычислении локальных нижних границ.

Наконец, опишем точный алгоритм "встретимся посередине" (meet-in-the-middle, MiM), который использует четную полную операцию ветвления BRANCH и полную функцию отсечения неперспективных ветвей BOUNDED. Обход дерева поиска T(BRANCH) алгоритмом MiM осуществляется "в ширину", начиная с корневой вершины. T(BRANCH) обходится только до глубины не более $\lceil n/2 \rceil$ от корня. Вершины, соответствующие лестничным частичным упаковкам из одного и того же набора прямоугольников I^+, объединяются в одну группу $g(I^+)$. Пусть вершины на расстоянии $\lfloor n/2 \rfloor$ от корня образуют множество B, а на расстоянии $\lceil n/2 \rceil$ — множество T. После завершения обхода T(BRANCH) и формирования множеств B и T алгоритм MiM переходит к заключительному шагу. Каждая лестничная частичная упаковка PRP_B, соответствующая некоторой вершине из $g(I^+) \in B$, комбинируется со всеми такими лестничными частичными упаковками PRP_T, которые соответствуют вершинам из $g(I \backslash I^*) \in T$. Под комбинацией подразумевается нахождение такой минимальной величины $H_{\min}$, что из PRP_B можно получить допустимую упаковку RP размещением прямоугольников из PRP_T со следующим преобразованием координат:

$$x_i' = W - x_i - w_i,\ y_i' = H - y_i - h_i.$$

При нахождении $H_{\min}$ нужно учитывать, что прямоугольники из PRP_T могут пересечься с прямоугольниками из PRP_B и выйти за границы полосы. Если $H_{\min} < UB$, то это означает, что было найдено лучшее решение, и верхняя граница заменяется на $H_{\min}$. Впервые похожая идея была предложена для задачи SUBSET-SUM [5].

Корректность предлагаемого алгоритма MiM подтверждается следующей теоремой.

Теорема 2. Алгоритм MiM находит оптимальное решение, если используются полная четная операция ветвления BRANCH и полная функция отсечения неперспективных ветвей BOUDED.

Область применения алгоритма MiM определяется следующей теоремой.

Теорема 3. Пусть A^* — произвольный точный алгоритм для решения задачи **2SPP**, использующий метод ветвей и границ с полной четной операцией ветвления BRANCH и полной функцией отсечения неперспективных ветвей BOUDED. Тогда число вершин дерева поиска T(BRANCH), посещенных алгоритмом A^*, будет не меньше чем число вершин, посещенных алгоритмом MiM, если $LB < Opt = UB$ (где Opt — минимально достижимое значение функции цели).

В результате, использование алгоритма MiM для случая $LB < Opt = UB$ будет эффективнее использования любого из алгоритмов на базе метода ветвей и границ, удовлетворяющих условиям теоремы 3. К таким алгоритмам относятся все известные автору алгоритмы на базе геометрического подхода (например, [2, 3, 4]). Основной трудностью является то, что перед началом работы точного алгоритма нельзя сказать, совпадает полученная верхняя граница с оптимумом или нет.

Численный эксперимент

Исходными данными являются задачи F. Clautiaux [6]. Эти задачи генерируются случайно по двум параметрам: числу прямоугольников и отклонению суммарной площади прямоугольников от площади занятой части полосы. Ширина полосы для всех задач составляет $W = 20$. В таблице задача обозначается D, X, N, где N — это число прямоугольников, D — это отклонение суммарной площади прямоугольников от площади занятой части полосы в процентах. Если оптимум для задачи не превышает 20, то X равно F, иначе N.

Для эксперимента были отобраны задачи с $N < 17$ из-за специфики предлагаемого алгоритма. Дело в том, что целью разработки алгоритма ставилась минимизация времени работы в худшем случае. В среднем (и даже для "простых" задач) алгоритм работает не намного быстрее, чем в худшем случае. Для $N > 17$ это время составляет более 5 мин. Поэтому было принято решение рассматривать только задачи с $N < 17$.

Результаты численного эксперимента для точного алгоритма MiM на задачах F.Clautiaux

Задача	LB_C	UB	Вершины	Кластеры	Комбинации	Время
00, N, 10	20	22	2082	507	528	0.3
00, N, 15	20	21	208309	14684	6571	2.7
03, N, 10	20	21	1151	359	98	0.01
03, N, 15	20	21	1266432	19513	4030872	46.5
03, N, 16	20	21	1670845	36030	10253122	52.1
04, F, 15	20	21	2897636	57032	35987267	96.5
04, N, 15	20	22	909047	21529	6645269	32.3
05, F, 15	19	20	289452	16624	121599	4.8
05, N, 15	19	21	494281	21574	1914092	13.5
05, X, 15	20	21	2143709	46075	23598726	187.8
07, F, 15	20	20	0	0	0	0
07, N, 10	19	22	4109	617	13708	0.06
07, N, 15	19	21	568805	17160	743037	19.4
07, X, 15	19	21	3564347	73343	6367347	231.3
08, F, 15	19	21	1225474	19036	2740212	39.2
08, N, 15	19	21	2384147	19188	30309026	215.1
10, N, 10	18	21	2952	464	1128	0.04
10, N, 15	18	22	381949	19957	1803654	11.9
10, X, 15	18	21	1744021	19685	26683894	109.6
13, N, 10	18	21	4889	607	16440	0.06
13, N, 15	18	22	2174100	22536	31105115	213.1
13, X, 15	19	22	2554458	49346	45457905	297.5
15, N, 10	17	21	7699	609	51122	0.1
15, N, 15	18	22	663303	17599	1180505	25.9
20, F, 15	16	17	383748	12898	51588	6.2

В качестве глобальной нижней границы для алгоритма MiM использовалась непрерывная нижняя граница LB_C, так как при использовании более качественных нижних границ их числовое значение совпадало с верхней границей, и запуск алгоритма MiM не требовался. В качестве верхней границы UB использовался эволюционный алгоритм (1 + 1)-EA с жадной модификацией декодера FirstFit [7]. На подсчет верхней границы выделялось 5 с, и это время не учитывалось при подсчете времени работы алгоритма MiM. На работу точного метода выделялось до 6 мин. Кроме времени работы также измерялось:

- число посещенных вершин дерева поиска;
- число кластеров, на которые были разбиты посещенные вершины;
- число прямоугольных упаковок, полученных путем комбинации частичных лестничных упаковок.

Алгоритм MiM справился со всеми предложенными задачами. Это доказывает эффективность алгоритма.

Заключение

Предложенный новый точный алгоритм решения задачи двухмерной упаковки в полубесконечную полосу позволяет уменьшить время, необходимое для доказательства оптимальности ранее найденного решения, по сравнению с другими точными методами на основе геометрического подхода.

Список литературы

1. **Garey M., Johnson D.** Computer and Intractability: A Guide to the Theory of NP-Completeness. — New-York: W. H. Freeman and Company, 1979.
2. **Martello S., Monaci M., Vigo D.** An exact approach to the strip-packing problem // INFORMS Journal on Computing. 2003. Vol. 15. N 3. P. 310—319.
3. **Липовецкий А. И.** Свойства прямоугольных укладок и алгоритмы прямоугольного раскроя. Препринт УрО АН СССР. Свердловск, 1988. 50 с.
4. **Kenmochi M., Imamichi T., Nonobe K., Yagiura M., Nagamochi H.** Exact algorithms for the two-dimensional strip packing problem with and without rotations // European Journal of Operational Research. 2009. Vol. 198. N 1. P. 73—83.
5. **Horowitz E., Sahni S.** Computing Partitions with Applications to the Knapsack Problem // Journal of the ACM. 1974. Vol. 21. N 2. P. 277—292.
6. **Clautiaux F., Jouglet A., Carlier J., Moukrim A.** A new constraint programming approach for the orthogonal packing problem // Computers and Operations Research. 2008. Vol. 35. P. 944—959.
7. **Назаров Д. А.** Эффективная реализация простейших алгоритмов конструирования прямоугольных упаковок // Принятие решений в условиях неопределенности. Межвузовский науч. сб. Уфа: УГАТУ, 2009. С. 186—192.

ИНФОРМАЦИЯ

9th IEEE EAST-WEST DESIGN & TEST SYMPOSIUM (EWDTS 2011)

Севастополь, Украина, 09—12 сентября 2011

Цель симпозиума **IEEE East-West Design & Test Symposium (EWDTS)** — расширение международного сотрудничества и обмен опытом между ведущими учеными Западной и Восточной Европы, Северной Америки и других стран в области автоматизации проектирования, тестирования и верификации электронных компонентов и систем. Симпозиум проводится, как правило, в странах бассейнов Черного и Балтийского морей, Центральной Азии. Оргкомитет приглашает ученых, аспирантов и студентов принять участие в работе международного симпозиума EWDTS'11.

Адрес оргкомитета: Проф. Владимир Хаханов, кафедра Автоматизации проектирования вычислительной техники Харьковского национального университета радиоэлектроники, пр. Ленина 14, Харьков, 61166, Украина.

Тел.: +380-57-702-13-26, E-mail: hahanov@kture.kharkov.ua, www.ewdtest.com/conf/

УДК 50.51.17

Д. И. Черемисинов,
канд. техн. наук, вед. науч. сотр.,
e-mail: cher@newman.bas-net.by,
Л. Д. Черемисинова,
д-р техн. наук, гл. науч. сотр.,
e-mail: cld@newman.bas-net.by,
Объединенный институт проблем информатики
Национальной академии наук Беларуси, г. Минск

Минимизация двухуровневых КМОП-схем с учетом энергопотребления

Предлагается комплекс методов и программ минимизации булевых функций в классе дизъюнктивных нормальных форм (ДНФ), позволяющих минимизировать площадь и среднее значение рассеиваемой мощности двухуровневых логических И—ИЛИ-схем, реализуемых на основе КМОП-технологии. Методы представляют собой модификации известных методов минимизации булевых функций путем добавления в них эвристик, направляющих процесс минимизации к получению систем ДНФ, реализуемых КМОП-схемами с меньшим энергопотреблением. Приводятся результаты вычислительных экспериментов, позволяющие оценить эффект учета энергопотребления КМОП-схем при минимизации реализуемых систем ДНФ.

Ключевые слова: *логическое проектирование СБИС, минимизация булевых функций, энергопотребление, КМОП-технология*

Введение

Проектированию схем с малым энергопотреблением придается в последние годы все большее значение в связи с тем, что рассеивание энергии становится камнем преткновения при дальнейшем повышении уровня интеграции, а также в связи с расширением рынка портативных устройств с автономным питанием, для которых очень важно увеличить время их автономной работы без подзарядки. От потребляемой микросхемой мощности зависят такие важные параметры портативных устройств, как мощность источника питания, размер шин питания, требования к системе охлаждения, время разрядки аккумуляторной батареи, а следовательно, и время работы мобильных систем без перезарядки батареи.

Переход на технологии сверхбольших (и ультрабольших) интегральных схем породил новые проблемы при их производстве и использовании. Высокая степень интеграции привела к появлению устройств, более чувствительных к рассеиванию мощности, к обострению проблемы надежности по сравнению с аналогичными устройствами меньшей степени интеграции. Увеличение плотности монтажа и частоты функционирования обусловливает чрезмерное тепловыделение при функционировании устройств на базе СБИС, что приводит к перегреву и, как следствие, к возрастанию вероятности сбоев при их работе, снижению надежности и сокращению срока службы. Снижение энергопотребления схем перестает быть задачей, имеющей второстепенное значение и решаемой только после сокращения площади и задержки микросхемы.

Снижение энергопотребления проектируемой схемы может обеспечиваться на разных уровнях проектирования. Характеристики (сложность, энергопотребление) спроектированной схемы существенно зависят от характеристик проектных решений на каждом из уровней проектирования, так как на каждом этапе преобразованиям подвергается схема, полученная на предшествующих этапах, и просчеты в качестве схемы, допущенные на ранних уровнях, не могут, в общем случае, далее компенсироваться в полной мере. Причем, чем более ранним является этап, тем более важно получать на нем более качественное решение. В частности, синтез логической схемы традиционно начинается с оптимизации двухуровневых И—ИЛИ-схем, обеспечиваемой путем минимизации систем булевых функций в классе дизъюнктивных нормальных форм (ДНФ) [1]. Сложность и оценка энергопотребления синтезированной многоуровневой логической схемы в технологическом базисе существенным образом определяются полученной в результате минимизации двухуровневой И—ИЛИ-схемой. Оптимизация этих характеристик на последующих за минимизацией этапах ограничена функциональностью двухуровневой И—ИЛИ-схемы, а существенное их улучшение требует иногда и получения другого варианта схемы.

В настоящей работе рассматривается задача энергосберегающего синтеза двухуровневых статических логических КМОП-схем [2, 3]. Исследуется случай синхронной реализации схем, когда основные схемотехнические решения проекти-

руемой схемы, такие как частота синхронизации и напряжение питания, фиксированы. Для оценки энергопотребления в процессе оптимизации двухуровневых схем используется статический метод (в противовес динамическому методу, в основе которого лежит моделирование), основанный на вероятностных характеристиках входных сигналов и функционально-структурных свойствах исследуемой схемы.

Энергопотребление схем в общем случае прямо пропорционально площади, занимаемой схемой на кристалле; это означает, что основной путь энергосбережения при проектировании логических схем прежде всего подразумевает сокращение площади, занимаемой схемой на кристалле. Однако применение традиционных методов минимизации площади в качестве канвы для методов минимизации энергопотребления требует рассмотрения также новых критериев, используемых наряду с критериями минимизации площади схем.

Оценка мощности, потребляемой КМОП-схемой

Известно, что компоненты СБИС, выполненные по КМОП-технологии, потребляют подавляющую часть необходимой для их функционирования энергии во время переключения [2—5]. Отсюда следует, что более активные, в переключательном плане, КМОП-схемы потребляют большую энергию. Таким образом, энергопотребление существенно зависит от переключательной активности элементов схемы, а она, в свою очередь, определяется последовательностью подаваемых входных воздействий на КМОП-схемы, т. е. динамикой функционирования.

В КМОП-технологии около 80 % всей рассеиваемой микросхемой энергии приходится на динамическую составляющую [2—5], порождаемую нестационарным поведением узлов микросхемы. Эта энергия рассеивается только во время переходных процессов, когда сигналы на выходах узлов микросхемы переключаются. Согласно упрощенной модели КМОП-микросхема рассеивает энергию всякий раз, когда изменяется сигнал на ее выходе. Среднее значение мощности, рассеиваемой на выходе синхронной микросхемы, выражается известным соотношением [2—4]

$$P_{dyn} = \frac{1}{2} V_{dd}^2 f_{clk} E_s C_L,$$

где V_{dd} — напряжение питания; f_{clk} — частота синхронизации; C_L — емкостная нагрузка схемы; E_s — переключательная активность выхода схемы, определяемая как математическое ожидание числа логических переходов сигнала (из 1 в 0 или из 0 в 1) за один период синхронизации.

Значения параметров V_{dd} и f_{clk} определяются при архитектурном проектировании, а на логическом уровне минимизация динамической мощности сводится к минимизации переключаемой емкости $E_i C_i$. На этом этапе рассеивание энергии микросхемой в целом можно оценить суммой мощностей, потребляемых всеми ее узлами [6]:

$$P_s = \sum_{i=1}^{n} E_i C_i, \qquad (1)$$

где n — число узлов в схеме; C_i — емкостная нагрузка и E_i — переключательная активность i-го узла схемы.

Чтобы получить схему с минимумом энергопотребления, необходимо оптимизировать значение этого параметра на всех стадиях логического синтеза. Важнейшим из этих этапов является начальный этап — оптимизация двухуровневого представления или минимизация реализуемой системы булевых функций. Известно множество алгоритмов минимизации двухуровневых представлений по критерию площади, выражаемой в этом случае числом простых импликант или литералов кратчайшей системы ДНФ.

Для учета энергопотребления при минимизации в классе ДНФ было введено расширение понятия простой импликанты [4] — энергосберегающей импликанты (в общем случае не простой), которая может содержать избыточные литералы, приводящие к уменьшению переключательной активности этой импликанты. Однако может оказаться, что избыточные литералы, внося вклад в потребляемую схемой мощность, могут привести не к уменьшению, а к увеличению энергопотребления схемы, так как с учетом формулы (1) энергопотребление подсхемы, реализующей импликанту k_r, оценивается как

$$P_{rs} = C_{\wedge} E(k_r) + \sum_{i=1}^{n(k_r)} C_{xi} E(x_i), \qquad (2)$$

где $C_{\wedge}$ и C_{xi} — емкостные нагрузки элемента И и входного полюса x_i; $E(k_r)$ и $E(x_i)$ — переключательные активности выхода элемента И и i-го входного полюса; $n(k_r)$ — число литералов импликанты k_r. Кроме того, построение и учет энергосберегающих импликант приводит к полному пересмотру и усложнению процедур минимизации булевых функций, а их вхождение в решение затрудняет тестирование спроектированного устройства.

Площадь и энергопотребление схемы в процессе синтеза тесно связаны в том смысле, что уменьшение площади схемы имеет тенденцию обеспечивать также и снижение ее энергопотребления, а увеличение площади, наоборот, как правило, ведет также к увеличению энергопотребления. Из этих соображений следует, что в процессе минимизации необходим компромисс между критериями

минимизации энергопотребления и площади. Основная проблема оптимизации технологически независимой схемы заключается в том, что на этапе логического синтеза, еще не привязанного к конкретному технологическому базису, трудно оценить энергопотребление реальной схемы на достаточно достоверном уровне. В связи с этим отступать от оптимизации сложности схемы в угоду оптимизации по вводимым оценкам энергопотребления нецелесообразно.

Предлагаемое в настоящей работе решение проблемы энергосберегающей оптимизации двухуровневых схем исходит из того, что целесообразной кажется минимизация, в процессе которой при оценке безызбыточных ДНФ на предмет принятия их в качестве решения используется ранжированный критерий: в первую очередь площадь, затем энергопотребление. При этом ищется несколько решений с минимумом площади, из которых выбирается решение с минимумом энергопотребления.

Реализованные в описываемом программном комплексе методы представляют собой модификации известных методов минимизации булевых функций в классе ДНФ путем добавления в них эвристик, направляющих процесс минимизации к получению систем ДНФ, реализуемых схемами с наименьшим энергопотреблением. За основной критерий минимизации принимается число литералов результирующей системы ДНФ, за второй — энергопотребление, оцениваемое суммарной переключательной активностью первого уровня двухуровневой схемы:

$$P_{rs} = \sum_{r=1}^{m} (E(k_r) + \sum_{i=1}^{n(k_r)} E(x_i)), \qquad (3)$$

где m — число импликант минимизированной формы системы булевых функций. Эта оценка отличается от оценки (2) тем, что не учитывается влияние емкостных нагрузок $C_{\wedge}$ и $C_{\text{вх}}$, существенно зависящих от технологического базиса проектирования и неизвестных на стадии оптимизации функциональных описаний (их значения полагаются константами).

Оценки переключательной активности сигналов

Моделирование, широко используемое в практике проектирования для установления разных параметров (в том числе и энергопотребления) схем, к сожалению, не годится для использования в процессе проектирования схемы для оценки вариантов синтеза. Более подходящими являются статические методы оценки энергопотребления схем, в основе которых лежит вероятностный подход к определению переключательной активности сигналов. Этот метод основан на: 1) задании вероятностных характеристик входных сигналов, отражающих частоту смены значений сигналов на входе; 2) распространении вероятностной информации о смене значений сигналов через всю схему, от входов к выходам.

Используемые на практике оценки переключательной активности узлов схемы выведены в предположении их пространственной (отсутствует корреляция сигналов на разных полюсах) и временной независимости (отсутствует корреляция значений одного и того же сигнала в разных тактах). Реально используемые оценки переключательной активности схемы выведены в предположениях нулевой и ненулевой задержек сигнала узлами схемы. Первые модели оценки предполагают, что все изменения на входах схемы распространяются через все ее элементы мгновенно и, значит, одновременно. Вторые модели оценок предполагают, что элементы схемы имеют конечные (но не нулевые) задержки, и принимают во внимание паразитные переключения, не предусмотренные функцией, реализуемой узлом.

В предположении нулевой задержки сигналов введены понятия [2—4] вероятности p_i^1 (p_i^0) появления сигнала 1(0) на некотором i-м полюсе и вероятности E_i смены сигнала на полюсе. Первая вероятность называется сигнальной (и обозначается далее p_i) и определяется средней долей тактов, на которых сигнал на i-м полюсе имеет значение 1. Вторая вероятность называется переключательной активностью и определяется средней долей тактов, на которых сигнал на i-м полюсе меняет свое значение по сравнению со значением в предшествующем такте. Переключательная активность i-го полюса схемы в предположении, что p_i^1 , $p_i^0 < 1$, равна

$$E_i = 2p_i^1\, p_i^0 = 2p_i(1 - p_i). \qquad (4)$$

Вероятность p_e появления сигнала 1 на выходе элемента y зависит от функции, реализуемой этим элементом. В частности для элемента e, реализующего функцию И, имеет место

$$p_y^{\wedge} = \prod_{i=1}^{n(e)} p_i. \qquad (5)$$

В предположении ненулевой задержки сигналов введены понятия равновесной вероятности и плотности переключений [7], которые определяют поведение сигнала во времени, а не только в устойчивом состоянии в течение такта. Равновесная вероятность P_x логического сигнала $x(t)$ определяется как средняя доля времени, в течение которого сигнал имеет значение 1. Плотность переключений A_x определяется как среднее число переключений сигнала $x(t)$ в единицу времени. В отличие от сигнальной вероятности и переключательной активности равновесная вероятность и плотность переключений позволяют учесть внутренние задержки схемы.

Показано [7], что если входы x_i элемента пространственно независимы, то плотность переключений сигнала на его выходе y вычисляется как

$$A_y = \sum_{i=1}^{n} P\left(\frac{dy}{dx_i}\right) A_{xi}, \quad (6)$$

где булева разность функции y относительно переменной x определяется как

$$\frac{dy}{dx} = y(x = 1) \oplus y(x = 0). \quad (7)$$

Например, для элемента, реализующего функцию И $y = x_1x_2...x_n$, в соответствии с формулами (5)—(7) имеет место

$$\frac{dy}{dx_i} = x_1 \ldots x_{i-1}x_{i+1} \ldots x_n;$$

$$A_y = \sum_{i=1}^{n} P(x_1)...P(x_{i-1})P(x_{i+1})...P(x_n)A_{xi}. \quad (8)$$

Состав комплекса программ минимизации

Практически все методы минимизации двухуровневых представлений булевых функций делят множество искомых импликант на три подмножества: существенных, несущественных и условно существенных. Первые должны быть включены в любое безызбыточное решение, вторые не должны включаться ни в одно решение, а из третьих выбирается некоторое безызбыточное подмножество, покрывающее все интервалы единичных областей задания минимизируемых функций, не покрытые существенными простыми импликантами. Методы минимизации различаются способами построения простых импликант из числа условно существенных и критериями, которым они должны удовлетворять для включения их в решение.

Различают методы:

- последовательного построения простых импликант, включаемых в решение (например, метод конкурирующих интервалов [8]), путем укрупнения интервала булева пространства аргументов, представляющего импликанту, за счет включения в него непокрытых элементарных конъюнкций;
- последовательного расширения интервала булева пространства аргументов, представляющего сначала конъюнкцию исходной ДНФ, а в перспективе — простую импликанту, покрывающую эту конъюнкцию, за счет исключения некоторых литералов, входящих в нее (например, методы, реализуемые ESPRESSO [1]). Простейший метод минимизации может использовать только операцию расширения условно существенных импликант.

Наиболее просто, в целях учета энергосбережения, модифицируются методы минимизации, в которых в качестве кандидатов в искомое решение строится сразу несколько простых импликант, или методы, в которых найденное решение модифицируется методом последовательных улучшений.

Разработанный программный комплекс включает в себя программы, реализующие приближенные методы минимизации и реализованные на языке программирования C++ для Windows SP. Результатом их работы является, в общем случае, безызбыточная ДНФ или их система. Точные методы не были включены в комплекс, так как для булевых функций практической размерности получение минимальных ДНФ за приемлемое время проблематично. Комплекс включает в себя программы, позволяющие варьировать:

- объект оптимизации: минимизация одной полностью или частично определенной булевой функции; минимизация системы полностью или частично определенных булевых функций;
- метод решения: метод конкурирующих интервалов [8]; модифицированный метод ESPRESSO [1]; итеративный метод минимизации; быстродействующий однопроходный метод минимизации;
- способ учета совместности минимизации системы ДНФ: раздельная минимизация функций системы, совместная минимизация функций системы;
- критерий оптимизации: сложность получаемых ДНФ с учетом и без учета энергопотребления;
- учет дополнительных настроек: парафазная реализация ДНФ [9]; заданный лимит времени минимизации; число циклов оптимизации.

Исходными данными для алгоритмов оптимизации являются:

- функциональное описание одной булевой функции или системы;
- сигнальные вероятности входных сигналов;
- метод решения;
- точность решения;
- дополнительные настройки.

Все упомянутые выше методы настраиваются на минимизацию как полностью, так и частично определенных булевых функций.

Модификация методов минимизации

Чтобы направить минимизацию к получению энергосберегающего решения, необходимо априори вычислить переключательные активности для всех входных сигналов, а в процессе получения простых импликант и безызбыточных покрытий вычислять и учитывать переключательные активности всех простых импликант, используя формулы (4), (5) или (5), (6) (в зависимости от принятой

модели задержек сигналов). Для пояснения процедур, направляющих минимизацию на получение нужного результата, рассмотрим операции, входящие практически во все методы минимизации.

Операция расширения интервала выполняется путем исключения литералов (переменных и их инверсий). При расширении интервала принимаются во внимание две цели: уменьшить сложность этого интервала, т. е. максимально расширить его (путем исключения литералов), и покрыть с его помощью (полностью или частично) как можно больше других интервалов. При минимизации энергопотребления желательно: 1) исключать не любые, а наиболее переключаемые литералы и 2) покрывать не любые интервалы, а соответствующие наиболее энергоемким конъюнкциям. В силу первого утверждения на предмет исключения литералы рассматриваются в порядке убывания переключательной активности, т. е. сначала делается попытка исключить литералы с большей переключательной активностью.

В силу второго утверждения важен порядок расширения интервалов: слишком раннее расширение интервала может препятствовать ситуации, когда некий другой интервал покроет этот рассматриваемый. Чтобы снизить энергопотребление, оценивается энергетический вклад каждого выбрасываемого интервала, и выбрасываются в первую очередь энергоемкие интервалы. Для этого энергоемкие интервалы берутся для расширения последними в надежде, что некоторые другие интервалы расширятся и покроют их. При этом перед выполнением этапа расширения интервалы упорядочиваются в порядке возрастания энергопотребления.

Операция нахождения безызбыточного покрытия заключается в приведении текущего покрытия ДНФ к безызбыточному виду. При поиске безызбыточного множества простых импликант выбирается минимальное число наименее активных (наименее энергоемких) импликант. При этом из безызбыточных множеств простых импликант, представляющих собой покрытие исходного множества интервалов единичных областей минимизируемых булевых функций, отбираются минимальные по мощности или по суммарному числу литералов всех импликант. Каждое из отобранных множеств оценивается суммой переключательных активностей (или плотностей переключений) входящих в него импликант. В качестве решения выбирается безызбыточное множество с минимальной оценкой переключения.

Операция сокращения интервала (такая, как REDUCE в методе ESPRESSO-MV [1]) выполняется путем добавления в него литералов. При сокращении интервала (с учетом минимизации энергопотребления) желательно включать в него не любые литералы из числа возможных, а наименее переключаемые литералы.

Некоторые оценки эффективности учета энергопотребления при минимизации

Для выявления эффекта, получаемого от применения методов энергосберегающей минимизации, были проведены экспериментальные исследования программных реализаций методов минимизации без учета и с учетом энергопотребления. Исследования проводились на одних и тех же тестовых примерах из набора [10]. Ниже приведены результаты исследования итеративного метода минимизации систем булевых функций без учета (параметры минимизации с индексом 1) и с учетом (параметры минимизации с индексом 2) энергопотребления (табл. 1 и 2).

В табл. 1 приведены оценки энергопотребления первого уровня двухуровневой И—ИЛИ-схемы, на котором реализуются конъюнкции минимизированных систем ДНФ. Жирным шрифтом выделены варианты минимизации, выигравшие по энергопотреблению. Здесь приняты следующие обозначения: n — число переменных исходной системы ДНФ; m — число функций исходной системы ДНФ; k — число конъюнкций исходной системы ДНФ; $k^1_{\min}$, $k^2_{\min}$ — число конъюнкций минимизированных систем ДНФ; l^1, l^2 — число литералов

Таблица 1

Результаты минимизации систем булевых функций

Пример	Исходная система ДНФ			Минимизированная без учета энергопотребления				Минимизированная с учетом энергопотребления			
	n	m	k	$k^1_{\min}$	l^1	P^1_s	t^1	$k^2_{\min}$	l^2	P^2_s	t^2
b12	15	9	431	45	286	61,8653	0,06	44	267	**61,8647**	0,06
in0	15	11	138	111	1348	**385,478**	0,01	111	1351	385,894	0,01
life	9	1	512	84	756	233,603	0,01	84	756	**230,48**	0,01
mlp4	8	8	256	160	1574	355,837	0,03	160	1577	**355,488**	0,03
root	8	5	256	57	430	**93,8343**	0,01	58	393	95,2714	0,01
tms	8	16	30	30	484	69,3707	0,00	30	484	**69,3705**	0,00
z9sym	9	1	420	90	630	**178,538**	0,01	101	707	197,323	0,04
ADDM4	9	8	512	249	2477	613,69	0,09	249	2485	**612,889**	0,09

в системе уравнений, описывающих минимизированные системы ДНФ; P_s^1, P_s^2 — оценки энергопотребления подсхем, реализующих конъюнкции минимизированных систем ДНФ, подсчитанные по формулам (3), (5), (6); t^1, t^2 — время вычислений в секундах на персональном компьютере с процессором Pentium 4 (3 ГГц).

Минимизация систем функций с учетом энергопотребления проводилась при следующих значениях сигнальных вероятностей на входах схемы:

$$p_1 = 0{,}10;\ p_2 = 0{,}13;\ p_3 = 0{,}16;\ p_4 = 0{,}19;$$
$$p_5 = 0{,}22;\ p_6 = 0{,}25;\ p_7 = 0{,}28;\ p_8 = 0{,}31;$$
$$p_9 = 0{,}34;\ p_{10} = 0{,}37;\ p_{11} = 0{,}40;\ p_{12} = 0{,}43;$$
$$p_{13} = 0{,}46;\ p_{14} = 0{,}49;\ p_{15} = 0{,}52. \qquad (9)$$

Следует заметить, что приведенные в таблице оценки энергопотребления имеют логический характер и основаны на использовании вероятностных методов вычисления переключательных активностей сигналов. В данном случае эти методы основаны на вычислении плотностей переключений сигналов (6) на входных полюсах схемы и выходных полюсах элементов И, реализующих конъюнкции.

Из табл. 1 видно, что для таких сравнительно несложных минимизируемых систем булевых функций вычислительные затраты на минимизацию с учетом и без учета энергопотребления практически не различаются. Не сильно различаются и оценки энергопотребления в силу того, что в обоих случаях используются приближенные методы минимизации.

Чтобы оценить реальное энергопотребление схем, построенных по минимизированным системам ДНФ, были проведены эксперименты по аналоговому моделированию схемотехнических описаний этих схем. Использовался маршрут аналогового моделирования BoardStation Flow в САПР Mentor Graphics [11]. Оценка энергопотребления проводилась по методике, предложенной в работе [12]. Описание систем автоматически [13] переводилось на язык VHDL. Затем средствами синтезатора LeonardoSpectrum [11] синтезировалась логическая схема в библиотечном базисе КМОП-схем, по составу, аналогичному базису, описанному в [14].

Полученное текстовое описание схемы сохранялось в формате Edif, а затем с помощью средств [12] переводилось в SPICE-описание схемы. Схемотехническое моделирование проводилось с помощью программы AccuSim II при следующих значениях основных параметров моделирования [12]: длительность передних и задних фронтов входных сигналов 1 нс, период подачи сигналов 40 нс, температура +27 °С, напряжение питания 3,3 В.

Моделирование проводилось на тестовой последовательности входных воздействий, наборы которой генерировались таким образом, что вероятности появления в них значений 1 и 0 для каждого входного сигнала были равны значениям, принятым при минимизации систем ДНФ (в данном случае они совпадали с (9)). Моделирование проводилось на 512 тестовых наборах для схем b12, life, root, tms, z9sym, и на 128 тестовых наборах для более сложных схем in0, mlp4.

Таблица 2

Результаты моделирования схем

Пример	Первая схема			Вторая схема		
	p^1	s^1	I^1	p^2	s^2	I^2
b12	91	33575	0,47293	97	35176	**0,47201**
in0	377	141141	**1,09385**	433	157802	1,28238
life	50	18341	0,30817	52	18922	**0,26176**
mlp4	422	155425	0,59325	428	158020	**0,58931**
root	118	41789	0,38138	115	41158	**0,38136**
tms	166	56838	**0,62831**	175	60297	0,67691
z9sym	103	38396	0,51769	116	43616	**0,49248**

При моделировании оценивалось среднее значение потребляемого схемами тока, так как при постоянном напряжении питания потребляемая мощность легко подсчитывается умножением на константу.

В табл. 2 приведены значения тока, потребляемого схемами, реализованными в КМОП-базисе. Жирным шрифтом выделены варианты, выигравшие по энергопотреблению. Приняты следующие обозначения: p^1, p^2 — число элементов используемой КМОП-библиотеки в схемах, построенных по минимизированным системам ДНФ; s^1, s^2 — площади КМОП-схем, построенных по минимизированным системам ДНФ; I^1, I^2 — средние значения тока, потребляемого схемами, в миллиамперах.

Следует заметить, что результаты вычисления энергопотребления на логическом и схемотехническом уровнях (табл. 1 и 2) несколько разнятся. Причина заключается в том, что проводилось опосредованное сравнение результатов минимизации систем булевых функций на схемотехническом уровне: моделировались схемы в библиотечном базисе КМОП, построенные по минимизированным системам ДНФ, а для их построения использовались, в свою очередь, приближенные методы.

Заключение

Схемотехническое моделирование двухуровневых И—ИЛИ-схем, соответствующих одним и тем же системам ДНФ, полученным в результате минимизации с учетом и без учета энергопотребления, показало, что минимизация булевых функций с учетом переключательной активности сигналов позволяет получить уменьшение энергопотребления схем без увеличения их сложности. При этом вычислительные затраты на минимизацию возрастают незначительно.

Список литературы

1. **Brayton R. K., Hachtel G. D., McMullen C., Sangiovanni-Vincentelli A. L.** Logic minimization algorithms for VLSI synthesis. Kluwer Academic Publishers, Boston, Massachusetts, 1984. 193 p.
2. **Roy K., Prasad S. C.** Low Power CMOS VLSI Circuit Design. New York: John Wiley and Sons Inc., 2000. 376 p.
3. **Рабаи Ж. М.** Цифровые интегральные схемы. Методология проектирования. М.: Издат. дом Вильямс, 2007. 912 с.
4. **Benini L., Micheli G. De.** Logic Synthesis for Low Power / Eds. S. Hassoun, T. Sasao, R. K. Brayton // Logic Synthesis and Verification. Boston, Dardrecht, London: Kluwer Academic Publishers, 2002. P. 197—223.
5. **Pedram M.** Power Minimization in IC Design: Principles and Applications // ACM Transactions Design Automation Electronic Systems. 1996. Vol. 1. P. 3—56.
6. **Черемисинова Л. Д.** Оценка энергопотребления КМОП-схем на логическом уровне // Информационные технологии. 2010. № 8. С. 27—35.
7. **Najm F. N.** A survey of Power Estimation Techniques in VLSI Circuits // IEEE Transactions on VLSI. 1994. N 12. OP. 446—455.
8. **Торопов Н. Р.** Минимизация систем булевых функций в классе ДНФ // Логическое проектирование / Под ред. А. А. Закревского. Минск: Ин-т техн. кибернетики НАН Беларуси. 1999. Вып. 4. С. 4—19.
9. **Торопов Н. Р.** Минимизация системы булевых функций с поляризацией их значений // Сб. матер. IV Междунар. конф. "Автоматизация проектирования дискретных систем (CAD DD'2001)" (14—16 ноября 2001). Ч. 2. Минск: Ин-т техн. кибернетики НАН Беларуси, 2001. С. 92—105.
10. **URL:** http://www1.cs.columbia.edu/~cs4861/sis/espresso-examples/ex/
11. **Бибило П. Н.** Системы проектирования интегральных схем на основе языка VHDL. StateCAD, ModelSim, LeonardoSpectrum. М.: СОЛОН-Пресс, 2005. 384 с.
12. **Авдеев Н., Бибило П.** Оценка энергопотребления цифрового блока СБИС // Современная электроника. 2009. № 9. С. 46—49.
13. **Черемисинов Д. И.** Анализ и преобразование структурных описаний СБИС. Минск: Белорусская наука, 2006. 275 с.
14. **Лукошко Г., Коннов Е.** КМОП базовые матричные кристаллы серии К1574 // Радиолюбитель. 1997. № 9. С. 39—40.

УДК 004.9

В. Н. Гридин[1], д-р техн. наук, проф.,
e-mail: info@dite.ras.ru,
Г. Д. Дмитревич[2], д-р техн. наук, проф.,
Д. А. Анисимов[2],
д-р техн. наук, проф., гл. науч. сотр.
[1]ЦИТП РАН
[2]СПбГЭТУ

Построение систем автоматизированного проектирования на основе Web-технологий

Рассматриваются вопросы внедрения Web-технологий в системы автоматизированного проектирования. Приводится сравнительная оценка возможных способов построения программного обеспечения систем моделирования с использованием технологии Web-сервисов. Дается описание реализации программного обеспечения распределенной системы схемотехнического проектирования.

Ключевые слова: *системы автоматизированного проектирования, системы моделирования, схемотехническое проектирование, Web-технологии, Web-сервисы*

Введение

Широкое внедрение систем автоматизированного проектирования в практику решения инженерных задач существенно ограничивается высокой стоимостью лицензионного программного обеспечения САПР. Вместе с тем, создание собственных интегрированных систем автоматизированного проектирования связано с огромными затратами трудовых и материальных ресурсов и не может быть реализовано в сжатые строки. Проблема усложняется также и тем, что в реальных условиях эксплуатации современные многофункциональные САПР используются, как правило, крайне неэффективно, поскольку из входящих в их состав подсистем в процессе решения конкретных задач часто применяется не более 10...20 % программного обеспечения, наиболее специфичного для данного подразделения.

Выходом из создавшегося положения во многих случаях может стать децентрализация архитектуры САПР путем перехода к распределенным системам проектирования, построенным на основе Интернет-технологий, решающих задачи коммуникации и информационного обмена между приложениями, которые разрабатываются и развертываются на гетерогенных платформах. Такие независимо управляемые приложения являются автономными и могут взаимодействовать друг с другом в процессе выполнения общей задачи. Протоколы Интернет-технологий предоставляют надежную базу для связывания отдельных подсистем и не требуют согласованного использования ресурсов, находящихся в разных узлах сети, что существенно упрощает процесс разработки и эксплуатации распределенной системы проектирования. Основным требованием для возможности реализации такой распределенной системы является согласованность интерфейсов, посредством которых отдельные подсистемы взаимодействуют между собой. При выполнении этого требования отдельные компоненты распределенной системы могут создаваться различными разработчиками и поддерживаться на различных сайтах, откуда они будут поставляться (возможно, на коммерческой основе) потребителям.

Технология Web-сервисов

Наиболее эффективным методом интеграции отдельных подсистем в распределенное приложение является удаленный вызов процедур на основе сервис-ориентированной архитектуры (Service-Oriented Architecture, SOA) с использованием технологии Web-сервисов, позволяющих приложениям взаимодействовать друг с другом независимо от платформы, на которой они развернуты, а также от языка программирования [1, 2].

Web-сервисы базируются на XML-стандарте и позволяют пользователям обеспечивать взаимодействие с внешними программными средствами через сеть Интернет. Технически Web-сервисы — это модульные бизнес-приложения, имеющие стандартные интерфейсы для работы через сеть Интернет и являющиеся слабосвязанными компонентами программного обеспечения, доступными для использования через стандартные Интернет-протоколы. При этом сервис действует как черный ящик: клиент должен только найти этот сервис, знать, какую функциональность он представляет, а способ реализации сервиса для клиента не имеет значения.

При построении Web-сервисов используются несколько спецификаций, которые основываются на открытых стандартах, при этом стек протоколов технологии Web-сервисов выполняется в приводимой ниже последовательности:

- поиск Web-сервиса с помощью протокола UDDI;
- описание Web-сервиса на основе WSDL;
- вызов Web-сервиса через протокол SOAP;
- кодирование данных (XML, XML Schema);
- транспортировка (HTTP).

Web-сервис описывает набор методов, которые могут быть вызваны удаленно по сети посредством стандартизированных XML-сообщений. С помощью этих сообщений для обмена информацией между системами можно описать любые данные независимым от платформы способом, что, в свою очередь, приводит к слабосвязанным приложениям. Поскольку Web-сервисы могут функционировать на более высоком уровне абстракции, анализируя и обрабатывая типы данных динамическим образом, то отдельным компонентам программного обеспечения предоставляется возможность взаимодействовать более открыто. При использовании универсально описанных интерфейсов появляется также возможность использовать программные компоненты повторно, что позволяет снизить трудоемкость разработки САПР. Следует отметить, что Web-сервисы — это не всякая услуга, оказываемая через сеть Интернет посредством клиент-серверной технологии. В отличие от услуг, предоставляемых клиенту-человеку через связанный с Web-сервером броузер, услуга Web-сервиса предоставляется клиенту-программе.

Существенным достоинством технологии Web-сервисов является возможность их постоянной модификации и расширения путем добавления новых, более продуктивных методов. Единственным требованием для реализации таких возможностей является соблюдение неизменности выбранного при разработке интерфейса Web-сервиса. Поскольку концепция распределенных сервис-ориентированных САПР предполагает возможность создания и постоянного совершенствования клиентских приложений программистами, которые не принимали участия в построении программного обеспечения Web-сервисов, все разрабатываемые Web-сервисы должны быть самодокументируемыми. Последнее означает, что каждый Web-сервис обязательно должен включать в себя чисто информационный метод, не имеющий параметров и возвращающий простую строковую переменную. В тексте, связанном с этой переменной, должны содержаться все сведения о методах, обеспечивающих бизнес-логику Web-сервиса, с описанием форматов передаваемых им параметров и возвращаемых данных.

Особенности построения клиентских приложений

При практической реализации построения систем автоматизированного проектирования с использованием Web-технологии существенное внимание следует уделять правильному разделению функциональных обязанностей, возлагаемых на основное клиентское приложение и на Web-сервис, взаимодействующий с ним. Так, можно сделать клиентское приложение САПР "тонким", сохранив за ним только отправку необработанных исходных данных и отображение окончательных результатов расчета без дополнительной обработки на клиентской стороне. В этом случае на Web-сервис возлагаются задачи окончательного формирования структуры входных данных, их обработки в соответствии с используемыми алгоритмами расчета и формирования результатов непосредственно в той форме, которая будет отображаться на клиентском приложении. Можно, наоборот, сделать клиентское приложение "толстым", возложив на него все задачи формирования и модификации исходных данных, некоторые этапы выполнения расчетных процедур, а также преобразование выходных данных Web-сервиса к требуемой форме, устанавливаемой в процессе диалогового взаимодействия с пользователем. В этом случае Web-сервис выполняет ограниченные функции и для его размещения не требуется мощный сервер, однако разработка клиентского приложения усложняется. Решение вопросов, связанных с разделением функций, выполняемых клиентским приложением и Web-сервисом, имеет принципиальное значение для выбора наиболее рациональной архитектуры САПР, особенно в случаях использования многоуровневой структуры, при которой один Web-сервис может обращаться в процессе своей работы к другим Web-сервисам.

При построении САПР на основе технологии Web-сервисов возможны следующие способы реализации клиентских приложений:

- консольное приложение;
- оконное приложение;
- Web-приложение.

Консольные приложения не имеют графического интерфейса и запускаются из командной строки любой операционной системы. Несмотря на ограниченные возможности таких приложений использование их может оказаться целесообразным для тестирования вновь разработанных Web-сервисов, а также при реализации простейших САПР на основе портативных компьютеров с ограниченной площадью экрана.

Оконные клиентские приложения позволяют в полной мере использовать графический интерфейс и наиболее всего подходят для построения САПР на основе технологии Web-сервисов. При этом для одного и того же Web-сервиса возможно построение нескольких однотипных приложений с разными возможностями диалогового взаимодействия с пользователем.

Web-приложения позволяют полностью разместить все программное обеспечение распределенной САПР в сети Интернет, при этом вызов Web-сервиса осуществляется из одной или из нескольких Web-страниц приложения. Достоинством такой структуры является возможность открытого доступа к использованию САПР через любой броузер, недостатком — значительное увеличение времени, требуемого для ввода описания компонентов моделируемой системы, вызванное необходимостью ожидания ответа на каждом шаге ввода исходных данных.

Вне зависимости от вида клиентского приложениям вызов Web-сервисов реализуется одинаковым способом, при этом для одного и того же Web-сервиса можно использовать любые разновидности клиентских приложений, написанные на различных языках программирования.

Реализация программного обеспечения распределенной САПР

Изложенные концепции построения сервис-ориентированных САПР реализованы путем разработки ряда автономных Web-сервисов, обеспечивающих поддержку процесса схемотехнического проектирования электронной аппаратуры, при этом каждый Web-сервис предназначен для решения одной типовой задачи схемотехнического проектирования. К таким задачам отнесены расчеты:

- частотных характеристик линейных систем;
- стационарного режима нелинейных систем;
- линеаризованных систем в частотной области;
- динамических режимов нелинейных систем;
- чувствительности схемных функций линейных систем к вариации параметров;
- чувствительности переменных нелинейных систем к вариации параметров.

В качестве компонентов моделируемых схем могут быть использованы двухполюсники типа *R*, *C*, *L*, линейные частотно-зависимые управляемые источники, нелинейные управляемые источники, трансформаторы, биполярные транзисторы, униполярные транзисторы, операционные усилители, а также задающие источники тока и напряжения.

Кроме основных методов, реализующих вычислительные процедуры типовых задач схемотехнического проектирования, все разработанные Web-сервисы включают в себя вспомогательный метод getInf0, не содержащий параметров и возвращающий текстовую строку. В этой строке содержится текст с информацией о назначении и имени основного метода, реализующего бизнес-логику решения той или иной типовой задачи, о форматах массивов, описывающих вводимые данные, о директивах расчета. Кроме этого, приводится формат возвращаемого обобщенного массива, в котором содержатся результаты работы основного метода. Вызов метода getInf0 можно осуществить непосредственно из интегрированной среды разработки клиентского приложения еще до реализации этого приложения. В случае построения клиентского приложения на языке Java необходимая информация обеспечивается путем вызова в интегрированной среде разработки программного обеспечения инструмента Web Services Explorer, в котором по URL-адресу Web-сервиса можно получить доступ к Soap-ответу метода getInf0. Конверт этого ответа Soap Response Envelope содержит отформатированный текст с полной информацией обо всех основных методах, которые могут быть вызваны из выбранного Web-сервиса, включая форматы всех передаваемых и возвращаемых этими методами данных. Аналогичная поддержка для работы с методом getInf0 обеспечивается также и при создании клиентского приложения на языке C# в среде Microsoft.Net.

В целях обеспечения максимальной степени платформенной независимости для программной реализации всех Web-сервисов использован язык Java. Для тестирования разработанных Web-сервисов и проверки возможности их использования в гетерогенных средах реализованы автономные клиентские приложения, написанные как на языке Java, так и на языке C#.

Разработанные Web-сервисы могут быть размещены произвольным образом в одном или нескольких узлах сети Интернет. Возможно также размещение их в корпоративной сети Интранет. Последний способ реализации системы, хотя и ограничивает доступ к ней внешних пользователей, но позволяет наиболее простым способом обеспечить возможность постоянной модификации серверных и клиентских приложений в соответствии с меняющимися условиями проектирования.

Заключение

Таким образом, использование Интернет-технологий при разработке программного обеспечения распределенных систем автоматизированного проектирования на основе Web-сервисов позволяет:

- перейти к концепции описания интерфейсов и взаимодействий на основе XML, объединяя любой тип приложения с другим приложением и предоставляя свободу изменения и развития с течением времени до тех пор, пока поддерживается соответствующий интерфейс;
- использовать более высокий уровень абстракции программного обеспечения, при котором оно может быть задействовано пользователями, работающими только на уровне бизнес-анализа;
- взаимодействовать на произвольной платформе между различными Web-сервисами и клиентскими приложениями, написанными на любых языках программирования;
- учитывать слабосвязанность программного обеспечения, благодаря которой взаимодействие между приложениями сервиса не нарушается каждый раз, когда меняется дизайн или реализация какого-либо сервиса;
- предоставлять существующему или унаследованному программному обеспечению сервисный интерфейс без изменения оригинальных приложений, давая им возможность полноценно взаимодействовать в сервисной среде;
- адаптировать существующие приложения к изменяющимся условиям проектирования и потребностям заказчика.

Практическая реализация полученных результатов позволяет существенно повысить эффективность разработки систем автоматизированного проектирования, обеспечивает функционирование распределенных САПР в гетерогенных средах и предоставляет возможность последующей модификации программного обеспечения в процессе эксплуатации.

Список литературы

1. **Морган С., Райан Б., Хорн Ш., Бломсма М.** Разработка распределенных приложений на платформе Microsoft.Net. М., СПб.: "Питер", 2008.

2. **Дей Н., Мандел Л., Райман А.** ECLIPSE: платформа Web-инструментов. Разработка Web-приложений на языке Java. СПб.: Кудиц-Пресс, 2008.

Поздравляем Юбиляра!

Главному научному сотруднику ИПУ им. В. А. Трапезникова РАН, ведущему научному сотруднику ИМАШ им. А. А. Благонравова РАН, профессору МГУ им. М. В. Ломоносова, МГУПИ и МИРЭА(ТУ), заместителю главного редактора журнала "Информационные технологии", доктору технических наук, профессору

Николаю Борисовичу ФИЛИМОНОВУ

исполнилось **60 лет**

Н. Б. Филимонов — талантливый ученый и преподаватель, видный специалист в области автоматизации процессов управления, один из ярких представителей отечественной научной школы по технической кибернетике В. В. Солодовникова. К числу его основных научных достижений следует отнести формирование нового перспективного научного направления — теории полиэдральной оптимизации дискретных динамических процессов управления и наблюдения. Он внес заметный вклад в развитие ряда ключевых направлений современной теории автоматического управления: проблемы динамического качества процессов управления, принципа динамической развязки многоконтурных систем регулирования, концепции многорежимного управления сложными динамическими объектами, методов анализа и синтеза систем терминального управления. В последние годы его научные интересы связаны с актуальными вопросами автоматизации эргатических систем управления подвижными объектами и интеллектуализации мехатронных информационно-управляющих систем.

Высокий профессионализм, широкая эрудиция, большая трудоспособность, чуткость и отзывчивость снискали ему большое уважение учеников и коллег.

Уважаемый Николай Борисович!
Редакционная коллегия и редакция журнала поздравляют Вас
и желают Вам крепкого здоровья, большого счастья, новых творческих успехов
в Вашей многогранной деятельности!

УДК 004.021

А. В. Бобченков, аспирант,
e-mail: groddenator@gmail.com,
В. В. Топорков, д-р техн. наук, проф., зав. каф.,
e-mail: ToporkovVV@mpei.ru,
Московский энергетический институт
(технический университет)

Метод оптимального планирования управляемых потоков заданий в распределенных вычислительных средах

Предложен и исследован метод планирования на уровне потоков заданий в распределенных вычислительных средах, использующий экономическую модель для повышения эффективности распределения ресурсов.

Ключевые слова: *распределенные вычисления, планирование заданий, метапланировщик, распределение ресурсов*

Введение

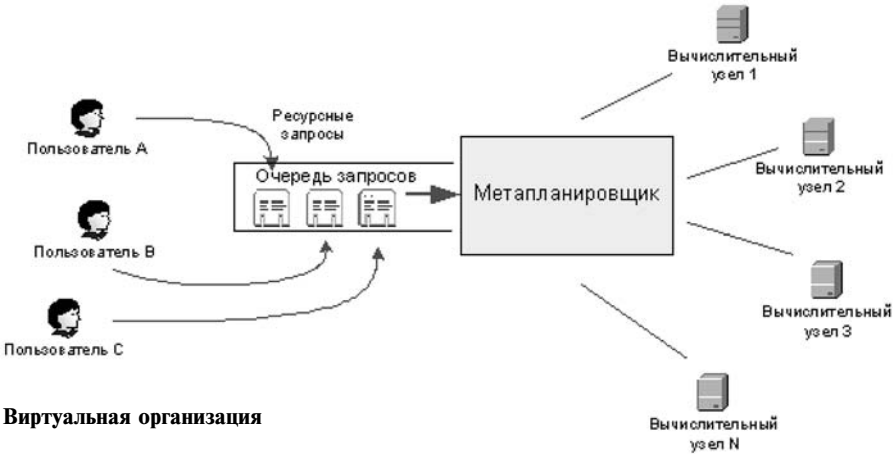
При организации вычислений в распределенных средах одной из основных задач является задача планирования и эффективного распределения ресурсов. В больших распределенных вычислительных системах решение данной задачи осложняется разнородностью состава доступных вычислительных ресурсов и большим числом наступающих событий, связанных с динамикой доступности ресурсов. В рамках работы виртуальной организации (ВО), объединяющей пользователей, собственников ресурсов и координаторов работы системы, возникает проблема одновременного соблюдения экономических интересов всех сторон, участвующих в работе системы. Наличие определенных правил предоставления и потребления ресурсов [1] позволяет повысить эффективность планирования и распределения ресурсов на уровне потоков заданий. В настоящее время различные варианты экономического подхода рассматриваются в рамках грида [5] и облачных вычислений [4]. Следует отметить две главные тенденции применения экономического подхода. Одна из них заключается в реализации децентрализованной модели рынка ресурсов, например в виде аукционов [2], другая — в оптимизации на уровне потоков заданий с помощью централизованного процесса-посредника (грид-диспетчера, метапланировщика) [3, 5, 6].

В данной статье рассматривается класс распределенных вычислительных систем с неотчуждаемыми ресурсами. Организация вычислений в системе построена иерархически и обеспечивается метапланировщиком, осуществляющим планирование на уровне потоков заданий и взаимодействующим с локальными планировщиками (менеджерами ресурсов) вычислительных узлов. В основе модели лежит принцип справедливого распределения ресурсов между пользователями [6], который реализуется с помощью введения ограничений в процессе планирования. Пользователь вправе рассчитывать на более ранний запуск своих заданий и скорейшее их выполнение, если согласен внести более высокую плату за ресурс.

Членами виртуальной организации являются (см. рисунок):

А. Пользователи, предоставляющие вычислительные задания вместе со спецификацией требований к качеству и количеству ресурсов в форме ресурсных запросов. В нашей модели ресурсный запрос представляет собой предлагаемую пользователем часть соглашения об уровне услуг (*service level agreement*, SLA).

В. Собственники вычислительного ресурса (*service providers*), устанавливающие стоимость единицы времени использования ресурса, а также контролирующие локальные расписания занятости ресурса.



Виртуальная организация

С. Администраторы метапланировщика, имеющие возможность устанавливать и изменять политику предоставления ресурсов.

Метапланировщик выбирает пользовательские задания на входе из общей очереди и формирует пакеты заданий, каждый из которых распределяется на определенном домене ресурсов — подмножестве из доступных на текущий момент ресурсов общего пула. Информация о занятости каждого из ресурсов на период, называемый циклом планирования, собирается с помощью специальных компонентов (агентов взаимодействия с вычислительными ресурсами) и интегрируется в форме таблицы ресурсных линий в памяти метапланировщика.

Метод планирования

Процесс планирования в нашей модели разбивается на два этапа. Первый этап заключается в подборе подходящих слотов для каждого из заданий пакета в соответствии с их ресурсным запросом. Алгоритм подбора слотов в нашей модели основан на алгоритме планировщика Maui для многопроцессорных задач и заключается в нахождении прямоугольного "окна" с высотой в N требуемых слотов с одинаковым временем начала [7]. В отличие от алгоритма Maui в нашей модели окно может иметь неровный правый край с учетом возможности включения слотов, предоставляемых вычислительными узлами с различной производительностью. Вторым отличием нашей модели является то, что этап подбора слотов заканчивается фиксацией нескольких альтернативных наборов слотов (альтернатив) для каждого из заданий, причем запущено и выполнено оно может быть на любой из альтернатив. Третье отличие заключается в наличии дополнительного этапа оптимизации в пространстве альтернативных наборов слотов.

Второй этап планирования — этап оптимизации — заключается в выборе такой комбинации из найденных на этапе подбора слотов альтернатив, которая бы обеспечивала оптимальное или близкое к оптимальному значение некоторого критерия с точки зрения прохождения всего пакета. Отдельная комбинация альтернативных наборов слотов (по одному для каждого задания) представляет собой *план выполнения*, в дальнейшем обрабатываемый подсистемами предварительного резервирования метапланировщика [8]. В рамках данной работы мы рассматриваем и исследуем метод поиска оптимального плана выполнения, основанный на методах динамического программирования.

Мы рассматриваем два типа критериев в рамках нашей модели. Это *стоимостные* и *временные показатели* эффективности прохождения пакета заданий $J = \{j_1, ..., j_n\}$ на одной из допустимых комбинаций альтернатив $\bar{s} = (s_1, ..., s_n)$, где $\bar{s}$ — план выполнения. К первой группе критериев относится суммарная стоимость выполнения пакета заданий

$$C(\bar{s}) = \sum_{i=1}^{n} c_i(s_i) \qquad (1)$$

или бюджет цикла.

Для того чтобы не допустить монополизации использования того или иного ресурса отдельными пользователями и реализовать принцип справедливого распределения ресурсов, вводится ограничение на бюджет цикла B^*.

Стоимость (1) выполнения пакета заданий, с одной стороны, отражает издержки пользователей, а с другой — доходы собственников ресурсов. Интересы собственников отражаются в таком критерии, как потери от недоиспользования бюджета цикла:

$$D(\bar{s}) = B^* - C(\bar{s}). \qquad (2)$$

Политика администрирования в виртуальной организации и отчасти интересы пользователей отражаются в таком критерии, как суммарное зарезервированное время выполнения пакета заданий

$$T(\bar{s}) = \sum_{i=1}^{n} t_i(s_i). \qquad (3)$$

Ограничение на суммарное время занятия слотов T^* выражает стремление собственников ресурсов сбалансировать доли внешних и локальных заданий. Как правило, доходы от предоставления ресурсов тем больше, чем большее время они задействованы в вычислениях. Упомянутые аспекты можно отразить в таком показателе, как простой ресурсов в цикле:

$$I(\bar{s}) = T^* - T(\bar{s}). \qquad (4)$$

Следует отметить, что предлагаемый метод можно применять в многокритериальном режиме для более эффективной реализации политики справедливого распределения ресурсов (например, с использованием вектора критериев (1)—(4)). В данной работе мы ограничиваемся рассмотрением случая однокритериальной оптимизации.

Формализуем задачу поиска оптимального по частному критерию плана выполнения. При описании схемы динамического программирования изложение будем вести в соответствии с *методом обратной прогонки* [11], который, как показывает опыт практического применения, является более эффективным с позиции вычислительной трудоемкости, чем метод прямой прогонки.

Обозначим $g_i(s_i)$ частную функцию, определяющую эффективность использования выбранного альтернативного набора слотов s_i для i-го задания. Иными словами, $g_i(s_i) = c_i(s_i)$ либо $g_i(s_i) = t_i(s_i)$.

Пусть $f_i(z_i)$ — экстремальное значение частного критерия при использовании набора s_i для выполнения заданий i, $i + 1$, ..., n, где z_i — суммарное время занятия или стоимость использования слотов s_i, s_{i+1}, ..., s_n для заданий j_i, j_{i+1}, ..., j_n.

Обозначим $z_i(s_i)$ допустимое значение времени или стоимости занятия слотов. Таким образом,

$$z_i(s_i) \leqslant Z_i \leqslant Z^*,$$

где Z^* — заданное ограничение. Например, если $z_i(s_i) = t_i(s_i)$, то $t_i(s_i) \leqslant T_i \leqslant T^*$, где T_i — суммарное время занятия слотов заданиями i, $i+1$, ..., n, а T^* — ограничение на значение T_i. Если критерием выбрана стоимость $z_i(s_i) = c_i(s_i)$, то $c_i(s_i) \leqslant C_i \leqslant B^*$, где C_i — суммарная стоимость использования ресурсов заданиями i, $i + 1$, ..., n, а B^* — ограничение на бюджет цикла. В схеме обратной прогонки $Z_1 = Z^*$ при $i = 1$, $0 \leqslant Z_i \leqslant Z^*$ при $1 < i \leqslant n$.

Функциональное уравнение для отыскания условного (при заданном $z_i(s_i)$) экстремума $f_i(z_i(s_i))$ процедуры обратной прогонки может быта записано следующим образом:

$$f_i(Z_i) = \operatorname*{extr}_{s_i}\{g_i(s_i) + f_{i+1}(Z_i - z_i(s_i))\},\ i = 1,\ ...,\ n;$$
$$f_{n+1}(Z_{n+1}) \equiv 0. \qquad (5)$$

Экстремум в (5) отыскивается по альтернативным наборам слотов для i-го задания пакета. В соответствии с терминологией, традиционно принятой в динамическом программировании, будем называть состоянием системы Z_i. Условно-оптимальные (при заданном Z_i) значения времени или стоимости использования ресурсов пакетом заданий определяются в соответствии с рекуррентным соотношением

$$z_i^\bullet = \arg \operatorname*{extr}_{z_i(s_i) \leqslant Z_i} f(z_i);$$
$$Z_i = Z_{i-1} - z_{i-1}^\bullet(s_{i-1}),\ i > 1;$$
$$Z_1 = Z^*. \qquad (6)$$

Условно-оптимальный план выполнения $\bar{s}^\bullet = (s_1^\bullet, ..., s_n^\bullet)$ для пакета заданий J образуется альтернативными наборами слотов для отдельных заданий, которые имеют следующий вид:

$$s_i^\bullet = \arg\operatorname*{extr}_{s_i}\{f_i(z_i^\bullet(s_i))\},\ i = 1,\ ...,\ n. \qquad (7)$$

Рассмотрим применение описанной выше схемы динамического программирования на примере задачи максимизации доходов собственников ресурса при ограничении на время занятия слотов.

Пусть необходимо найти такую комбинацию наборов слотов для выполнения пакета из трех заданий (табл. 1), чтобы обеспечить максимум суммарной стоимости (1) использования ресурсов.

Таблица 1

Стоимость и суммарное время альтернативных наборов слотов

Альтернативный набор	Задание 1		Задание 2		Задание 3	
	c_1	t_1	c_2	t_2	c_3	t_3
1	7	7	3	3	5	5
2	10	5	2	1	8	4
3	9	3	—	—	6	2
4	—	—	—	—	4	1

Задано ограничение на суммарное время занятия слотов заданиями:

$$T^* = \sum_{i=1}^{n} \sum_{s_i} \left[\frac{t_i(s_i)}{l_i}\right], \qquad (8)$$

где l_i — число допустимых наборов слотов для выполнения i-го задания.

Таким образом, в соответствии с формулой (8) T^* представляет собой сумму средних значений временных отрезков по всем альтернативным для i-го задания наборам слотов, округленных до целого числа, не большего $t_i(s_i)/l_i$. В соответствии с данными табл. 1 $T^* = 10$ единиц.

Решение задачи проведем поэтапно, ассоциируя i-й этап с получением максимального дохода собственников ресурсов на этапах i, $i + 1$, ..., n.

Согласно формуле (5)

$$f_i(T_i) = \max_{s_i}\{c_i(s_i) + f_{i+1}(T_i - t_i(s_i))\},\ i = 1,\ 2,\ 3,$$
$$f_4(T_4) \equiv 0. \qquad (9)$$

В соответствии с методом обратной прогонки [11] определим состояние системы T_i следующим образом:

T_1 — суммарное время занятия слотов заданиями 1, 2 и 3;

T_2 — суммарное время занятия слов заданиями 2 и 3;

T_3 — время занятия слотов заданием 3.

Следовательно, в соответствии с процедурой обратной прогонки (5)—(7) порядок поэтапных вычислений определяется последовательностью $f_3(T_3) \to f_2(T_2) \to f_1(T_1)$.

Приведем выражения для каждого этапа. Полагаем, что $f_i(T_i) = 0$ при $T_i = 0$.

Этап 3: $f_3(T_3) = \max\limits_{s_3=1,...,4}\{c_3(s_3)\}$, $t_3(s_3) \leqslant T_3$.

Этап 2: $f_2(T_2) = \max\limits_{s_2=1,2}\{c_2(s_2) + f_3(T_2 - t_2(s_2))\}$, $t_2(s_2) \leqslant T_2$, где значения $f_3(T_2 - t_2(s_2))$ берутся из табл. 2 этапа 3.

Этап 1: $f_1(T_1) = \max\limits_{s_1=1,2,3}\{c_1(s_1) + f_2(T_1 - t_1(s_1))\}$, $t_1(s_1) \leqslant T_1$, $T_1 = T^*$, где значения $f_2(T_1 - t_1(s_1))$ определены в табл. 3 этапа 2.

Таблица 2

Оптимизация (этап 3)

Суммарное время T_3	$c_3(s_3)$				Оптимальное решение	
	$s_3 = 1$	$s_3 = 2$	$s_3 = 3$	$s_3 = 4$	$f_3(T_3)$	s_3^*
0	—	—	—	—	0	—
1	—	—	—	4	4	4
2	—	—	6	4	6	3
3	—	—	6	4	6	3
4	—	8	6	4	8	2
5	5	8	6	4	8	2
...	...	...	...	...	...	...
10	5	8	6	4	8	2

Таблица 3

Оптимизация (этап 2)

Суммарное время T_2	$c_2(s_2) + f_3(T_2 - t_2(s_2))$		Оптимальное решение	
	$s_2 = 1$	$s_2 = 2$	$f_2(T_2)$	s_2^*
0	—	—	0	—
1	—	—	0	—
2	—	2 + 4 = 6	6	2
3	—	2 + 6 = 8	8	2
4	3 + 4 = 7	2 + 6 = 8	8	2
5	3 + 6 = 9	2 + 8 = 10	10	2
6	3 + 6 = 9	2 + 8 = 10	10	2
7	3 + 8 = 11	2 + 8 = 10	11	1
...	...	...	...	...
10	3 + 8 = 11	2 + 8 = 10	11	1

Таблица 4

Оптимизация (этап 1)

Суммарное время T_1	$c_1(s_1) + f_2(T_1 - t_1(s_1))$			Оптимальное решение	
	$s_1 = 1$	$s_1 = 2$	$s_1 = 3$	$f_1(T_1)$	s_1^*
10	7 + 8 = 15	10 + 10 = 20	9 + 11 = 20	20	[2], [3]

Таблица 5

Условно-оптимальные комбинации альтернатив

T_1	$s_1^\bullet$	T_2	$s_2^\bullet$	T_3	$s_3^\bullet$	$(s_1^\bullet, s_2^\bullet, s_3^\bullet)$
→(**10**)	[1]	→10−7=3	[2]	→ 3−1=2	[3]	→ (1, 2, 3)
	[2]	→10−5=5	[2]	→ 5−1=4	[2]	→ **(2, 2, 2)**
	[3]	→10−3=7	[1]	→ 7−3=4	[2]	→ (3, 1, 2)

В таблицах прочерк в столбце $c_i(s_i)$ означает, что для заданного значения T_i не существует набора слотов. В столбце s_i указывается номер альтернативного набора слотов, соответствующего значению критериальной функции. Как можно видеть из табл. 4, некоторому значению критериальной функции может соответствовать более одной альтернативы.

Завершающий этап (прямая прогонка) заключается в построении списка условно-оптимальных комбинаций альтернатив. В табл. 5 последовательность восстановления комбинации показана стрелками. Начиная с каждой из альтернатив первого задания, мы последовательно вычитаем суммарное время $t_i(s_i)$ из значения запаса суммарного времени T, которое сначала приравнивается к T^*. Например, если $s_1^* = 2$, тогда на выполнение второго и третьего задания требуется выделить время $T_2 = T_1 - t_1^*(2) = 10 - 5 = 5$ единиц. Оптимальная альтернатива, соответствующая $T_2 = 5$, ищется в табл. 3 и так до полного построения комбинации. В рассматриваемом примере оптимальными являются комбинации (2,2,2) и (3,1,2), так как при использовании таких наборов слотов обеспечивается максимум критериальной функции. В семейство условно-оптимальных комбинаций входит также вариант (1,2,3).

Экспериментальные данные

Для исследования предложенного метода поиска оптимального плана выполнения была разработана система имитационного моделирования виртуальной организации, позволяющая ставить массовые эксперименты [10]. В систему были включены модули, генерирующие виртуальную среду, пакеты заданий, а также имитирующие непосредственно работу метапланировщика.

В качестве задачи для эксперимента был предложен сравнительный анализ эффективности описанного метода планирования и метода планирования Maui [7], в котором заданию назначается первый найденный подходящий набор слотов. В нашем случае мы моделируем такое поведение подбором случайной комбинации альтернатив, однако подчиняющейся тем же правилам ограничения при планировании.

Суть эксперимента заключалась в следующем. Было проведено 5000 циклов планирования в двух режимах:

- с использованием описанного выше метода поиска оптимального плана;
- случайного выбора комбинации альтернатив, но не превышающей по суммарным показателям (стоимость, время занятия слотов) ограничение цикла.

В каждом цикле планирования задавалось новое состояние среды и пакет заданий. Число заданий в пакете равнялось 30. Подбор слотов в обоих случаях проводился одинаково. Если для некоторых заданий пакета не находилось ни одной альтернативы, в таком случае данные задания не обрабатывались. Данный эксперимент был поставлен в условиях четырех модельных задач:

- Задача 1. Максимизация доходов собственников ресурсов при ограничении на суммарное время использования слотов.

Таблица 6

Результаты серии экспериментов (задача 1)

Режим	Число обработанных заданий	Суммарная стоимость слотов $C(s)$, ед. стоимости	Суммарное занятие слотов $T(s)$, ед. времени	Ограничение на суммарное занятие слотов T^*, ед. времени
МПОПВ	20,08	11945,98	421,22	471,14
Случайный выбор	20,09	10588,53	459,36	471,85

- Задача 2. Минимизация суммарного времени завершения пакета заданий при ограничении на бюджет виртуальной организации.
- Задача 3. Минимизация суммарной стоимости выполнения пакета заданий при ограничении на суммарное время использования слотов.
- Задача 4. Максимизация загрузки слотов при ограничении на время их использования.

Таким образом, всего было проведено четыре серии циклов, каждый из которых включал обработку 5000 циклов планирования исследуемым методом (в табл. 6 сокращенно "МПОПВ") и 5000 циклов случайным выбором.

В каждой серии циклов планирования были получены следующие существенные выходные параметры (на примере задачи 1):

В ходе работы было показано, что выходные параметры эксперимента имеют распределение, близкое к нормальному закону, что дает право сравнивать их усредненные значения для обоих методов. Полученные значения ограничения времени занятия слотов T^* (или бюджета цикла C^*), зависящие по формуле (8) только от параметров найденных альтернатив, различались не более чем на 0,3 %. Не более чем на 0,1 % в обоих случаях различалось число обработанных заданий пакета. Такие результаты позволяют утверждать, что исследуемый метод и метод случайного выбора тестировались в равных условиях.

Из табл. 6 можно видеть, что в условиях модельной задачи 1, где критерием эффективности является суммарная стоимость слотов, метод поиска оптимального плана выполнения дает существенный выигрыш в стоимости слотов при меньшем суммарном занятии слотов, что делает его применение более выгодным для собственников ресурсов. Выигрыш в эффективности планирования был достигнут во всех модельных задачах.

Ниже приведены оценки выигрыша исследуемого метода по сравнению со случайным выбором альтернатив:

Задача 1 . . . . . 12,82 %
Задача 2 . . . . . 10,6 %
Задача 3 . . . . . 6,18 %
Задача 4 . . . . . 2,9 %

Заключение

В данной статье предложен и исследован метод поиска оптимального плана выполнения, в результате работы которого пакету заданий сопоставляется оптимальная по некоторому критерию комбинация наборов слотов. Гибкость метода позволяет использовать его как для обеспечения и усиления экономических интересов одной из сторон, так и для балансировки экономических интересов всех участников и организации справедливого распределения ресурсов. В ходе экспериментов было показано преимущество перед существующими методами планирования в метапланировщиках с возможностью обработки многопроцессорных заданий.

В качестве целей будущего исследования можно назвать повышение эффективности работы метода в однокритериальном и многокритериальном режимах, реализацию в метапланировщике обратной связи между результатами и параметрами планирования, а также поиск возможности прогноза результата планирования исходя из текущего состояния среды и характеристик пакета заданий.

Список литературы

1. **Kurowski K., Nabrzyski J., Oleksiak A.** et al. Multicriteria Aspects of Grid Resource Management // J. Nabrzyski, J. M. Schopf and J. Weglarz (eds.), Grid resource management. State of the art and future trends. Boston: Kluwer Academic Publishers. 2003. P. 271—293.

2. **Buyya R., Abramson D., Giddy J.** et al. Economic Models for Resource Management and Scheduling in Grid Computing // J. of Concurrency and Computation: Practice and Experience. 2002. Vol. 14. N 5. P. 1507—1542.

3. **Ernemann C., Hamscher V., Yahyapour R.** Economic scheduling in Grid computing // Proc. of the 8th Int. JSSPP Workshop. LNCS. 2002. Vol. 2537. P. 129—152.

4. **Ailamaki A., Dash D., Kantere V.** Economic Aspects of Cloud Computing // Flash Informatique. Special HPC, 27 October 2009. P. 45—47.

5. **Garg S. K., Buyya R., Siegel H. J.** Scheduling Parallel Applications on Utility Grids: Time and Cost Trade-off Management // Proceedings of the 32nd Australasian Computer Science Conference (ACSC 2009), Wellington, New Zealand. 2009. P. 151—159.

6. **Коваленко В. Н., Коваленко Е. И., Корягин Д. А.** и др. Управление параллельными заданиями в гриде с неотчуждаемыми ресурсами. Препринт № 63. М.: ИПМ им. М. В. Келдыша РАН, 2007. 28 с.

7. **Jackson D., Snell Q., Clement M.** Core algorithms of the Maui scheduler // Job Scheduling Strategies for Parallel Processing, D. G. Feitelson and L. Rudolf (eds.). LNCS. Heidelberg: Springer, 2001. Vol. 2221. OP. 87—102.

8. **Топорков В. В.** Опорные планы согласованного выделения ресурсов при организации распределенных вычислений на масштабируемых системах // Программирование. 2008. № 5. С. 50—64.

9. **Топорков В. В.** Модели распределенных вычислений. М.: Физматлит, 2004. 320 с.

10. **Бобченков А. В., Топорков В. В., Целищев А. С.** Система имитационного моделирования планирования потоков независимых заданий в распределенных вычислительных средах // Тр. научно-технической конференции "Информационные средства и технологии". В трех томах. М.: МЭИ. 2010. Т. 1. С. 11—17.

11. **Таха Х.** Введение в исследование операций. В 2-х книгах. Кн. 1. М.: Мир, 1985. 479 с.

УДК 002.55, 004.4’234, 004.514.4

А. С. Зуев, канд. техн. наук, доц.,
О. Б. Кучеров, студент,
e-mail: zuev_andrey@mail.ru
Московский государственный университет
приборостроения и информатики

Внедрение контекстно-зависимых элементов управления в графические интерфейсы компьютерных программ

Обоснована целесообразность внедрения в графические интерфейсы компьютерных программ контекстно-зависимых элементов управления, обеспечивающих интерактивный доступ пользователя к командам, предоставляемым выбранным объектом, и его свойствам. Предложены варианты развития и расширения сферы применения данных элементов управления на примере текстового редактора MS Word, выполнен обзор существующих аналогов, обоснованы эргономичность и эффективность применения выполненных разработок.

Ключевые слова: *графический интерфейс, оптимизация графических интерфейсов, проектирование графических интерфейсов, эргономика программного обеспечения, человеко-компьютерное взаимодействие, контекстные элементы интерфейса*

Введение

Одним из важнейших свойств современного программного обеспечения (ПО) является интерактивность — способность без участия человека, активно и разнообразно реагировать на действия пользователя [1]. В настоящее время в развитии всех видов ПО, ориентированного на интерактивное взаимодействие с пользователем, наблюдается тенденция увеличения числа предусмотренных функциональных возможностей (опций). Основной причиной этого является стремление производителей ПО повысить конкурентоспособность своих продуктов за счет обеспечения их универсальности и расширения сферы применения. Следствием данной тенденции явилось усложнение графических пользовательских интерфейсов (ГПИ), являющихся средством реализации человеко-компьютерного взаимодействия.

Обеспечение пользователя средствами выполнения каждой дополнительной функциональной возможности ПО предполагает необходимость включения в состав ГПИ одного или нескольких соответствующих элементов интерфейса — дополнительных окон, пунктов меню, кнопок, переключателей, полей ввода и т. д. Увеличение числа элементов интерфейса приводит к росту информационной нагрузки на пользователя и усложнению процесса его взаимодействия с ПО. Это обусловливает необходимость оптимизации принципов работы пользователя с ГПИ и поиска наилучших вариантов размещения его элементов в основном и дополнительных окнах программы.

В то же время конкурентоспособность современного ПО определяется не только составом предусмотренных в нем функциональных возможностей, но и качеством ГПИ, подразумевающим, в том числе, обеспечение удобства работы оператора и его взаимодействия с программой. С точки зрения принципов организации человеко-компьютерного взаимодействия [2] это означает, что:

- любая задача пользователя должна решаться минимальным числом ментальных и физических действий;
- логика требующихся действий должна быть очевидна пользователю;
- требующиеся от пользователя перемещения курсора и внимания (взгляда) должны быть оптимизированы по критериям минимизации их количества и сложности выполнения (расстояния).

Ментальные действия обусловлены необходимостью принятия пользователем решений, их примерами могут являться визуальный поиск требующегося объекта, распознавание изображения, чтение надписи, выбор элемента интерфейса, вспоминание и т. д. Физические действия пользователя связаны с выполнением движений, например, перемещение курсора мыши, нажатие клавиши на клавиатуре, перемещение рук с мыши на клавиатуру и т. д.

В настоящее время взаимодействие пользователя с ГПИ в подавляющем большинстве случаев реализуется с помощью указательных координатных устройств с относительным указанием позиции (перемещения) курсора — мыши, трекбола, трекпоинта, тачпада и т. д. Интерактивность ГПИ проявляется в учете позиции курсора на экране, соответствующей области фокусировки внимания пользователя и конкретному выбранному объекту (например, тексту, таблице, рисунку, графику и т. д.), для которого предусмотрены команды и свойства, реализуемые с помощью определенного набора элементов интерфейса. Следует заметить, что многие объекты могут состоять из совокупности элементов, для каждого из которых также могут быть предусмотрены отдельные команды и свойства. Это обусловливает сложность работы пользователя с такими объектами и, как следствие, — необходимость реализации наиболее простых вариантов обеспечения доступа к соответствующим элементам ГПИ.

Наиболее эффективным техническим решением, используемым многими производителями ПО для обеспечения его интерактивности, является реализация в ГПИ контекстно-зависимых элементов управления, отображающих команды, соответствующие выбранному объекту, и средства изменения его свойств. Наиболее известными примерами элементов данного вида являются [3]:

- контекстные всплывающие меню — списки типовых команд, вызываемые в текущем положении курсора на выбранном объекте;
- контекстные панели — наборы инструментов, соответствующие выбранному объекту и отображаемые в определенной части окна программы;
- смарт-теги — списки дополнительных команд, реализующих опции, предусмотренные разработчиком ПО или заданные пользователем для текстовых фрагментов определенного содержания.

Использование контекстно-зависимых элементов управления позволяет повысить интерактивность ПО, упростить восприятие его ГПИ пользователем и сократить сложность требующихся от него действий с указательным координатным устройством и клавиатурой. Показателем, характеризующим эффективность ГПИ, являются затраты времени, требующиеся пользователю для решения с его помощью конкретных задач и связанные с необходимостью выполнения последовательностей ментальных и физических действий.

В статье изложено описание модели ContextTag (контекстная метка), представляющей результаты развития существующих контекстно-зависимых элементов ГПИ и расширения сферы их применения, на примере работы с таблицами в текстовом редакторе MS Word. Теоретические исследования, явившиеся предпосылками данной разработки, изложены в [4, 5]. Представляемая модель не является дополнением к редактору MS Word, но позволяет оценить удобства работы пользователя, обеспеченные предложенными техническими решениями, и сделать выводы о целесообразности реализации ее функциональных возможностей в интерактивном ПО.

Основными новшествами и преимуществами модели ContextTag являются:

- элемент интерфейса "контекстная метка", отображаемый для выбранного пользователем объекта с учетом координат указателя курсора и обеспечивающий возможность вызова контекстно-зависимых панелей инструментов;
- контекстно-зависимые панели инструментов, отображаемые относительно элемента "контекстная метка" и обеспечивающие доступ пользователя к свойствам и командам выбранного объекта;
- контекстное отображение дополнительных объектов (меню, окон и т. д.) относительно соответствующих элементов контекстно-зависимых панелей;
- опции настройки отображения элемента "контекстная метка", контекстно-зависимых панелей, а также дополнительных объектов, позволяющие продемонстрировать и исследовать особенности организации работы пользователя с контекстно-зависимыми элементами управления.

Описание функциональных возможностей модели

Модель ContextTag интегрирована в модель WordModel [6] усовершенствованного интерфейса текстового редактора MS Word, главное окно которой представлено на рис. 1 (см. вторую сторону обложки). Ключевым элементом, реализующим основные функциональные возможности модели ContextTag, является элемент интерфейса "контекстная метка", отображаемый для объекта "ячейки таблицы", выделенного на рабочем листе с помощью указателя курсора или клавиш клавиатуры.

Таблица является показательным примером сложного объекта, состоящего из совокупности отдельных элементов (ячеек), каждому из которых соответствует определенный доступный пользователю набор свойств и команд, например, оформление границ и заливка, изменение ширины столбцов и высоты строк, добавление, удаление и объединение ячеек и т. д.

В большинстве ГПИ ПО работа пользователя с объектами организована с помощью главного и контекстных меню. Данный способ организации доступа к свойствам и командам выбранного объекта связан с существенными затратами времени пользователя, так как требует от него выполнения нескольких перемещений курсора и внимания на значительные расстояния [6]. Минимизация затрат времени пользователя и, как следствие, сложности требующихся от него действий, может быть достигнута в результате поиска наилучших вариантов группировки и геометрического размещения элементов интерфейса, обеспечивающих доступ к командам и свойствам объектов. Наилучшим вариантом размещения элементов управления будет являться их отображение в непосредственной близости от координат текущего положения указателя курсора.

В модели ContextTag на примере работы с таблицами реализован принцип контекстного интерактивного взаимодействия ГПИ с пользователем. В результате выделения ячеек (выбора объекта) в непосредственной близи от текущего положения курсора отображается элемент интерфейса "контекстная метка" (далее — "метка"), сигнализирующий пользователю о наличии доступных ему свойств и команд. При воздействии на "метку" рядом с ней отображаются контекстно-зависимые панели инструментов (далее — панели инструментов),

содержащие элементы управления, обеспечивающие работу пользователя со свойствами выбранного объекта и выполнение соответствующих ему команд.

"Метка" и панели инструментов располагаются таким образом, что не перекрывают область выделенных ячеек таблицы. На рис. 2 (см. вторую сторону обложки) проиллюстрированы примеры вариантов отображения "метки" и панелей инструментов в зависимости от направления перемещения указателя курсора в процессе выделения ячеек. При работе с объектами, не требующими подобного выделения входящих в их состав элементов, координаты отображения "метки" могут определяться на основании координат указателя курсора в момент выбора объекта.

Элементы управления на панелях инструментов представлены кнопками с пиктограммами, иллюстрирующими их функциональные назначения. Отсутствие текстовых надписей позволяет существенно сократить размер элементов интерфейса, а использование интуитивно понятных пиктограмм ориентировано на пользователей, знакомых с функциональным назначением соответствующих кнопок. Для начинающих пользователей предусмотрена система контекстных подсказок, отображаемых при задержке указателя курсора на элементах управления, примеры представлены на рис. 2.

Каждому элементу панелей инструментов может соответствовать определенная выполняемая в результате воздействия на него команда или дополнительный объект (меню, панель, окно и т. д.), обеспечивающий доступ к конкретному набору команд и свойств. В целях минимизации затрат времени пользователя координаты отображения дополнительных объектов определяются относительно координат соответствующих им элементов панелей инструментов, что обеспечивает их контекстную связь. Если для элемента панели инструментов предусмотрен набор последовательно вызываемых дополнительных объектов, то координаты отображения каждого следующего объекта определяются относительно предыдущего, что также обеспечивает их контекстную связь.

На рис. 3 представлен пример контекстной связи объектов: меню отображено относительно соответствующего элемента панели инструментов, а дополнительное окно — относительно соответствующего пункта меню.

Контекстная зависимость отображения "метки" относительно положения (и перемещения) указателя курсора, панелей инструментов относительно "метки", а также дополнительных объектов относительно друг друга и соответствующих элементов панелей инструментов позволяет минимизировать число и сложность требующихся от пользователя действий — перемещений внимания и курсора. Это обеспечивает минимизацию затрат времени пользователя и максимизацию эффективности применения контекстно-зависимых элементов управления в задачах организации человеко-компьютерного взаимодействия. Контекстный принцип позиционирования меню, окон и панелей [6] позволяет повысить эргономичность ГПИ и, как следствие, конкурентоспособность ПО.

В модели ContextTag реализованы основные опции работы с таблицами, аналогичные предусмотренным в текстовом редакторе MS Word, соответствующие элементы панелей инструментов пронумерованы на рис. 3:

1. Автоматический подбор ширины столбцов и высоты строк.
2. Выделение таблицы, столбцов или строк.
3. Добавление таблицы (дополнительное окно), столбцов (слева или справа), строк (выше или ниже) или ячеек (дополнительное окно).
4. Удаление таблицы, столбцов, строк или ячеек (дополнительное окно).
5. Преобразование текста в таблицу или таблицы в текст.
6. Разбиение ячеек (дополнительное окно) или таблицы.
7. Объединение выделенных ячеек.
8. Оформление границ и заливки выделенных ячеек (дополнительное окно).

Функциональность модели ContextTag ориентирована на повышение эргономичности работы пользователя с указательным координатным устройством и клавиатурой, в том числе в случае комбинирования данных средств манипулирования. Выделение ячеек таблицы может быть выполнено с помощью курсора (например, мышью), либо клавиш с указателями стрелок при нажатой клавише Shift. Вызов и закрытие панелей инструментов и дополнительных объектов осуществляется воздействиями на "метку" и кнопки с пиктограммами, а также

Рис. 3. Пример контекстной связи элемента управления, меню и дополнительного окна

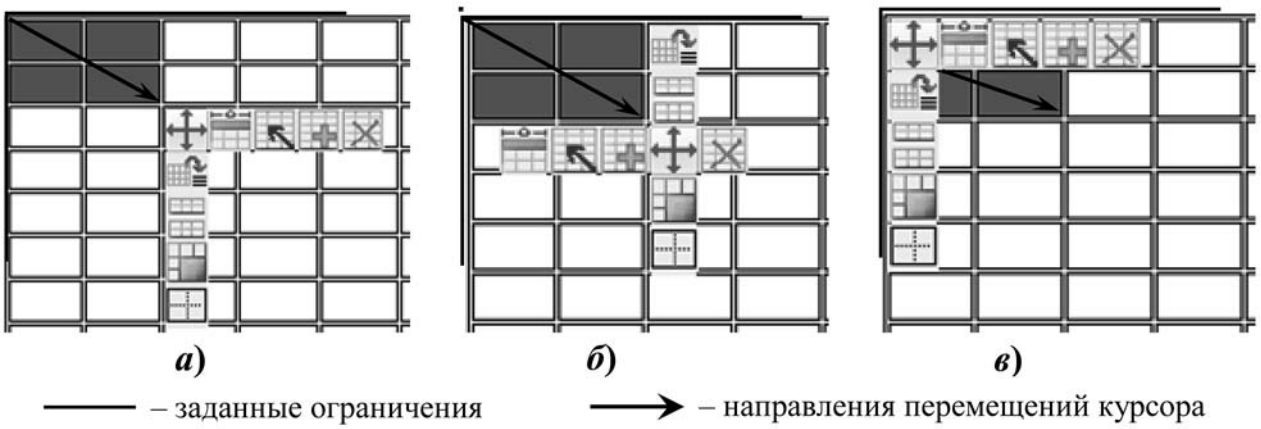


Рис. 4. Примеры перестроения панелей инструментов при ограничениях слева и сверху

нажатиями клавиш Enter и Esc. Перемещение фокуса (выделения) между "меткой", элементами панелей инструментов, а также между элементами в дополнительных объектах может быть выполнено с помощью указателя курсора, а также с помощью клавиш с указателями стрелок.

В настройках модели ContextTag предусмотрена опция, позволяющая вызывать панели инструментов и дополнительные объекты не воздействиями на "метку" и соответствующие кнопки с пиктограммами, а наведениями курсора с определенной задержкой. Также предусмотрена возможность отображения панелей инструментов одновременно с элементом "метка".

В настройках модели реализованы опции включения режимов имитации ограничений на области возможного отображения панелей инструментов и дополнительных объектов. Для панелей инструментов предусмотрена возможность установки ограничений на размещение в пределах области, занятой таблицей, в соответствии с ее левой, правой, нижней и верхней границами отдельно и в любых сочетаниях. Для дополнительных объектов, соответствующих элементам панелей инструментов, область возможного отображения может быть ограничена рабочим листом в главном окне модели (см. рис. 1 на второй стороне обложки).

Режимы имитации ограничений демонстрируют возможности и особенности организации работы с контекстно-зависимыми элементами управления в условиях ограниченной области их возможного отображения в окне программы. При включении соответствующего режима панели инструментов модели ContextTag автоматически перестраиваются с учетом заданных ограничений одним из следующих способов, определяемых настройками:

1. Перенос панелей инструментов (рис. 4, *а*).

2. Перегруппировка элементов панелей инструментов (рис. 4, *б*).

3. Перенос элемента "метка" (рис. 4, *в*).

Настройки модели представляют собой опции дополнительного управления ее функциональными возможностями, повышающие удобство эксплуатации и позволяющие пользователю реализовать свои предпочтения.

Вызвать окно настроек можно одним из следующих способов:

- выбором пункта "Настройки" главного меню (см. рис. 1);
- нажатием горячей клавиши F10.

Настройки модели ContextTag расположены на закладке "Таблица" (рис. 5, см. вторую сторону обложки).

На рис. 5 отмечены следующие блоки настроек модели ContextTag:

1. Включение режима имитации ограничений на область размещения дополнительных объектов пределами рабочего листа в основном окне модели.

2. Включение режима имитации ограничений на область размещения панелей инструментов в соответствии с левой, правой, верхней и нижней границами таблицы отдельно и в любых сочетаниях.

3. Выбор варианта автоматического перестроения панелей инструментов в режиме имитации ограничений на область их возможного размещения.

4. Выбор варианта отображения элемента "метка" относительно границы диапазона выделенных ячеек.

5. Выбор варианта действия, обеспечивающего вызов панелей инструментов и дополнительных объектов с помощью указателя курсора.

6. Включение режима отображения панелей инструментов одновременно с элементом "метка".

Существующие аналоги и обзор примеров реализации в ГПИ контекстно-зависимых элементов управления

В данном разделе представлены результаты сравнения функциональных возможностей "метки" и контекстно-зависимых панелей инструментов, реализованных в модели ContextTag, с наиболее близ-

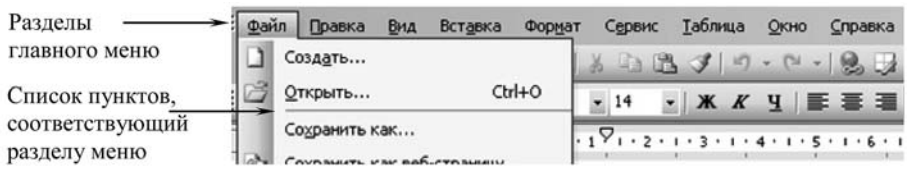


Рис. 6. Главное меню текстового редактора MS Word 2003

кими аналогами, а также рассмотрены примеры существующих контекстно-зависимых элементов управления: контекстное главное меню, контекстное всплывающее меню, контекстные панели инструментов и смарт-теги. Как показывает представленный материал, модель ContextTag обладает достаточной функциональностью и эргономичностью, а также является оригинальным результатом развития и совершенствования элементов управления, повышающих интерактивность ГПИ.

Контекстные главные меню. На рис. 6 представлен пример типового главного меню, реализованного в текстовом редакторе MS Word 2003. Данный элемент интерфейса присутствует в подавляющем большинстве ГПИ интерактивного ПО и предназначен для тематической группировки опций программы. Состав разделов типового главного меню статичен — не изменяется в соответствии с текущей задачей пользователя, определяемой, например, выбранным им объектом.

В настоящее время в некоторых приложениях реализуется контекстное главное меню, состав и содержание разделов которого зависят от текущей выполняемой пользователем задачи и могут соответствовать командам и свойствам выбранного объекта. На практике возможны два принципа реализации функционирования контекстного главного меню:

- разделы меню и пункты списков, не соответствующие текущей задаче пользователя или выбранному объекту (контексту), отображаются, но автоматически блокируются — являются недоступными для выбора. Основной недостаток данного подхода связан с тем, что наличие заблокированных разделов и пунктов меню рассеивает внимание пользователя;
- в меню отображаются только доступные для выбора разделы и пункты списков — состав меню реорганизуется в соответствии с контекстом. Основной недостаток данного подхода связан с тем, что изменения в составе разделов и пунктов меню могут вводить пользователя в заблуждение.

Контекстное главное меню не может рассматриваться в качестве аналога модели ContextTag, так как является техническим решением, ориентированным на упрощение доступа пользователя к функциональным возможностям ПО в соответствии с текущим контекстом, но без учета позиции курсора.

Контекстные всплывающие меню. Контекстное всплывающее меню — элемент ГПИ, отображаемый в текущем положении курсора в окне программы при нажатии кнопки указательного координатного устройства или специальной клавиши на клавиатуре и содержащий список типовых команд, соответствующих объекту (контексту), находящемуся под курсором. Примеры типового контекстного всплывающего меню текстового редактора MS Word и модифицированного меню, реализованного в модели WordModel [6], представлены на рис. 7.

Вызов дочернего окна с помощью типового меню приводит к его закрытию, в случае использования модифицированного меню дочерние окна отображаются напротив соответствующих пунктов. Основным недостатком рассматриваемого элемента ГПИ является большой размер, определяемый наименованиями отображаемых в нем пунктов и их представлением в виде списка. Следствием данного недостатка является существенное усложнение требующихся от пользователя действий — при большом числе пунктов меню существенно возрастают расстояния перемещения курсора и внимания.

Модель ContextTag может рассматриваться как более эффективный аналог типового и модифицированного контекстного всплывающего меню, позволяющий исключить недостаток, связанный с большим размером данного элемента ГПИ, и минимизировать сложность требующихся от пользователя действий. В то же время модель ContextTag и контекстное всплывающее меню могут использоваться совместно, разделяя свойства и команды объекта и являясь взаимодополняющими элементами управления.

Рис. 7. Примеры типового и модифицированного всплывающего меню

Контекстные панели инструментов. Наиболее эффективным и широко распространенным техническим решением, позволяющим повысить интерактивность ГПИ современного ПО, являются контекстные панели инструментов, содержащие элементы интерфейса, обеспечивающие выполнение команд и изменение свойств объекта, выбранного пользователем в текущий момент времени. Состав элементов интерфейса на панелях автоматически изменяется в соответствии с типом и особенностями выбранного объекта. Данные элементы управления реализованы в настоящее время в подавляющем большинстве последних версий ПО, ориентированного на работу со сложными объектами, например, двухмерной и трехмерной графики. Примерами соответствующего ПО могут являться Adobe Photoshop, AutoCad, ArchiCAD, SolidWorks, CorelDRAW, Autodesk Combustion и т. д.

Наглядным примером результатов внедрения данных элементов управления является ГПИ пакета программ Microsoft Office 2007 и 2010, реализующий новую концепцию организации человеко-компьютерного взаимодействия. Согласно основной идеи данной концепции, ГПИ должен отображать только те элементы управления, которые имеют непосредственное отношение к задаче, выполняемой пользователем в текущий момент времени. Переход пользователя между задачами, например, при работе с таблицами и текстом, приводит к изменению состава соответствующих отображаемых в ГПИ элементов управления.

Для реализации данных возможностей разработчикам Microsoft Office потребовалось отказаться от стандартных атрибутов оконного интерфейса — главного меню и панели инструментов. Их аналогами явились *Лента* (Ribbon), *Кнопка* "Office" (Office Button) и *Панель быстрого доступа* (Quick Access Toolbar). На рис. 8 (см. вторую сторону обложки) представлена верхняя часть окна приложения MS Word 2007, содержащая перечисленные элементы ГПИ, основной из которых — *Лента*.

Визуально *Лента* представляет собой результат объединения главного меню и расширенной панели инструментов. Данный элемент управления содержит набор тематических вкладок, каждая из которых отображает инструменты управления отдельным аспектом работы программы. Инструменты на вкладках объединены в логические группы, каждая из которых соответствует определенному набору команд. В каждый текущий момент времени активной (выбранной) является только одна вкладка. Результатом выполнения некоторых команд (воздействий на соответствующие элементы интерфейса *Ленты*) может являться контекстное отображение групп инструментов.

Некоторые вкладки *Ленты* являются контекстно-зависимыми относительно выполняемых пользователем задач и отображаются в том случае, если он перешел к выполнению соответствующей задачи. Контекстно-зависимые вкладки объединены в группы, каждая из которых содержит инструменты, предусмотренные для работы с конкретным видом объектов (для рисунков — *Работа с рисунками* (Picture Tools), для диаграмм — *Работа с диаграммами* (Chart Tools) и т. д.). На рис. 9 представлен пример вкладки *Формат* (Format), отображаемой при работе с рисунком в приложении MS Word 2007 и входящей в состав группы контекстных вкладок *Работа с рисунками*.

Основным преимуществом и особенностью контекстной панели инструментов является возможность отображения различного состава элементов интерфейса, соответствующего текущей задаче пользователя, на ограниченной площади окна программы. Ее основными недостатками являются:

- достаточно большой размер, ограничивающий возможности отображения информации в окне программы;
- необходимость перемещения курсора и внимания пользователя между выбранным в окне программы объектом и элементами ГПИ на панели, являющаяся следствием ее статичного расположения;
- отсутствие организации контекстной работы с дочерними окнами — вызываемые окна отображаются в том месте окна программы, где пользователь разместил их в предыдущем сеансе работы. Это не обеспечивает минимума сложности требующихся от пользователя действий;
- целесообразность применения при достаточно большом числе элементов интерфейса, соответствующих текущей задаче пользователя.

Помимо *Ленты* примером контекстных панелей инструментов в Microsoft Office 2007 и 2010 является элемент управления, содержащий опции форматирования текста (рис. 10). Данная панель инст-

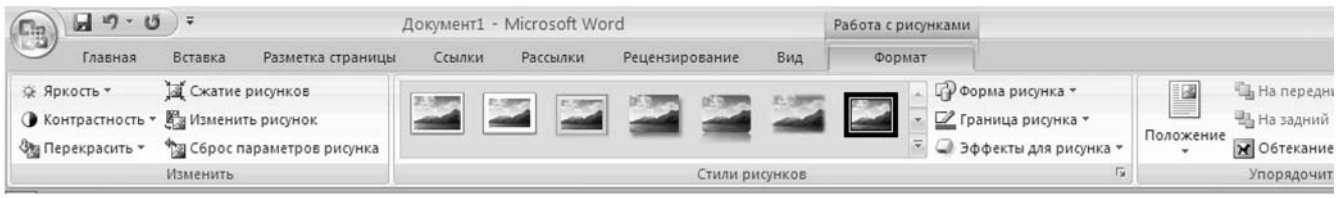


Рис. 9. Контекстная панель инструментов в ГПИ MS Word 2007

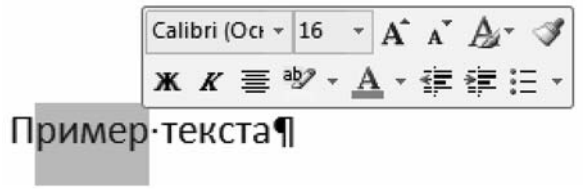


Рис. 10. Контекстная панель инструментов форматирования текста

рументов отображается при выделении фрагмента текста с учетом координат текущего положения курсора.

Основным отличием представленного элемента управления от модели ContextTag является отсутствие организации работы с дополнительными объектами (меню, панелями, окнами и т. д.), обеспечивающими доступ пользователя к свойствам и командам выбранного объекта. В результате данный элемент управления может объединять малое число опций программы, что ограничивает возможности его прикладного применения.

На основании представленного материала можно сделать вывод, что контекстно-зависимые панели модели ContextTag и контекстные панели инструментов целесообразно рассматривать не как конкурирующие, а как взаимодополняющие технические решения.

Смарт-теги. Смарт-тег (smart-tag, интеллектуальная метка) — это программная технология, состоящая из интерфейсной и программной частей, позволяющая подключать к приложению дополнительные модули и реализовывать дополнительные функциональные возможности. Корпорация Microsoft впервые применила данный термин в 2001 г. для обозначения специального механизма контекстно-зависимого исполнения действий на основе ассоциативной связи с текстовыми фрагментами. Данная концепция, являющаяся, по сути, результатом совмещения гипертекста и контекстного всплывающего меню, реализована в пакете Office версий 2003, 2007 и 2010. Пример смарт-тега представлен на рис. 11.

В документах пакета Microsoft Office применение смарт-тегов оправдано в том случае, когда

Рис. 11. Пример смарт-тега в пакете MS Office

есть необходимость инициирования контекстно-зависимых действий или реализации интерактивного взаимодействия с пользователем на основе данных о том, какую часть документа он редактирует. Смарт-теги представляют собой технологию, расширяющую информационное наполнение документов Microsoft Office. Эта технология позволяет связывать релевантную информацию (хранящуюся в документе) с ресурсами, из которых можно получить дополнительные данные, что удобно при создании и заполнении документа.

Принципы организации работы со смарт-тегами и контекстно-зависимыми панелями, реализованными в модели ContextTag, во многом схожи — оба технических решения позволяют пользователю получить доступ к опциям, предусмотренным для текущего контекста. В случае смарт-тегов контекстом являются текстовые фрагменты, в то время как для модели ContextTag им может быть фактически любой выбранный пользователем объект. Это позволяет утверждать, что оба технических решения являются результатом развития элементов управления ГПИ, ориентированных на повышение эффективности доступа пользователя к опциям ПО в целях обеспечения интерактивности интерфейса, а также сокращения числа и сложности требующихся от оператора действий. В то же время можно утверждать, что фактически смарт-теги являются частным случаем более универсальной "контекстной метки", реализованной в модели ContextTag. Данное утверждение основывается на том, что в общем случае понятие контекста модели ContextTag существенно шире, чем аналогичное понятие для смарт-тегов.

Заключение

Модель ContextTag является конкурентоспособной оригинальной разработкой, соответствующей передовому уровню организации человеко-компьютерного взаимодействия. Предложенные в ней технические решения позволяют повысить интерактивность ГПИ современного ПО, сократить сложность требующихся от пользователя действий и реализовать возможность эффективного доступа к опциям ПО в соответствии с выполняемой задачей.

Представленная разработка является, по сути, результатом развития идеи смарт-тегов и может рассматриваться как взаимодополняющее техническое решение относительно контекстных панелей инструментов, являющихся передовой разработкой, реализованной, в частности, корпорацией Microsoft в ГПИ пакетов Microsoft

Office 2007 и 2010. Это обосновывает перспективность применения контекстно-зависимых панелей, подобных реализованным в модели ContextTag, в современном интерактивном ПО.

Представленная модель является прототипом, позволяющим оценить преимущества реализованных в ней технических решений, а также обосновать целесообразность их внедрения в прикладное программное обеспечение. Данная разработка может рассматриваться в качестве паттерна дизайна взаимодействия (interaction design) — универсального решения проблемы, часто повторяющейся в дизайне ГПИ или организации взаимодействия пользователя с ним. Предложенные способы отображения и перестроения панелей инструментов модели ContextTag не претендуют на универсальность и могут быть адаптированы в соответствии с особенностями конкретной прикладной задачи организации человеко-компьютерного взаимодействия.

Модель ContextTag обладает теоретической и практической ценностью, так как расширяет представление об особенностях работы пользователей с ГПИ и может быть реализована в большом числе программных продуктов.

Список литературы

1. **Мандел Т.** Дизайн интерфейсов: Пер. с англ. М.: ДМК Пресс, 2005. 416 с.
2. **Раскин Д.** Интерфейс: новые направления в проектировании компьютерных систем: Пер. с англ. СПб.: Символ-Плюс, 2005. 272 с.
3. **Купер А., Рейман Р., Кронин Д.** Алан Купер об интерфейсе. Основы проектирования взаимодействия: Пер. с англ. СПб.: Символ-Плюс, 2009. 688 с.
4. **Зуев А. С.** Математическое и программное обеспечение средств проектирования и совершенствования интерактивных графических человеко-машинных интерфейсов [Текст]: дис. ... канд. техн. наук / А. С. Зуев; Московский гос. ун-т приборостроения и информатики. Москва, 2006. 125 с.
5. **Зуев А. С.** Подход к разработке и модернизации структур интерфейсов компьютерных программ // Информационные технологии. 2007. № 1. С. 55—62.
6. **Зуев А. С., Кучеров О. Б.** Модификация принципов работы с дочерними окнами программ, панелями инструментов, главными и контекстными меню // Информационные технологии. 2009. № 4. С. 50—55.

УДК 004.4’27

С. А. Шейпак, аспирант,
В. В. Шилов, канд. техн. наук, зав. кафедрой,
"МАТМ"—РГТУ им. К. Э. Циолковского
e-mail: serega.sheypak@gmail.com

Архитектура распределенного сервиса создания цифровой графики и анимации для Интернет-систем

Рассматривается разработанное архитектурное решение распределенной системы, предназначенной для создания графики и анимации.

Ключевые слова: *распределенная система, графика, анимация*

В развитии индустрии разработки программного обеспечения (ПО), предназначенного для массового использования, можно выделить следующие наиболее критичные проблемы:

1. Защита интеллектуальной собственности.
2. Поддержка программной разработки, выведенной в массовую эксплуатацию.
3. Развитие программной разработки.

Все эти проблемы актуальны и при разработке ПО, предназначенного для создания цифровой графики и анимации, однако к ним добавляется также проблема, специфичная для конкретной области.

4. Совместимость форматов.

Решения указанных выше проблем возможно в рамках предлагаемой клиент-серверной архитектуры программного обеспечения, модель которой представлена на рис. 1. Она позволяет разделить

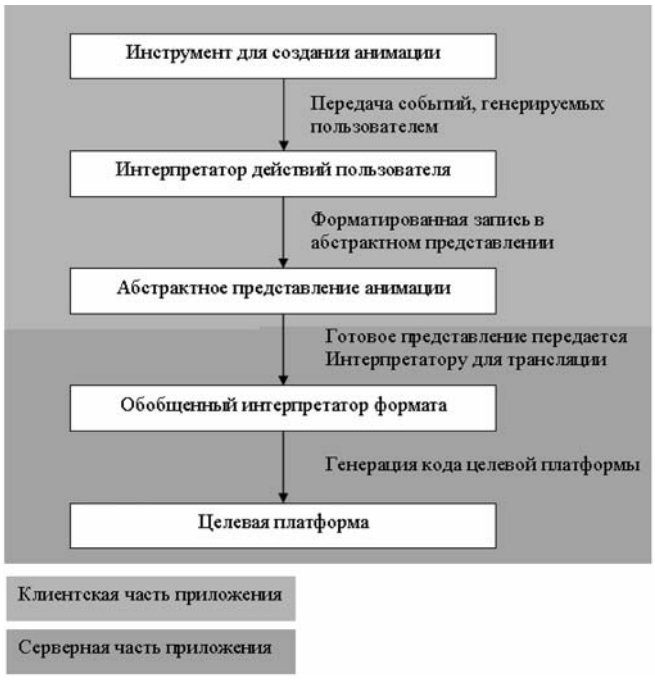


Рис. 1. Компонентная модель архитектурного решения

программное решение на два логических компонента — клиентский модуль, отвечающий за отображение данных пользователю, и серверный модуль, реализующий алгоритмы и логику, скрытые от пользователя. Типичным примером клиентского модуля являются веб-браузеры: Mozilla, Chrome, Internet Explorer, Opera и т. д. Браузеры только отображают данные, приходящие от веб-сервера (серверный компонент), такие как картинки, таблицы, текст. Какие картинки, какой текст отправлять браузеру — решает веб-сервер. Тонкие клиенты относительно недороги в разработке и служат своеобразным проводником между пользователем и серверным модулем, обрабатывающим пользовательские запросы и генерирующим ответ на действия пользователя. В предлагаемой модели наиболее ценные технологии перенесены на сервер, а клиентские приложения не имеют прямого доступа к ядру системы. Это обеспечивает защиту от несанкционированного доступа к сервисам и компонентам системы, являющимся интеллектуальной собственностью.

При росте объема кода, а также сложности и высоком темпе обновления реализуемого функционала программного обеспечения, крайне важных для коммерческого проекта, неизбежно растет число подлежащих исправлению ошибок. Проблема эффективной поддержки пользователей, эксплуатирующих программное решение, может быть разрешена посредством выделения ядра системы, ответственность за качество которого полностью ляжет на производителя. Система в этом случае будет представлять собой основу, платформу для создания дополнительных сервисов. Право разработки дополнений на основе ядра предоставляется сторонним разработчикам. Таким образом, производитель программного решения концентрирует усилия на платформе, а не на созданных на ее базе сервисах. Благодаря такому подходу из работы исключается непрофильная деятельность, что позволяет сэкономить ресурсы и предоставить пользователям качественный продукт.

Помимо поддержки и обслуживания ядра системы, необходимо постоянно дорабатывать и улучшать сервис. В предлагаемом подходе предполагается, что сторонним разработчикам предоставляются инструменты для создания платформенных сервисов. Такой подход известен как SaaS (Software as a Service — программное решение, как сервис). Яркими примерами SaaS являются Google AppEngine, платформа iPhone, Android. Во всех перечисленных примерах компания-разработчик ядра (операционной системы iPhone, серверной инфраструктурной оболочки AppEngine, операционной системы Android) предоставляет инструменты для создания дополнительных приложений и сервисов на базе своей платформы. С помощью защищенного сервера проводятся процедуры валидации используемых клиентом расширений (плагинов), таким образом, исключена возможность несанкционированного доступа к плагину со стороны пользователя. Защищенный сервер обеспечивает защиту интеллектуальной собственности сторонних разработчиков, создающих расширения для системы.

Разрешив общие проблемы, возникающие при разработке практически любой многопользовательской системы, необходимо разрешить проблемы, специфичные для конкретной области, — создание цифровой графики и анимации. В настоящее время существует множество форматов и технологий, позволяющих создать графическую или анимационную работу, отображаемую в веб-браузере пользователя. К числу наиболее популярных относятся технологии Flash, Silverlight, JavaFX, SVG, Canvas, анимация с использованием JavaScript и DOM. У каждой из них есть свои преимущества и недостатки, тем не менее крайне важно, что в результате использования любой из них получается одно и то же: графическая или анимационная работа, отображаемая в браузере пользователя.

На основе описанного архитектурного решения предлагается разработать универсальный сервис, позволяющий пользователям абстрагироваться от конкретного формата или технологии. Производитель программного обеспечения сервиса отвечает за поддержку и развитие ядра системы, в функционал которого включены возможности по трансляции создаваемой работы из одного формата в другой.

В модели, представленной на рис. 1, отражены ключевые узлы системы. С помощью защищенного сервера проводятся процедуры валидации используемых клиентом расширений (плагинов), а также трансляция графической работы из промежуточного формата в формат целевой платформы (Flash, SVG, JavaScript и т. д.). Следует отметить дополнительные свойства предлагаемого архитектурного решения, позволяющие эффективно вести коммерческую деятельность. В частности, в компонентной модели отражен сервис демонстрации работ пользователей. Сервис организован таким образом, что работа может быть показана, но несанкционированное использование работы исключено. При желании пользователь может приобрести понравившуюся работу в нужном ему формате. Кроме продаж работ пользователей, предусматривается выставка и продажа расширений, создаваемых сторонними разработчиками.

На рис. 2 представлен так называемый "workflow" — последовательность действий, выполняемых над анимационной работой, созданной пользователем и хранимой в промежуточном формате. Как упоминалось выше, пользователю предостав-

ляется специализированный инструмент создания анимации. В процессе работы пользователя формируется специализированное представление, особым свойством которого является способность быть транслированным в код целевой платформы. Так называемое абстрактное представление анимации, созданной на основе пользовательской работы, подвергается интерпретации.

В процессе интерпретации перед генерацией кода целевой платформы происходит анализ совместимости абстрактного представления и формата целевой платформы. Необходимость стадии анализа объясняется тем, что не все анимационные работы могут быть транслированы под любую целевую платформу. Это связано с принципиальными различиями как в устройстве платформ, так и в технологическом уровне их развития и доступном инструментарии. Например, с помощью JavaScript DOM Animation невозможно в полной мере отобразить сложную графическую анимацию, но это можно сделать при трансляции под JavaScript Canvas, так как Canvas — это специализированный объект в окне браузера, предназначенный для отображения графики, а JavaScript DOM — это имитация графики и анимации через элементы Document Object Model, являющиеся элементами HTML-разметки. После стадии анализа и устранения непереносимых деклараций происходит кодогенерация в формат целевой платформы.

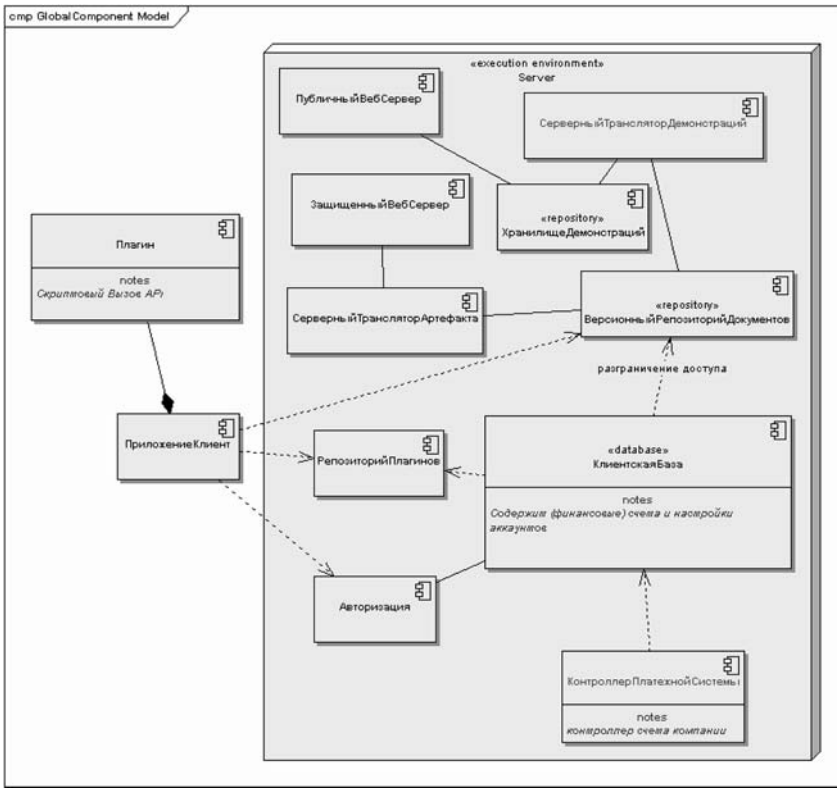


Рис. 2. Схема использования абстрактного представления анимации, применяемого для платформонезависимой демонстрации

Благодаря тому, что система разделена на независимые компоненты, повышается живучесть системы в целом. Отказ одного из компонентов не ведет к выходу из строя системы в целом. Например, сбой в работе репозитория плагинов не приводит к тому, что пользователи не могут продолжать работу, используя клиентское приложение, а процесс трансляции пользовательских работ, отправленных на сервер, не будет прерван. Кроме того, возможна как горизонтальная, так и вертикальная кластеризация компонентов, обозначенных на схеме. Четкое выделение технологического процесса обеспечивает возможность дублирования отдельных компонентов, испытывающих повышенную нагрузку. На основании анализа технологических рисков можно предположить, что в первую очередь потребуется кластеризация компонента "СерверныйТранслияторАртефакта", отвечающего за преобразование в целевой формат работ пользователей, находящихся в промежуточном представлении.

Стоит особо отметить, что предложенная модель архитектуры предлагает пользователям полный комплекс услуг, начиная со средства создания анимационной работы ("ПриложениеКлиент") и заканчивая интернет-площадкой, позволяющей демонстрировать и продавать готовые работы. Как было отмечено выше, такая интернет-площадка гарантирует защиту демонстрируемых работ.

Разработанная архитектура решения, приложенная к предметной области, позволяет решить ряд проблем, возникающих при разработке, поддержке, сопровождении и развитии программного продукта.

Список литературы

1. **Ahmed K. Z., Umrysh C. E.** Developing Enterprise Java Applications with J2EE and UML. Addison-Wesley, 2002. 330 p.
2. **Shannon B., Hapner M., Matena V.** et al. Java 2 Platform, Enterprise Edition: Platform and Component Specifications. Prentice Hall, 2000. 800 p.
3. **Metsker S. J.** Design Patterns Java Workbook. Addison-Wesley, 2002. 496 p.
4. **Шейпак С. А., Шилов В. В.** Концепция SaaS для создания цифровой анимации // Proc. 5th Central and East European Software Engineering Conference in Russia (CEE-SECR 2009). Moscow, 27—29 October 2009 // IEEE. 2009. P. 203—206. (IEEE Catalog Number CFP09CER-ART.)

УДК 501.519.2;531

Э. И. Владимирский,
канд. техн. наук, ст. науч. сотр,
e-mail: eduard.vladimirsky@hotmail.ru;
Б. И. Исмайлов, науч. сотр.
Азербайджанская государственная
нефтяная академия, г. Баку

Нелинейный рекуррентный анализ как математическая модель управления хаотическими процессами

Предлагается структура системы распознавания и управления хаотическими процессами, идеологической основой которых является нелинейный рекуррентный анализ. Парадигма управления продемонстрирована с использованием отображений Чирикова, Лози, Икеды. Получены рекуррентные диаграммы новых хаотических систем Чириков + Икеда, Лози + Икеда.

Ключевые слова: *хаотические процессы, нелинейный рекуррентный анализ, отображения Чирикова, Лози, Икеды*

Введение

Интенсификация и когерентность, в широком смысле, информационных потоков в сложных системах актуализируют проблему их оперативного анализа и принятия решений.

Если рассматривать инфраструктуру взаимодействия информации как открытую систему, то рост и интенсивность этих потоков приводит к усложнению информационной составляющей, и та, в свою очередь, вызывает нарастание хаотических процессов. Эволюционные траектории в пространстве состояний системы становятся чувствительными к малым информационным воздействиям (флуктуациям). В результате кумулятивного эффекта в определенный момент система может переходить с одной траектории эволюционного развития на другую. Так наступает точка бифуркации — точка ветвления вариантов развития. Ветвления, в свою очередь, катализируют инновационный всплеск, вызывающий интенсификацию различных форм взаимодействий информации.

Изложенное выше позволяет информационную среду определить как самоорганизующуюся коммуникативную систему. В комплексном фазовом пространстве, характеризующем поведение среды, выделяются состояния, которые притягивают к себе возможные траектории эволюции информационных систем. Эти состояния определяются как информационные аттракторы, по аналогии с понятием странный аттрактор в теории динамического хаоса.

Однако традиционные методы анализа нелинейных процессов требуют значительного объема информации, что, естественно, усложняет принятие корректного решения (прогноза).

Фундаментальную перспективу представляет использование метода рекуррентного анализа нелинейных процессов, не предъявляющего особых требований к данным и дающего удовлетворительные результаты [1, 2].

Нелинейный рекуррентный анализ катализирует реализацию такой важной проблемы, как управление динамикой хаотических систем (информационных процессов), т. е. позволяет посредством достаточно слабых воздействий переводить первоначально хаотические системы на требуемый динамический режим и тем самым стабилизировать их поведение [3].

Как известно, даже самое малое возмущение в сложной динамической системе может вызывать экспоненциальное отклонение от его предшествующего состояния, однако по прошествии некоторого времени система стремится вернуться в режим, достаточно близкий к предыдущему, или проходит через похожие ступени развития.

Рекуррентные графики визуализируют такое рекуррентное поведение динамической системы.

В связи с этим предлагается структура системы распознавания и управления хаотическими процессами, идеологической основой которых является нелинейный рекуррентный анализ.

Управление хаотическими динамическими системами. Общие положения

Пусть в метрической теории динамических систем тройка $\{M, S, \mu\}$ — пространство с мерой, где S — σ-алгебра подмножеств M и μ — мера, определенная тем или иным образом на S. Мера μ называется инвариантной мерой относительно преобразования T, если $\mu(C) = \mu(T^{-1}C)$ для любого $C \in S$.

Когда $\{t\}$ имеет дискретный ряд значений, $t \in Z$, $t \equiv k$, то $\{T_k\}$ называется динамической системой с дискретным временем или отображением [4].

Отображение, заданное на компактном множестве, может быть определено как хаотическое, если оно обладает чувствительной зависимостью от начальных условий и имеет плотные циклы. В дальнейшем для целей управления будут использованы отображения с хаотической динамикой.

Для управления поведением хаотических динамических систем в работе [4] рассмотрен метод так называемых резонансных возбуждений. Этот метод основан на наблюдении, что вследствие нелинейных модовых взаимодействий периодически возбуждаемая система не будет проявлять периодического поведения. Поэтому для получения предписанного (т. е. заранее заданного) режима движения представляется естественным возмущать систему специальным образом. Основную роль в данном методе играет допущение, что уравнение движения, на которое выходит система после введения возмущения, заранее известно.

Для достижения контроля посредством резонансных возбуждений в динамическую систему, находящуюся в хаотическом режиме, аддитивно включают внешнее возмущение $F(t)$:

$$\dot{x} = v(x, \alpha) + F(t), \alpha \in R, \quad (1)$$

где R — действительная числовая ось.

Далее, пусть требуемая динамика задается функцией $y(e)$, которая удовлетворяет так называемому уравнению предписанного движения

$$\dot{y} = g(y). \quad (2)$$

Теперь, выбирая возмущение в виде $F = g(y(t) - v(y(t), \alpha))$ и подставляя его в формулу (1), получим уравнение контролирования [4]:

$$\dot{x} = v(x, \alpha) + g(y) - v(y, \alpha). \quad (3)$$

Однако графическая визуализация фазовой траектории позволяет определить состояние системы без интегрирования уравнений, указывая, например, на периодическое или хаотическое поведение.

Прежде чем перейти к постановке задачи управления в терминах нелинейного рекуррентного анализа кратко остановимся на элементах теории [1, 2, 5].

Рекуррентный анализ

Фундаментальным свойством динамических диссипативных систем является рекуррентность (повторяемость) состояний в смысле прохождения последующей траектории достаточно близко к предыдущей. Это свойство было отмечено в 80-х гг. XIX века французским математиком Пуанкаре (Poincare) и сформировано в виде "теоремы рекуррентности" [2].

Теорема. Пусть T — сохраняющее меру преобразование пространства с конечной мерой (x, μ) и пусть $A \subset X$ — измеримое множество. Тогда для любого натурального $N \subset \mathrm{N}$

$$A(\{x \in A: \{T^{\mathrm{n}}(x)\}_{n \geqslant N} \subset (X \backslash A)\}) = 0,$$

где T — рекуррентное время; X — произвольное измеримое множество; $\mu(\cdot)$ — вероятностная мера; x — параметр нормализованного ряда; N — длина ряда; N — множество натуральных чисел.

Краткое доказательство теоремы рассмотрено в работе [2].

Однако возможности такого анализа сильно ограничены, поскольку, как правило, размерность фазового пространства сложной динамической системы более 3. В связи с этим в работе [1] был предложен способ отображения m-мерной фазовой траектории состояния системы $x(t)$ длиной N на двумерную квадратную двоичную матрицу размерностью $N \times N$, в которой 1 (черная точка) соответствует повторению состояния при некотором времени i в некоторое другое время j, а обе координатные оси являются осями времени. Такое представление было названо *рекуррентной диаграммой* (*recurrence plot*, *RP*) и формально выглядит как [1]:

$$R_{i,j}^{m, \varepsilon_i} = \theta(\varepsilon_i - \|x_i - x_j\|), x \in R^m, i, j = 1, ..., N, \quad (4)$$

где N — число рассматриваемых состояний x_i; ε_i — размер окрестности точки x в момент i, $\|\cdot\|$ — норма; $\tilde{\theta}(\cdot)$ — функция Хэвисайда.

Как отмечено в [5], невозможно обнаружить полную рекуррентность в смысле $x_i = x_j$ (состояния динамической, а особенно — хаотической системы не повторяются полностью эквивалентно начальному состоянию, а подходят к нему сколь угодно близко). Таким образом, *рекуррентность определяется как достаточная близость состояния x_j состоянию x_i*, т. е. рекуррентными являются состояния x_j, попадающие в m-мерную окрестность с радиусом ε_i и центром в x_i. Эти точки x_j называются *рекуррентными точками* (*recurrence plots*).

Так как $R_{i=j} = 1$ $(i = 1, ..., N)$, то рекуррентная диаграмма всегда содержит черную диагональную линию — *линию идентичности* (*line of identity*, *LOI*), под углом $\pi/4$ к осям координат [1].

Здесь важно отметить, что произвольно взятая рекуррентная точка (i, j) не несет какой-либо полезной информации о состояниях во времена i и j. Только вся совокупность рекуррентных точек позволяет восстановить свойства системы.

Далее рассмотрим адаптивную систему типа "измерение—распознавание—управление".

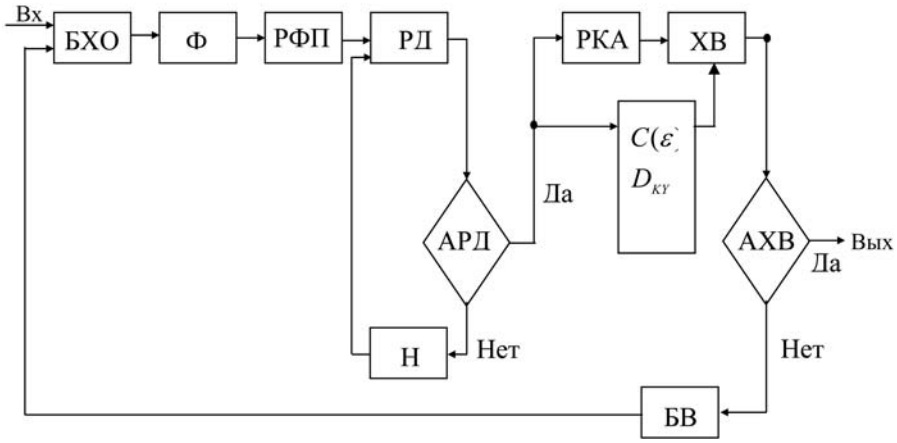


Рис. 1. Структура адаптивной системы измерение—распознавание—управление

Структура адаптивной системы измерение—распознавание—управление

Структура системы (рис. 1) состоит из следующих блоков:

- БХО — библиотека хаотических отображений;
- Ф — фильтр;
- РФП — блок реконструкции фазового пространства;
- РД — блок формирования рекуррентных диаграмм;
- РКА — блок количественного анализа рекуррентных диаграмм;
- ХВ — блок формирования характеристического вектора;
- АРД — блок анализа рекуррентных диаграмм;
- АХВ — блок анализа характеристического вектора;
- Н — блок норм (L_1, L_2, L_∞);
- БВ — блок выбора данных;
- блок определения корреляционного интеграла $C(\varepsilon)$, новой фрактальной размерности Каплана—Йорке D_{KY}, а также осуществляющий реализацию задач: фильтрации хаотической информации, рекуррентного анализа хаотических процессов и управления хаотическими процессами.

Постановка задачи

Пусть $x(t)|_1^N$, $y(t)|_1^N$ — m-мерные фазовые траектории состояний систем диной N, $\{x(t), y(t)\}_1^N \in R^m$.

Пусть $x(t)$, $y(t)$ соответствуют временные ряды $\{u_i\}_1^{\tilde{N}}$, $\{\upsilon_i\}_1^{\tilde{N}}$, где $u_i = u(t)$, $\upsilon_i = \upsilon(t_i)$, $t_i = i\Delta t$, $\tilde{N}$ — длина ряда (число наблюдений), Δt —интервал выборки.

Пусть заданы [6—8]:

Отображение Чирикова—Тейлора — получено при решении задачи об устойчивости движения заряженных частиц в магнитных ловушках. Представляет собой уравнение вида

$$x' = x - k\sin(y) \ (\mathrm{mod}\, 2 \cdot pi),$$
$$y' = y + k\sin(y) \ (\mathrm{mod}\, 2 \cdot pi),$$

где k — параметр, определяющий ширину эргодического слоя, отделяющего захваченные частицы от пролетных.

При $k = 0{,}9$ отображение имеет вид, представленный рис. 2, А (см. четвертую сторону обложки), фазовый портрет которого напоминает сэндвич с бесконечным числом чередующихся инвариантных кривых и стохастических слоев.

Красным, голубым, желтым и малиновым цветами отмечены области хаоса.

Отображение Икеда — возникает при моделировании оптических носителей информации и имеет вид

$$x_{n+1} = 1 + u(x_n \cos t_n - y_n \sin t_n),$$
$$y_{n+1} = 1 + u(x_n \cos t_n + y_n \sin t_n),$$

где u — параметр и

$$t_n = 0{,}4 - \frac{b}{1 + x_n^2 + y_n^2}.$$

Для $u = 0{,}9$ система имеет хаотический аттрактор (рис. 2, В).

Отображение Лозы — представляет уравнение вида

$$x_{n+1} = 1 - a|x_n| + y_n,$$
$$y_{n+1} = bx_n,$$

где $a = 1{,}4$, $b = 0{,}3$ имеет место отображение, представленное рис. 3, А (см. четвертую сторону обложки).

Характеризуется множеством неустойчивых периодических орбит хаотического аттрактора.

Пусть длины всех модельных рядов указанных отображений составляют $N = 5000$ точек, длины реальных рядов наблюдений различны. Пусть управление динамикой хаотических процессов будет трактоваться в терминах нелинейного рекуррентного анализа.

Требуется:

- реализовать рекуррентную диаграмму, характеризующую процесс управления методом резонансных возбуждений;
- реализовать рекуррентную диаграмму, характеризующую процесс управления таким образом, чтобы малые изменения параметров системы не выводили бы ее из окрестности, в которой она находилась;

- определить корреляционную размерность полученных новых отображений;
- определить новую фрактальную (ляпуновскую) размерность Каплана—Йорке.

Алгоритм реализации адаптивной системы измерение—распознавание—управление

1. Отображаем временные ряды $\{u\}_1^{\widetilde{N}}$, $\{\upsilon_i\}_1^{\widetilde{N}}$ соответствующих хаотических процессов на квадратные матрицы $[M_1, M_2] \in R^2$:

$$M_1:\{u_i\}_1^{\widetilde{N}} \Rightarrow [\widetilde{N} \times \widetilde{N}], \quad (5)$$

$$M_2:\{\upsilon_i\}_1^{N} \Rightarrow [\widetilde{N} \times \widetilde{N}]. \quad (6)$$

2. Определяем возмущение как

$$G : M_1 \to M_2. \quad (7)$$

3. Результирующее возмущенное отображение определяется как

$$M_G = M_1 + M_2,\ M_G \in R^2. \quad (8)$$

4. Рекуррентная диаграмма результирующего отображения M_G определяется как

$$M_G \Rightarrow R_{i,j}^{m,\varepsilon} = \theta(\varepsilon_i - \|x_i - x_j\|),\ x \in R^m,\ i, j = \overline{1, N}. \quad (9)$$

5. Определяется корреляционная размерность [2]:

$$C(\varepsilon) = \frac{1}{N^2} \sum_{\substack{i,j=1 \\ i \neq j}} \theta(\varepsilon - \|x_i - x_j\|).$$

6. Определяется размерность Каплана—Йорке [9]:

$$D_{KY} = 2 - \lambda_1/\lambda_2,$$

где λ_1 и λ_2 —старший и младший показатели экспонент.

7. Формируется хаотический вектор (ХВ)

$$XB \underset{def}{=} \Psi(M_G,\ C(\varepsilon),\ D_{KY}).$$

В случае получения неудовлетворительного результата в контексте принятия решений проводится корректировка по нормам Н или возмущающим воздействиям. В последнем случае используется интеллектуальный итерационный алгоритм с *p*-активными интервалами временных рядов, в котором последующее приближение на каждом шаге вычисляется только для части переменных, называемых активными.

Примеры визуализации алгоритма управления представлены на четвертой стороне обложки.

Анализ полученных результатов показал новые возможности в управлении динамическими системами, в контексте использования такого инструментария как нелинейный рекуррентный анализ. Кроме того, внешний вид рекуррентных диаграмм несет информацию о характере протекающих процессов (взаимодействии информационных потоков), наличии и влиянии шума, наличии состояний повторения и замирания (ламинарность), резких изменений и т. д.

Заключение

Использование ограниченного объема анализируемой информации и нетрадиционной визуализации в виде рекуррентных диаграмм позволяет, формируя новые структуры , повысить эффективность распознавания и управления в контексте принятия корректного решения.

Предложенная структура системы реализована в программной среде MatLab.

Список литературы

1. **Eckman J. P., Kamphorst S. O., Ruelle D.** Recurrence plots of dynamical system Europhys. 1987. V. 4. N 9. P. 973—977.

2. **Marvan N., Romano M. C., Thiel M., Kurths J.** Recurrence plots for the analysis of complex systems // Physics Reports. 2007. V. 438. P. 237—329.

3. **Владимирский Э. И., Исмайлов Б. И.** Синергетические аспекты нелинейного рекуррентного анализа хаотической информации // Тез. докл. Международной научно-практической конф. "Информационные технологии и компьютерная инженерия". Винница, Украина, 19—21 мая 2010 г. Винницкий Национальный Технический Университет, 2010. С. 96—97.

4. **Лоскутов А. Ю.** Проблемы нелинейной динамики. Ч. II. Подавление хаоса и управление динамическими системами // Вестник МГУ им. М. В. Ломоносова. Сер. 3. Физика. Астрономия. 2001. № 3. С. 3—19.

5. **Киселев В. Ю.** Разработка и исследование методов и алгоритмов построения математических моделей с использованием рекуррентных диаграмм. Автореф. дис. на соиск. ученой степени канд. технич. наук. На правах рукописи. СПб., 2009. 15 с.

6. **Морозов А. Д., Драгунов Т. Н.** Визуализация и анализ инвариантных множеств динамических систем. М.-Ижевск: Институт компьютерных исследований, 2003. 304 с.

7. **Кузнецов А. П., Савин А. В., Савин Д. В.** Особенности динамики почти консервативного отображения Икеды // Письма в ЖТФ. 2007. Т. 33. Вып. 3. С. 57—63.

8. **Sprott J. C**. Time-Series Analysis. Oxford University Press. 2003. 528 p.

9. **Chloverakis Konstantinos E., Sprott J. C.** A comparison of correlation and Lyapunov dimensions // Physica D. 2000. P. 156—164.

УДК 004.421

В. В. Мокшин, аспирант,
И. М. Якимов, канд. техн. наук, доц.,
Казанский государственный
технический университет им. А. Н. Туполева
e-mail: vladimir_kgtu@mail.ru

Метод формирования модели анализа сложной системы

Предлагается метод формирования модели исследования вероятностного объекта, являющегося сложной системой. Он основан на сочетании методов построения многофакторной нелинейной регрессионной модели, группового учета аргументов, параллельного генетического алгоритма отбора значимых признаков и искусственного интеллекта. Такой подход позволяет отобрать в модель наиболее значимые совокупности входных признаков (факторов). Рассматривается применение предлагаемого метода для формирования модели производственного предприятия.

Ключевые слова: *сложные системы, нелинейный регрессионный анализ, самоорганизация, параллельный генетический алгоритм, отбор значимых признаков*

Введение

В большинстве случаев для исследования сложных систем, таких как информационные системы, предприятия, территориальные регионы и т. д., поведение которых обусловливается различными входными воздействиями (признаками) и ответными реакциями, откликами (результативными показателями), используется концепция "черного ящика". Разработка методов формирования математических моделей при использовании концепции "черного ящика" является одной из важных кибернетических проблем [1, 2]. Кроме того, проблемой является и сам отбор признаков для представления ответных реакций в виде функциональных зависимостей вида

$$y_j = f_j(x_1, x_2, ..., x_M),\ j = \overline{1, K}, \qquad (1)$$

где K — число зависимых (результативных) признаков функционирования системы; M — число признаков, влияющих на функционирование системы; $(M + K)$ — общее число признаков.

Существующие итерационные методы прямого отбора, обратного исключения признаков и корреляционный метод имеют такие недостатки, как возможное включение в модель (1) незначимых признаков и значительное увеличение отношения стандартной ошибки к среднему при использовании тестовых значений входных признаков x_i, $i = \overline{1, M}$, для модели (1) [2, 3]. Использование генетических алгоритмов также может привести к включению незначимых признаков ввиду малого значения вероятности мутации [2—5].

Цель работы — разработка метода формирования модели анализа сложной системы, позволяющего включать в модель значимые признаки. Для достижения поставленной цели необходимо разработать метод отбора значимых признаков и метод формирования функциональных зависимостей вида (1).

1. Описание метода формирования модели анализа сложной системы

В данной работе предлагается использовать метод, в котором запускается несколько относительно коротких параллельных эволюционных путей отбора признаков. Определяется частота появления каждого признака относительно всех параллельных эволюционных путей. В работе алгоритма используется "ранняя остановка", как в нейронных сетях [3]. Это позволяет исключить незначимые признаки и уменьшить отношение стандартной ошибки к среднему при использовании проверочных значений входных признаков x_i, $i = \overline{1, M}$, для зависимостей (1) [2, 6]. Наиболее вероятно, что если признак действительно важен, то частота его появления на всех или большинстве эволюционных путей будет высокой. Если признак неважен, то частота его появления будет высокая только для некоторых эволюционных путей. В результате средняя частота появления признаков на всех параллельных эволюционных путях будет высока только для признаков, которые действительно важны. В ходе выделения значимых признаков определяется также необходимое число эволюционных путей.

После выделения совокупности важных признаков при формировании модели исследования сложной системы используется рекурсивно-регрессионная самоорганизация на основе метода группового учета аргументов (рис. 1), на этапе оптимизации и принятия решения — метод упорядоченного предпочтения по сходству с идеальным решением [7—10]. На рис. 1 схематически представлена реализация отбора значимых признаков и формирование регрессионного уравнения в ви-

де обобщенного полинома Колмогорова—Габора с учетом изменения системы во времени:

$$y = a_0 + \sum_{i=1}^{M} a_i x_i + \sum_{i=1}^{M}\sum_{j=i}^{M} a_{ij} x_i x_j + \sum_{i=1}^{M}\sum_{j=i}^{M}\sum_{k=j}^{M} a_{ijk} x_i x_j x_k + \ldots, \quad (2)$$

весовые коэффициенты a определяются по формуле

$$\mathbf{A}_j = (\mathbf{X}^T \cdot \mathbf{X})^{-1}(\mathbf{X}^T \cdot \mathbf{Y}_j),\ j = \overline{1, K},$$

где $\mathbf{A}_j$ — матрица весовых коэффициентов a j-й функции (2) признаков; $\mathbf{X}$ — матрица значений членов полинома (2); $\mathbf{Y}_j$ — матрица значений j-го результативного показателя y.

На первом этапе в блоке 1 (см. рис. 1) происходит сбор информации по признакам x_i, $i = \overline{1, M}$, и по откликам y_j, $j = \overline{1, K}$. После этого запускается параллельный генетический алгоритм отбора значимых признаков. Отбор значимых признаков осуществляется в блоках 1 и 2, состоящих из этапа настройки и этапа отбора значимых признаков. На этапе настройки происходит поиск наилучшего числа параллельных эволюционных путей B. Далее происходит отбор значимых признаков с помощью параллельного генетического алгоритма при заданном числе эволюционных путей B. После отбора значимых признаков запускается алгоритм формирования структуры регрессионного уравнения (2) с учетом изменения системы во времени. Его составляют следующие блоки: генерации моделей, оценки качества моделей, выбора моделей, анализа и контроля. Генерация модели продолжается до тех пор, пока не достигнуто минимальное значение критерия регулярности [7]:

$$\Delta^2 = \frac{1}{n_{\text{пров}}} \sum_{i=1}^{n_{\text{пров}}} [y - \hat{y}]^2, \quad (3)$$

где $n_{\text{пров}} = n - n_{\text{обуч}}$ — число временных наблюдений проверочной выборки признаков; y — действительное значение результативного показателя на i-м временном интервале проверочной выборки $n_{\text{пров}}$; $\hat{y}$ — выходное значение на i-м временном интервале проверочной выборки $n_{\text{пров}}$ в соответствии с моделью вида (2); $n_{\text{обуч}}$ — число временных наблюдений обучающей выборки признаков; n — общее число временных наблюдений признаков.

Алгоритм отбора значимых признаков и алгоритм формирования многомерного аддитивного ряда (2) представлены в п. 1.1 и в п. 1.2.

1.1. Параллельный генетический алгоритм отбора значимых признаков и определение наилучшего числа эволюционных путей

Идея отбора значимых признаков заключается в том, что вместо одного длинного эволюционного пути запускаются несколько относительно коротких параллельных эволюционных путей B [8]. Для каждого $b = \overline{1, B}$ запускается свой эволюционный путь отбора признаков с числом поколений N и размером популяции m. Пусть $P(b, t)$ — есть t-е поколение популяции на b-м эволюционном пути, где $t = \overline{1, N}$, $b = \overline{1, B}$. В результате необходимо определить частоту появления i-го входного признака на всех параллельных эволюционных путях $b = \overline{1, B}$.

Алгоритм отбора значимых признаков с числом параллельных эволюционных путей B.

Входные данные:

$\mathbf{Y}$ — матрица $n \times 1$ (временные наблюдения одного из $j = \overline{1, K}$ результативных показателей функционирования системы); $\mathbf{X}$ — матрица $n \times M$ (временные наблюдения признаков x_i, $i = \overline{1, M}$); n — число временных наблюдений признаков; m — размер популяции; N — число поколений; B — число параллельных эволюционных путей; v_t — вероятность мутации особи поколения t (по умолчанию $v_t = 1/M$).

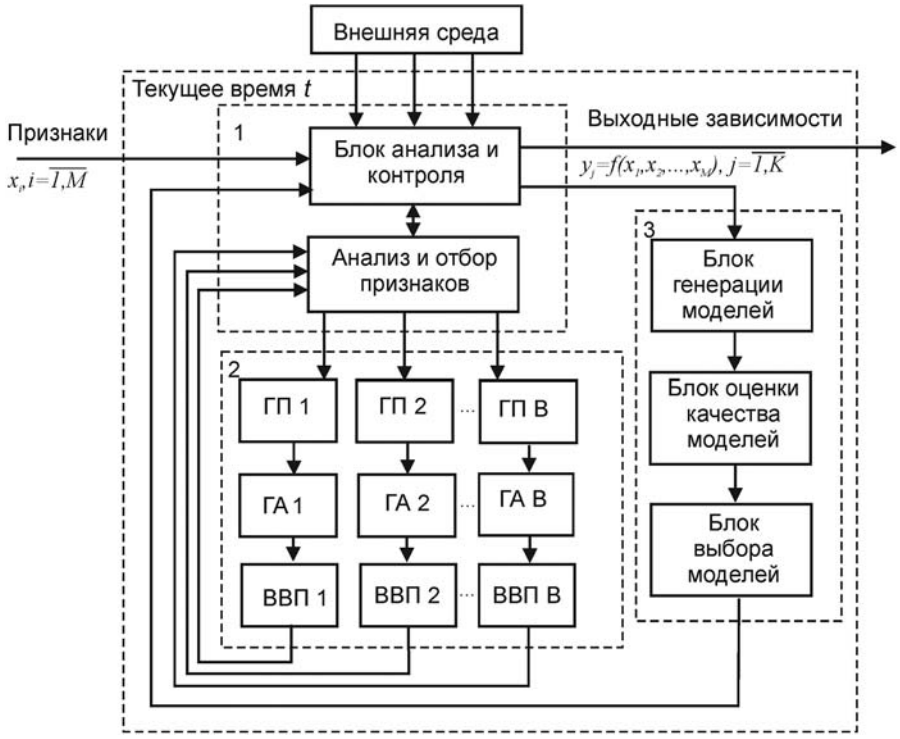


Рис. 1. Блок-схема формирования регрессионного уравнения с учетом изменения системы во времени (ГА — генетический алгоритм со стандартными генетическими операторами, ГП — генерация популяции, ВВП — вычисление весов признаков, B —число параллельных эволюционных путей)

Шаг 1. Установка числа параллельных эволюционных путей B.

Шаг 2. Для каждого параллельного эволюционного пути $b = \overline{1, B}$ выполняются шаги 3, 4.

Шаг 3. Пусть $P(b, N)$ — результат генетического алгоритма с параметрами ($\mathbf{Y}$, $\mathbf{X}$, m, N, v_t).

В качестве функции приспособленности в генетическом алгоритме используется критерий обобщенной перекрестной проверки [2]:

$$F(\omega) = $$
$$= -\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i(\omega)}{1 - \mathrm{tr}(\mathbf{H})/n}\right)^2 = -\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i(\omega)}{1 - (q+1)/n}\right)^2,$$

где $\mathbf{H} = \mathbf{X}_\omega(\mathbf{X}_\omega^T\mathbf{X}_\omega)^{-1}\mathbf{X}_\omega^T$, $\mathrm{tr}(\mathbf{H})$ — элементы главной диагонали матрицы $\mathbf{H}$, соответствующие $i = \overline{1, n}$, n — число временных наблюдений признаков; q — число входных признаков, используемых для описания результативного показателя y_j, $j = \overline{1, K}$; ω — особь, формируемое сочетание входных признаков, представленное в виде двоичной записи [4]. Здесь 1 соответствует отобранному фактору, а 0 — фактору, которым мы пренебрегаем.

Шаг 4. Вычисляется $r(i, b)$ — вес каждого признака x_i, $i = \overline{1, M}$, на b-м генетическом пути, который характеризует частоту его появления,

$$r(i, b) = \frac{1}{m}\sum_{p=1}^{m}\omega_p(i), \qquad (4)$$

где $\omega_p \in P(b, N)$, $p = \overline{1, m}$, $0 < r(i, b) < 1$.

Шаг 5. Определение частоты появления $\bar{r}_i$ входного признака i относительно всех параллельных эволюционных путей $b = \overline{1, B}$,

$$\bar{r}_i = \frac{1}{B}\sum_{b=1}^{B} r_i(i, b). \qquad (5)$$

Шаг 6. Сортировка частоты появления признаков матрицы $\bar{r}_i$ в порядке убывания и определение максимального расстояния d между частотами $\bar{r}_i$ и $\bar{r}_{i+1}$, $i = \overline{1, M-1}$.

Шаг 7. Вычисление $d_{\max}$, которое характеризует максимально допустимое расстояние для отбора признаков между упорядоченными в порядке убывания частотами появления признаков $\bar{r}_i$ и $\bar{r}_{i+1}$, $i = \overline{1, M-1}$,

$$d_{\max} = \alpha\sqrt{\frac{1}{4B}}, \qquad (6)$$

где $\alpha = 1{,}645$ — параметр, описанный в работе [2].

Шаг 8. Если выполняется условие

$$d \geqslant d_{\max}, \qquad (7)$$

то переходим к шагу 9, иначе все признаки будут являться значимыми.

Шаг 9. Среди отсортированных весов признаков матрицы $\bar{r}_i$ для дальнейшего исследования выбираются те признаки, веса которых находятся выше максимального расстояния d. Отобранные признаки включаются в массив $\bar{R}(i)$, где $i = \overline{1, M}$ (см. рис. 1). Если признак включается в модель, то в соответствующей позиции массива $\bar{R}(i)$ будет стоять 1, иначе — 0.

Выходные данные:

$\bar{R}(i)$ — массив отобранных признаков с числом параллельных эволюционных путей B.

Основная идея поиска оптимального числа параллельных эволюционных путей B заключается в следующем. Алгоритм отбора значимых признаков запускается U раз для каждого количества параллельных эволюционных путей $b = \overline{1, B}$. В результате определяется некоторый трехмерный массив частоты появления признаков $\bar{r}(M, B, U)$ для каждого входного признака x_i, $i = \overline{1, M}$ при каждом количестве параллельных эволюционных путей $b = \overline{1, B}$ с учетом повторных запусков алгоритма отбора признаков (рис. 2).

Например, при $B = 1$ и $U = 1$ получается массив частот $\bar{r}(M, 1, 1)$. Если этот алгоритм отбора признаков повторять U раз, то сформируется массив частот $\bar{r}(M, 1, U)$. Для каждого входного признака x_i, $i = \overline{1, M}$, из каждой выборки $\bar{r}(i, 1, U)$ определяется $\Delta r = r_{\max} - r_{\min}$ и формируется массив $\Delta\bar{r}$. После чего из получившегося массива $\Delta\bar{r}$ также определяются $\Delta r_{\min}$ и $\Delta r_{\max}$. Аналогичные рассуждения применяются для каждого количества параллельных эволюционных путей $b = \overline{1, B}$.

Наилучшим числом параллельных эволюционных путей будет считаться такое число B, при котором разброс частот появления признаков будет наименьшим:

$$\Delta r_{\max} - \Delta r_{\min} \to \min. \qquad (8)$$

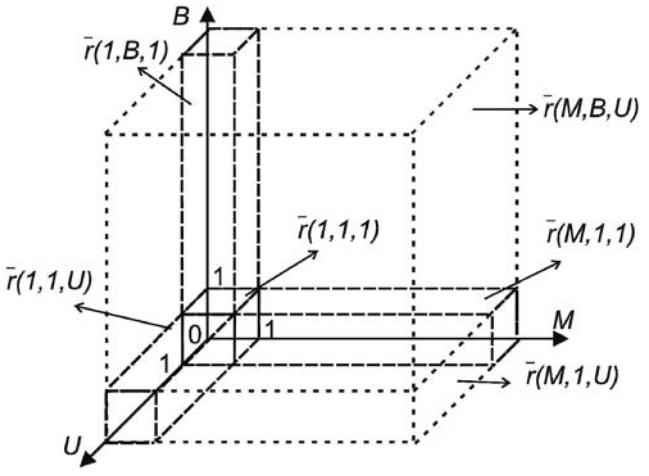


Рис. 2. Схематическое представление трехмерного массива весов входных признаков x_i, $i = \overline{1, M}$, для результативного показателя y

В случае, когда совокупность отобранных признаков становится стабильной, увеличение числа параллельных эволюционных путей прекращается.

Отобранные признаки и число параллельных эволюционных путей передаются в блок анализа и контроля. Здесь отбор признаков останавливается и запускается алгоритм формирования структуры многомерного регрессионного уравнения с учетом изменения системы во времени.

1.2. Формирование структуры регрессионного уравнения

Формирование регрессионного уравнения (2) методом группового учета аргументов можно описать в виде генетического алгоритма, используя стандартные генетические операторы. Описание популяции $P_2(t_2)$ с номером поколения t_2 можно представить в виде **G** — матрицы возможных степеней входных признаков, **A** — матрицы коэффициентов регрессионных уравнений претендентов и **W** — матрицы координат. Элементы матрицы **W** указывают на номера строк матрицы **G**. В этом случае каждая особь $\xi_i \in P_2(t_2)$ при $i = \overline{1, m_2}$ и $j = \overline{1, l}$ популяция размером m_2 содержит информацию о структуре регрессионного уравнения (2) для $y_j, j = \overline{1, K}$ (рис. 3) через l — число возможных членов регрессионного уравнения.

На рис. 3 a_{ij} — коэффициенты регрессионного уравнения, которые образуют матрицу **A**. Значения w_{ij} являются элементами матрицы **W**, а Δ^2 — критерий регулярности (3). Таким образом, модель-претендент или регрессионное уравнение (2) с учетом матриц **G**, **A**, **W** можно представить в виде

$$y_j = \sum_{i=0}^{l-1} a_{ji} \prod_{q=0}^{m-1} x_{q+1}^{g_{w_{j-1iq}}}, \; j = \overline{1, K}, \qquad (9)$$

где m — число входных признаков, $q = \overline{0, m-1}$, $g_{w_{j-1iq}}$ — элементы матрицы **G**; l — число членов регрессионного уравнения, определяемого в виде

$$l_M^p = 1 + \sum_{r=1}^{p} \sum_{i=1}^{M} i^{r-1},$$

где M — число входных признаков; p — степень полинома (2).

a_{i0}	a_{i1}	a_{i2}	a_{i3}	. . .	a_{il}	Δ^2
w_{i0}	w_{i1}	w_{i2}	w_{i3}	. . .	w_{il}	

Рис. 3. Схематическое представление особи ξ_i, $i = \overline{1, m_2}$, содержащей информацию о структуре некоторого регрессионного уравнения для результативного показателя y

Например, для некоторого результативного показателя y формируются регрессионные уравнения из двух входных показателей x_1 и x_2, степени которых не превышают 1. Эти уравнения будут иметь следующий вид:

$$y_1 = a_{10} + a_{11}x_1 + a_{12}x_2,$$
$$y_2 = a_{20} + a_{21}x_1 + a_{22}x_2 + a_{23}x_1x_2,$$
$$y_3 = a_{30} + a_{31}x_2 + a_{32}x_1x_2,$$
$$y_4 = a_{40} + a_{41}x_1 + a_{42}x_1x_2.$$

В этом случае $\mathbf{G} = \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix}$; $\mathbf{A} = \begin{pmatrix} a_{10} & a_{11} & a_{12} & 0 \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & 0 \\ a_{40} & a_{41} & a_{42} & 0 \end{pmatrix}$;

$\mathbf{W} = \begin{pmatrix} 0 & 1 & 2 & 0 \\ 0 & 1 & 2 & 3 \\ 0 & 2 & 3 & 0 \\ 0 & 1 & 3 & 0 \end{pmatrix}$, а уравнение регрессии для $y = y_1$ можно записать в общем виде:

$$y = \sum_{i=0}^{3} a_{1i} \prod_{q=0}^{1} x_{q+1}^{g_{w_{j-1iq}}}.$$

Рассмотрим алгоритм формирования структуры регрессионного уравнения.

Входные данные:

$\mathbf{Y}_i$ — матрица размерности $n \times 1$ (временные наблюдения одного из $j = \overline{1, K}$ результативных показателей функционирования системы); **X** — матрица размерности $n \times M$ (временные наблюдения входных признаков x_i, $i = \overline{1, M}$); $\mathbf{R}(i, B)$ — матрица отобранных признаков с числом параллельных эволюционных путей B и числом входных признаков $i = \overline{1, M}$; m_2 — размер популяции; v_{t2} — вероятность мутации особи поколения t.

Шаг 1. Деление выборки временных наблюдений признаков на две части:

обучающую $n_{\text{обуч}}$ и проверочную $n_{\text{пров}}$, т. е. $n = n_{\text{обуч}} + n_{\text{пров}}$. Генерация моделей (9) осуществляется на основе обучающей выборки $n_{\text{обуч}}$.

Шаг 2. Формирование начальной популяции размером m_2.

Шаг 3. До тех пор пока значение критерия регулярности Δ^2 уменьшается, генерация и изменение структуры моделей (9) продолжается (выполняются шаги 3—5).

Шаг 4. На проверочной выборке $n_{\text{пров}}$ для каждой модели (9) вычисляется критерий регулярности Δ^2 и отбираются лучшие модели.

Шаг 5. Выполнение генетических операторов [4].

Выходные данные:
матрицы **G**, **A**, **W** и номер отобранной модели с наименьшим критерием регулярности Δ^2.

После того как будет получена система наилучших моделей (9) для результативных показателей y_j, $j = \overline{1, K}$, можно выполнять последующий поиск наилучшего решения и оптимизацию.

2. Пример практического использования предлагаемого метода

Рассмотрим применение предлагаемого метода для формирования модели функционирования производственного предприятия. Изначально имеется определенная поквартальная совокупность входных признаков x_i, $i = \overline{1, M}$ (например, заемные средства, управленческие расходы и т. д.) и результативных показателей y_j, $j = \overline{1, K}$ (например, прибыль от продаж) за пять последних лет функционирования объекта исследования. На примере результативного показателя y_3 был проведен поиск множества значимых входных признаков и проведено формирование регрессионного уравнения вида (9).

На рис. 4 приведены результаты изменения отклонения весов $\overline{r}_{\min}$, $\overline{r}_{\max}$ и $\Delta\overline{r}$ (4), (5) признаков с учетом повторных запусков алгоритма отбора значимых признаков для каждого количества параллельных эволюционных путей $b = \overline{1, B}$. В результате отбора значимых признаков определено наилучшее число параллельных эволюционных путей $B = 25$, при котором веса отбираемых признаков стали стабильными. Отклонения весов (8) признаков при повторных запусках параллельного генетического алгоритма отбора значимых признаков (см. п. 1.1) наименьшие, а дальнейшее увеличение числа параллельных эволюционных путей B не дает существенных изменений $\Delta\overline{r}$.

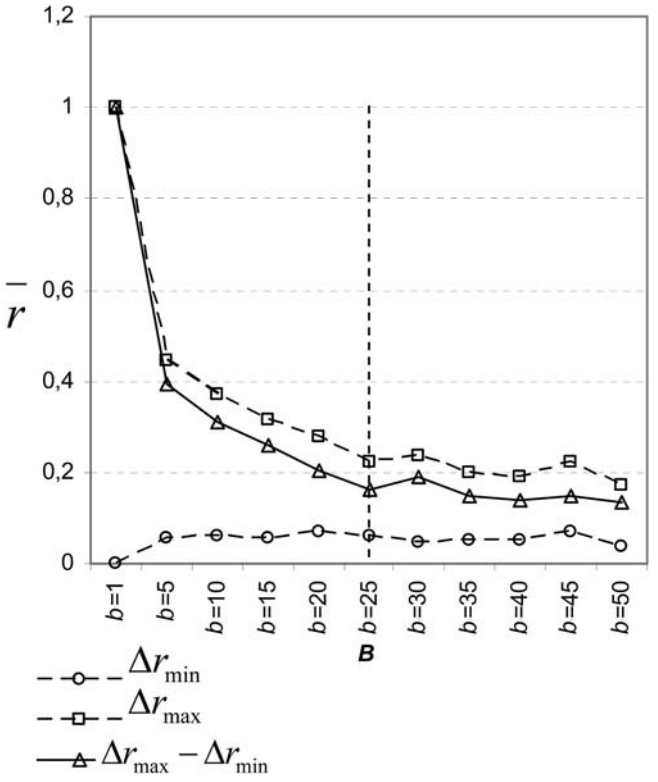


Рис. 4. Изменение отклонений весов признаков x_i, $i = \overline{1, 23}$ для y_3 при повторных запусках алгоритма отбора признаков x_i, $i = \overline{1, 23}$, с каждым количеством параллельных эволюционных путей B. На каждом $b = \overline{1, B}$ выбирается наименьшее и наибольшее отклонения весов признаков x_i, $i = \overline{1, 23}$

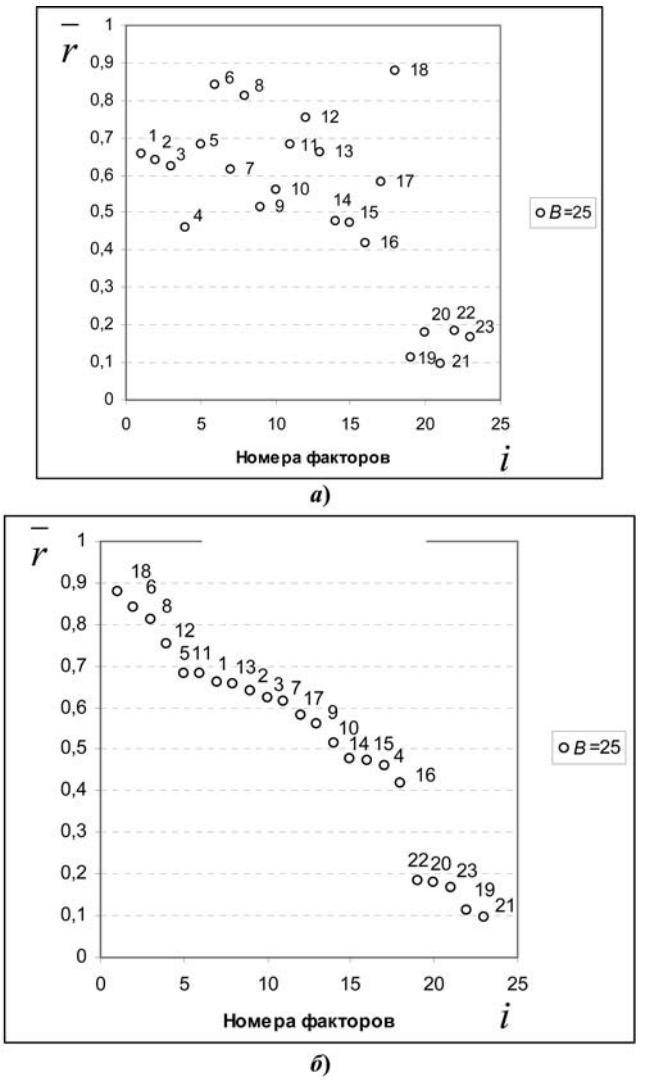


Рис. 5. Степени значимости $\overline{r}_i$ признаков x_i, $i = \overline{1, M}$, для описания некоторого результативного показателя y_3:
а — по порядку номеров признаков; *б* — упорядоченные по убыванию $\overline{r}_i$. Число эволюционных путей $B = 25$

После определения наилучшего числа параллельных эволюционных путей осуществляется отбор значимых признаков (см. п. 1.1), результаты которого приведены на рис. 5.

Из результатов алгоритма отбора значимых признаков (см. п. 1.1.) и рис. 5 следует, что наиболее оптимальными признаками являются представленные признаки x_i, $i = \overline{1, 18}$. Они имеют высокий

Оценка качества уравнения регрессии для y_3

	$S_{ст}$	$S_{ст}/\overline{y}$	F	R^2	Q	Члены регрессионного уравнения
Метод прямого отбора	24300,46 143471,91	0,04 0,49	133,79 10,23	0,990 0,857	10	X_{17}, x_3, x_{11}, x_2, x_8, x_6, x_{12}, x_{15}, x_7, x_5
Метод обратного исключения	28971,79 138921,80	0,12 0,47	133,49 15,56	0,983 0,845	7	x_3, x_5, x_6, x_7, x_8, x_{11}, x_{15}
Корреляционный метод	29021,63 164191,84	0,12 0,56	62,61 5,09	0,992 0,864	15	x_2, x_3, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}, x_{13}, x_{15}, x_{16}, x_{17}, x_{18}
Генетический алгоритм	47452,56 265847,88	0,19 0,91	32,21 1,48	0,967 0,504	14	X_{21}, x_2x_{20}, x_3x_{14}, x_4x_{14}, x_5x_{11}, x_5x_{14}, x_6x_9, $x_{13}x_{15}$, $x_{13}x_{23}$, x_{15}, $x_{15}x_{20}$
Предлагаемый метод	26230,18 51560,58	0,11 0,21	115,95 43,24	0,989 0,962	12	x_6, x_1x_{13}, x_3x_5, x_4x_9, x_4x_{14}, x_6x_{18}, x_7x_{15}, x_8x_{14}, x_8x_{18}, $x_{13}x_{14}$

уровень значимости $\overline{r}$ (частоты появления признаков), т. е. эти признаки используются для представления результативного показателя y_3 в виде (9),

$$y_3 = -44296{,}34 + 5{,}96x_6 + 4{,}69 \cdot 10^{-3}x_1x_{13} + \\ + 8{,}73 \cdot 10^{-6}x_3x_5 - 5{,}05 \cdot 10^{-5}x_4x_9 + \\ + 5{,}32 \cdot 10^{-5}x_4x_{14} - 3{,}83 \cdot 10^{-5}x_6x_{18} - \\ - 7{,}93 \cdot 10^{-4}x_7x_{15} + 3{,}84 \cdot 10^{-4}x_8x_{14} + \\ + 1{,}62 \cdot 10^{-4}x_8x_{18} - 1{,}65 \cdot 10^{-3}x_{13}x_{14}, \qquad (10)$$

с использованием алгоритма формирования структуры регрессионного уравнения (см. п. 1.2).

Оценка качества регрессионной зависимости для y_3, сформированной методами прямого отбора, обратного исключения признаков, а также с использованием генетического алгоритма, матрицы корреляции и предлагаемого в работе метода приведена в таблице. Отбор признаков для формирования структуры регрессионного уравнения вида (9) с использованием рассмотренного в работе метода проведен выше (см. рис. 5). Учитывается тот факт, что вся временная выборка признаков поделена на обучающую, проверочную и тестовую, т. е. $n = n_{обуч} + n_{пров} + n_{тест}$.

Для каждого рассматриваемого метода в таблице приведены результаты на этапе формирования и тестирования (прогнозирования) регрессионной зависимости, используя $n_{обуч} + n_{пров}$ и $n_{обуч} + n_{пров} + n_{тест}$ временных интервалов соответственно. Отношение стандартной ошибки к среднему ($S_{ст}/\overline{y}$) на этапе формирования уравнения регрессии для y_3 с учетом рассматриваемого в работе метода составляет 0,11, а на этапе прогнозирования это значение увеличивается до 0,21. Это увеличение невелико относительно результатов, полученных с помощью методов прямого отбора, обратного исключения признаков, а также с использованием генетического алгоритма и матрицы корреляции. Учитывая также высокие значения коэффициентов множественной детерминации (R^2) и критерия Фишера (F) на этапе прогнозирования с использованием предложенного метода, можно сделать вывод о высоком качестве регрессионной зависимости, сформированной этим методом (см. таблицу). Регрессионное уравнение, полученное с использованием предложенного в работе метода, имеет вид (10).

Заключение

Предложенный метод построения модели сложной системы основан на многофакторной нелинейной регрессионной модели с использованием методов группового учета аргументов, параллельного генетического алгоритма отбора значимых признаков и искусственного интеллекта.

Предложенный метод позволяет получить более качественные совокупности входных признаков.

Применение параллельного генетического алгоритма и методов искусственного интеллекта позволяет генерировать регрессионные уравнения, дающие возможность делать более качественный прогноз развития сложной системы.

Список литературы

1. **Матвеев Ю. Н.** Основы теории систем и системного анализа. Тверь: Изд. Твер. гос. техн. ун-та, 2007. 100 с.
2. **Larose D. T.** Data mining methods and models. John Wiley & Sons Inc., 2006.
3. **Mu Zhu, Hugh A. Chipman.** Darwinian evolution in parallel universes: a parallel genetic algorithm for variable selection // Technometrix. 2006. Vol. 48, N 4. P. 491—502.
4. **Cantu-Paz E.** Efficient and Accurate Parallel Genetic Algorithms. Massachusetts: Kluwer Academic Publishers. 2000. 162 p.
5. **Курейчик В. М., Курейчик В. В., Гладков Л. А.** Теория и практика эволюционного моделирования. М: ФИЗМАТЛИТ, 2003. 432 с.
6. **Елисеева И. И., Юзбашев М. М.** Общая теория статистики. М.: Финансы и статистика, 1995. 368 с.
7. **Царев Р. Ю.** Модификация метода упорядоченного предпочтения через сходство с идеальным решением для задач многоцелевого принятия решения // Информационные технологии. 2007. № 7. С. 19—23.
8. **Madala H. R., Ivakhnenko A. G.** Inductive Learning Algorithms for Complex System Modeling. Florida: CRC Press, 1994. 368 p.
9. **Мокшин В. В.** Параллельный генетический алгоритм отбора значимых факторов, влияющих на эволюцию сложной системы // Вестник КГТУ. 2009. № 3. С. 89—93.
10. **Мокшин В. В., Якимов И. М., Юльметьев Р. М., Мокшин А. В.** Рекурсивно-регрессионная самоорганизация моделей анализа и контроля сложных систем // Нелинейный мир. 2009. № 1. С. 48—63.

УДК 004.38; 514.174

В. В. Семин, аспирант,
Московский государственный институт электроники и математики (МГИЭМ),
e-mail: Noric-12@yandex.ru

Алгоритмы построения и оптимизации симплициальных комплексов на квазистационарных решетках

Рассмотрены основные положения алгоритмов построения и оптимизации симплициальных комплексов на квазистационарных решетках. Данные алгоритмы могут быть использованы в блоке вычислений с плавающей точкой в программно-аппаратном комплексе "Топологический процессор". Предложенные алгоритмы включают в себя методы построения и оптимизации триангуляции на решетках. Метод обеспечения качества и оптимизации основан на минимизации интегральной квадратичной нормы на симплексах, в качестве критерия будет рассматриваться значение телесного угла в вершинах симплексов разбиения.

Ключевые слова: *триангуляция Делоне, квазистационарная решетка, оптимизация, симплекс*

Введение

Трехмерные симплициальные разбиения некоторой конечной области R^3 состоят из тетраэдров, любые два из которых либо не имеют пересечений, либо имеют одну общую грань, размерности меньше 3. Подобные разбиения пространства востребованы для большинства методик моделирования физических процессов: реалистичного моделирования деформации объектов в компьютерной графике; в комплексах САПР; в решении большинства основных разностных дифференциальных задач в вычислительной математике, как например, в методах конечных элементов или конечных объемов. Большинство приложений предполагают специфические требования к размеру и числу симплексов в разбиении.

Создание высококачественных тетраэдральных разбиений — это сложная задача по многим причинам. Во-первых, мера линейных параметров результирующего разбиения должна быть устойчивой, строго связывая структуру данных и алгоритмы. Также существуют базовые математические проблемы, которые могут сильно усложнить процесс построения тетраэдрального разбиения, отсутствующие в двухмерном случае. Учет различных условий и ограничений также фундаментально более сложная задача для R^3. Все эти факты делают разработку алгоритмов триангуляции для R^3 сложной и подверженной ошибкам, поэтому необходимы некие общие критерии и подходы для обеспечения качества симплициальных разбиений в R^3.

Ключевые факторы и проблемы, влияющие на процесс триангуляции

За последние 20 лет число методов разбиения сильно увеличилось, но мы не будем рассматривать все существующие работы [1, 2 и т. д.]. Кратко рассмотрим проблемы, связанные с процессом построения тетраэдрального разбиения.

Процесс построения разбиения требует успешного выполнения определенного числа шагов, зависящих от приведенного ниже набора ключевых факторов.

- **Требования к качеству элементов разбиения.** Требования к объему/размеру элементов специфичны для различных видов приложений. Вообще существует очень большое число требований к качеству, начиная от минимума и максимума для двойного или телесного угла, заканчивая сложными геометрическими конструкциями. Работа [3] очень полезна для полного понимания требований к качеству элементов триангулированной решетки и их связи с различными приложениями. Наиболее популярное требование к качеству (качеству формы) тетраэдров разбиения — это требование, основанное на соотношении между радиусом описанной сферы и длиной кратчайшего ребра тетраэдра. Однако данная мера не является корректной, так как не стремится к нулю для класса вырожденных тетраэдров, называемых слайверами. Хорошая мера для данного класса вырожденных тетраэдров — отношение радиуса ребра к радиусу вписанной сферы.
- **Требования к размеру.** Точность и эффективность численных решений зависит от локального размера тетраэдров. Следовательно, распределение линейных размеров — некоторых предопределенных идеальных длин ребер, как функция пространства всегда должно учитываться. Большинство случаев предполагает наличие некоторого константного распределения для обобщенного представления разбиения и априорные или апостериорные оценки ошибок вычислений. Для избежания появления плохих значений вершинных углов симплексов требуется, чтобы функция распределения линейных размеров была достаточно гладкой [4].
- **Требования к точности аппроксимации.** Важно отметить, что требования к точности аппроксимации связаны непосредственно с требова-

ниями к размеру элементов разбиения. В некоторых случаях требуется конформность (т. е. точное приближение) области путем добавления так называемых точек Штейнера, если это возможно [5, 6]. В других случаях разбиение области должно аппроксимировать исходную область с некоторой точностью. Тогда для повышения качества тетраэдров не требуется точного соответствия исходной поверхности. Последнее особенно важно, когда исходная поверхность имеет низкое качество.

- **Методы и алгоритмы.** Среди двух основных классов методов триангуляции — прямых и итерационных — последние достаточно универсальны и поэтому, в отличие от прямых методов могут быть применены для триангуляции областей произвольного вида. Но благодаря такой универсальности приходится расплачиваться существенно бóльшим потреблением ресурсов и более трудоемкой реализацией метода в конкретном алгоритме. В настоящее время разработано большое число программных пакетов на основе того или иного итерационного метода, реализующих построение сеток (частично или полностью) в автоматическом режиме. Разбиения, построенные итерационными методами, как правило, неструктурированы и неоднородны. Это обусловлено тем, что топология разбиения формируется в процессе построения, и поэтому естественно может варьироваться даже в пределах одной подобласти. Поскольку перед построением сетки ничего нельзя сказать о ее будущей структуре, нельзя гарантировать и ее качества. Часто построенное разбиение можно существенно улучшить с помощью одного из известных многочисленных методов оптимизации. Неплохой обзор алгоритмов триангуляции Делоне можно найти в работе [7]. Также внимания заслуживает работа [8], в которой рассмотрены итерационные методы трехмерной дискретизации пространственных областей и содержатся варианты алгоритмов. Существующие методы разбиения также можно условно классифицировать по общей стратегии, которую они используют.

Расширение фронта. Начиная от границы области, новые вершины добавляются согласно локальной эвристике для того, чтобы построить тетраэдры приемлемых форм и размеров и аппроксимирующих целевую область. Существуют различные варианты [9], которые обеспечивают хороший контроль формы и размера тетраэдров, хотя и добавляют значительные вычислительные накладные расходы.

Методы, основанные на восьмеричных деревьях. Сначала каждое восьмеричное дерево обрабатывается, пока каждый лист не окажется либо строго внутри или за пределами конечной вокселизованной версии исходной области. К сожалению, сетки, построенные с помощью алгоритмов, основанных на применении восьмеричного дерева, имеют предопределенные направления ребер, что может быть неприемлемо при последующем использовании.

Триангуляция Делоне. Для заданного множества точек в трехмерном пространстве триангуляция Делоне по определению обеспечивает минимизацию максимального радиуса минимальной описанной сферы. Триангуляция Делоне обеспечивает хороший контроль над интерполяционными ошибками исходной области. Методы на основе триангуляции Делоне предлагают некоторые теоретические гарантии качества для результирующей сетки: они обеспечивают ограничения на отношение радиуса описанной сферы к ребру тетраэдра, и для них показана асимптотическая оптимальность по отношению к числу элементов в результирующей сетке. Однако и данная группа методов подвержена недостаткам: в процессе построения сетки также могут генерироваться элементы типа слайвер. Были предприняты некоторые попытки справиться с проблемой [10]. К сожалению, теоретические гарантии данного метода достаточно бедны, и полученная сетка больше не является напрямую триангуляцией согласно методу Делоне.

- **Методы оптимизации.** Даже при существовании надежных и устойчивых методов триангуляции, таких как методы на основе триангуляции Делоне, требуется очень много усилий, чтобы адаптировать их для трехмерных областей произвольного вида. В большом числе практических методов триангуляции используются методы локальной оптимизации, которые призваны улучшить качество плохих тетраэдров путем смещения их вершин. В сочетании с операциями удаления и вставки тетраэдров данные методы позволяют получить триангулированные разбиения хорошего качества. К сожалению, в данных методах часто используют не выпуклый функционал в качестве целевых функций оптимизации, что может привести к ошибочному принятию локального минимума функции как искомого.

Базовые определения и понятия

Будем рассматривать некоторую конечную подобласть R^3 произвольной топологии, обозначая ее как Ω или некоторое число связных компонентов. Сразу оговоримся, что наибольший интерес будут представлять регулярные симплициальные разбиения [11]. А также обозначим набор ограничений для Ω как $\partial\Omega$. Поверхность, описываемая $\partial\Omega$, должна представлять непрерывное триангулируемое многообразие без самопересечений и наложений.

Техника обеспечения качества и оптимизации будет основываться на минимизации интегральной квадратичной нормы на симплексах, также в качестве критерия будем рассматривать значение телесного угла в вершинах симплексов разбиения.

Точнее, в качестве целевой функции качества будем рассматривать функционал вида $\| f - f_T \|_{L^p(\Omega)}$, где f — некая функция, f_T — линейная интерполяция на основе триангуляции T области Ω. Часто в литературе, посвященной данной тематике, функционал данного вида называют ошибкой интерполяции или энергией [12].

Введем следующие обозначения: S — некоторое конечное множество точек из R^n; Ω — выпуклая оболочка S; T_s — множество всех возможных триангуляций Ω, содержащих точки из S; $Q(T, f, p) = \| f - f_{I,T} \|_{L^p(\Omega)}$, где $f_{I,T}$ — линейная интерполяция f, основанная на триангуляции T исходной области $\Omega \subset R^n$.

Заметим, что функция $Q(T, f, p)$ также представима в следующем виде:

$$Q(T, f, p) = (\int_\Omega |f(x) - f_{I,T}(x)|^p dx)^{1/p}.$$

В качестве целевого класса триангулированных решеток будем рассматривать симплициальные комплексы на квазистационарных решетках. Чтобы пояснить понятие квазистационарных решеток, приведем несколько определений.

Рассмотрим множество всех целых точек n-мерного евклидова пространства Z^n, т. е. точек с целочисленными координатами. Определим решеточную структуру соответствующей размерности как множество всех вершин из Z^2, Z^3 и Z^4 и множества простых ребер-отрезков V_p нормы p, инцидентных вершинам, где $p = \max_{i = 1...n} |x_i|$, если $\chi = (x_1 ... x_n)$ — вектор ребра. Тогда решеточная структура образует неориентированный граф, каждая вершина которого является целой точкой. Собственно пару (Z^n, V_p), где p, n — натуральные числа, будем называть решеткой. Если потребовать, чтобы евклидова длина простого ребра из V_p не превышала единицы, т. е. $p = 1$, то для пространств размерности не выше 4 это будет означать построение решетки соответствующей размерности. Определение решеточной структуры, данное выше, следует логике определений, приводимых в работе [11].

Также зададим *правильную триангуляцию некоторой области* на решетке (Z^n, V_1), где n может принимать значения из множества {1, 2, 3, 4}, как триангуляцию граней решетки соответствующей размерности, получаемую проведением диагонали в соответствующей грани и порождающую разбиение области решетки на правильно расположенные симплексы [13].

Строго говоря, правильная триангуляция (Z^n, V_1) не является триангуляцией Делоне. Поэтому очень важным для такой решетки становится свойство квазистационарности, которое будет сформулировано ниже. Но сначала введем два вспомогательных определения.

Определение 1

Малыми локальными деформациями *симплициального комплекса будем называть такие деформации, при которых любая вершина симплициального комплекса может перемещаться внутри сферы такого радиуса R, который исключает возможность совпадения с любой другой вершиной и изменения положения вершины относительно гиперплоскости основания данного симплекса.*

Необходимым и достаточным условием для осуществления такой конформной деформации является совпадение числа вершин симплициального комплекса с числом вершин в целевом кубическом комплексе, а также совпадение числа симплексов.

Определение 2

Будем называть симплициальный комплекс ***приводимым****, если он может быть однозначно преобразован в кубический симплициальный комплекс, полученный как правильная триангуляция некоторой подобласти решетки* (Z^n, V_1)*, где* $n \in \{1, 2, 3, 4\}$*, путем малых локальных деформаций без нарушения связности и выпуклости.*

В общем случае требуется применение последовательности малых локальных деформаций, сходящейся к предельной форме, на которой достигается минимум ошибки аппроксимации области Ω.

Пусть x_i — i-й элемент последовательности малых локальных деформаций для некоторого узла решетки. Тогда обозначим последовательность малых локальных деформаций для узла j как $\{x_i^j\}_{i = 1}^M$. Элемент последовательности x_M — предельная операция локальной деформации, для которой $d(j, S(j)) \leqslant \varepsilon$, $\varepsilon \in R$, для некоторого ε, где $d(x, S(x))$ — расстояние между узлом x и локальной аппроксимирующей плоскостью $S(x)$, построенной по ближайшим трем точкам к x из S.

Обозначим через X_i совокупность всех малых локальных деформаций, соответствующих i-м элементам последовательностей для N узлов аппроксимирующей триангулированной решетки. Тогда последовательность $\{X_i\}_{i = 1}^N$ образует набор итераций деформаций решетки, сходящихся к некото-

рой предельной форме, для которой выполняется условие $\|Ts| - |\Omega\| \leqslant \varepsilon$, $\varepsilon \in R$, для некоторого заданного ε.

Наконец дадим определение квазистационарности.

Определение 3

Решетка (Z^n, V_1), *где* $n \in \{1, 2, 3, 4\}$, *называется* ***квазистационарной****, если узлы решетки могут смещаться, образуя последовательность малых локальных деформаций, т. е. решетка допускает проведение над ней операции приведения.*

Введенные определения позволяют строго сформулировать алгоритм преобразования кубического симплициального комплекса к аппроксимирующему область Ω симплициальному комплексу.

Целесообразность такого преобразования заключается в том, что вследствие начального расположения в узлах решетки вершин, приближающих некоторую область Ω, не всегда обеспечивается достаточно точная аппроксимация последней или недостаточная гладкость границы приближения $\partial\Omega$. Рис. 1 иллюстрирует вышесказанное.

Алгоритм сглаживания состоит из следующего набора шагов.

Шаг 1. Выделение очередной звездной окрестности вершины x_i – star(x_i) для текущей вершины из $\partial\Omega$ ($x_i \in \partial\Omega$).

Шаг 2. Построение локальной аппроксимирующей поверхности по инцидентным x_i вершинам из $\partial\Omega$.

Шаг 3. Применение малых локальных деформаций к подобласти (Z^n, V_1) соответствующей star(x_i) в целях минимизации функционала f при соблюдении некоторого набора условий a.

Формально процедура сглаживания описывается с помощью оператора отображения с целевой функцией f и условием a над областью Ω:

$$\breve{G}:\Omega \xrightarrow[Q(T, f, p)]{} \Omega^*|_a, \qquad (1)$$

где $Q(T, f, p)$ — функционал описанного выше вида и $a = \{g_i(\Omega^*)\}$, $i = 1...k$ — набор условий, ограничивающих применение целевой функции.

Отметим, что вообще такой оператор не обязан во всех случаях обеспечивать конформность задаваемого им отображения. Это означает, что возможны склейка и добавление вершин, а также перестройка ребер, если не требуется строгое приближение по данным вершинам.

Обычно условия $g_i(\Omega^*)$ определяются как неравенства для некоторых метрических отношений, например, широко используются отношения максимального модуля ребра данного симплекса к радиусу вписанной в симплекс сферы:

$$Q = \frac{L_{\max}}{R},\ A \leqslant Q < C, \qquad (2)$$

где A, C — const.

Однако будем рассматривать более сложное условие, а именно, значение тесного угла в вершинах симплекса и связь этого значения с (2):

$$\varphi_j = 2\operatorname{arctg}\frac{(r_1 r_2 r_3)}{r_1 r_2 r_3 + (r_1 r_2) r_3 + (r_2 r_1) r_3 + (r_1 r_3) r_2}, \qquad (3)$$

где $r_i = (x_i - x_j)$, $i, j \in \{0, 1, 2, 3\}$, φ_j — телесный угол при вершине x_j в некотором симплексе.

При этом $\varphi \in (0, \pi/2]$, а его оптимальным значением является $\pi/2$, в то время как оптимальным значением Q является 4.898979 [8].

Теперь кратко рассмотрим два основных подхода к определению функционала $Q(T, f, p)$.

Диаграммы Вороного

В работе [12] показана техника построения и оптимизации разбиения как дуального к построенным диаграммам Вороного для области Ω. При этом должен достигаться минимум для следующей интегральной суммы:

$$Q(T, \|x^2\|, 1) = \sum_{i=1}^{N} = \int_{V_i} \|x - x_i\|^2 dx, \qquad (4)$$

где $f = \|x\|^2$, $p = 1$, $x_i \in S$, $i = 1...N$, V_i — объем симплекса, содержащего вершину x_i. Однако даже в таком случае не удается избежать появления элементов типа слайвер (рис. 2).

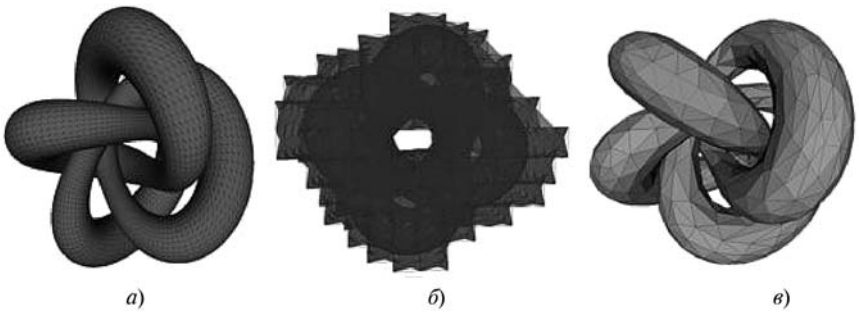


Рис. 1. Аппроксимация области (*а*) кубическим комплексом (*б*) и его сглаживание (*в*)

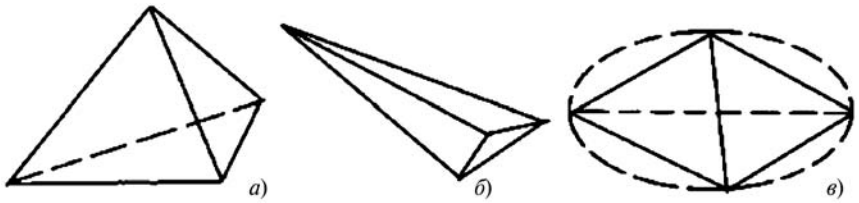


Рис. 2. Невырожденный симплекс (*а*, *б*), слайвер (*в*)

Оптимальная триангуляция Делоне

Другой метод, предложенный в работе [14], основан на построении триангуляции Делоне при достижении минимума следующей интегральной суммы:

$$Q(T, \|x^2\|, 1) = \frac{1}{n+1} \sum_{i=1}^{N} \int_{\text{star}(x_i)} \|x - x_i\|^2 dx, \quad (5)$$

где $x_i \in S$, $i = 1...N$, $\text{star}(x_i)$ — звездчатая окрестность, содержащая вершину x_i. В данном случае важно то, что $Q(T, f, p)$ минимизируется для первоначального симплициального разбиения, в то время как в первом случае, описанном выше, функционал минимизируется для дуального разбиения. Такой подход позволяет устранить проблему возникновения элементов разбиения типа слайвер. Однако при использовании данного метода для сглаживания не существует никаких формальных гарантий, доказывающих, что возможно достижение оптимальной формы решетки.

Основные положения

В данной работе рассматривается $Q(T, f, p)$ в виде (5) с некоторыми модификациями.

Приведем выражение (5) к виду

$$Q(T, \|x^2\|, 1) = \frac{1}{n+1} \sum_{i=1}^{N} x_i^2 |\text{star}(x_i)| - \int_{\Omega} x^2 dx. \quad (6)$$

Представление (6) может быть получено из (5) с учетом того, что f — выпуклая и $f_{I,T} \geqslant f$, $(f = \|x\|^2)$.

Для нахождения минимума (6) для некоторой вершины $x_i \in T_{DT}$, где T_{DT} — триангуляция Делоне области Ω, необходимо решить следующее уравнение:

$$\nabla_{x_i} Q(T, \|x^2\|, 1) = 0. \quad (7)$$

Для решения представим (6) в виде

$$Q(T, \|x^2\|, 1) = \frac{x_i^2 |\text{star}(x_i)|}{n+1} +$$

$$+ \frac{1}{n+1} \Sigma_{t_j \in |\text{star}(x_i)|} (|t_j| \Sigma_{x_k \in t_j,\, x_k \neq x_i} \|x_k\|^2) - \int_{\Omega} x^2 dx.$$

Тогда

$$\nabla_{x_i} Q(T, \|x^2\|, 1) = \frac{2x_i |\text{star}(x_i)|}{n+1} +$$

$$+ \frac{1}{n+1} \Sigma_{t_j \in |\text{star}(x_i)|} (\nabla_{x_i} |t_j| \Sigma_{x_k \in t_j,\, x_k \neq x_i} \|x_k\|^2),$$

и искомое решение (7) представимо в виде:

$$x_i^* = -\frac{1}{2x_i |\text{star}(x_i)|} \times$$

$$\times \Sigma_{t_j \in |\text{star}(x_i)|} (\nabla_{x_i} |t_j| \Sigma_{x_k \in t_j,\, x_k \neq x_i} \|x_k\|^2). \quad (8)$$

Формула (8) описывает оптимальное положение вершины x_i в звездчатой окрестности $|\text{star}(x_i)|$. В обобщенном виде решение (8) можно записать следующим образом:

$$x_i^* = \frac{1}{\Sigma_{t_k \in |\text{star}(x_i)|} \frac{|t_k|}{\rho(m_k)}} \Sigma_{t_j \in |\text{star}(x_i)|} \frac{|t_j|}{\rho(m_j)} c_j. \quad (9)$$

В качестве $\frac{1}{\rho(m_j)}$ обычно [15] выбирают некоторую функцию $\frac{1}{\mu^3}$, по смыслу являющуюся функцией плотности симплексов t_j с центром масс m_j и центром описанной сферы c_j, где μ — некая идеальная длина ребра для симплекса. Очевидно, что плотность покрытия симплексами области Ω обратно пропорциональна объему симплексов и примерно равна $\frac{1}{\mu^3}$.

Однако напрямую реализации данного метода хоть и позволяют получить решетку высокого качества, но достаточно требовательны к вычислительным ресурсам.

Воспользовавшись тем фактом, что на начальном этапе решетка однородна и структурирована, положим

$$\rho(m_j)^{1/3} = 2 \left| \sqrt[3]{\frac{1}{|\text{conf}(\Omega)|} \Sigma_{t_k \in |\text{conf}(\Omega)|} |t_k|} \right|, \quad (10)$$

$\forall j: m_j$ — центр масс тетраэдра $|t_k|$, где $\text{conf}(\Omega)$ — множество уникальных тетраэдров в регулярной триангуляции $\Omega \backslash \partial\Omega$ (см. рис. 1, *а*), т. е. фактически функция $\rho(m_j)$ зависит только от набора уникальных, отличимых друг от друга по объему тетраэдров и может быть вычислена только один раз. Такой подход помогает сократить накладные вычислительные расходы по сравнению с работой [15], а также использовать более простую функцию качества решетки $Q(T, \|x^2\|, 1)$ по сравнению с работой [16]. Корректность и устойчивость данного метода следует из работы [14].

Качество элементов разбиений, полученных в результате применения данного метода, соответствует качеству разбиений, полученных в результате применения методов работ [12, 14].

Заключение

Опираясь на методы представления кубических симплициальных комплексов [17], можно говорить о компактном и эффективном с точки зрения ресурсов ЭВМ представлении различных геометрико-топологических объектов с помощью кодирования в троичном или расширенном четверичном алфавите. Нужно отметить, что при использовании данного вида кодирования предполагается применение целочисленной арифметики для непосредственного представления геометрии объекта. В то же время в большом ряде областей, связанных с реалистичным моделированием объектов, задач гидро- и газодинамики, моделированием деформации материалов, необходимо точное отображение геометрии объекта в симплициальную модель.

Для того чтобы сделать возможным применение кубических симплициальных комплексов, построенных на узлах решетки с целочисленными координатами, для аппроксимации произвольной области из R^n, становятся необходимы алгоритмы аппроксимации и оптимизации. Примером могут служить алгоритмы, основные положения которых рассмотрены выше.

К преимуществам предложенного в данной работе подхода можно отнести быструю начальную аппроксимацию с предсказуемым, регулярным расположением симплексов с заранее известными метрическими параметрами. При использовании эффективного метода оптимизации, предложенного в данной работе, может быть получена решетка высокого качества, с равномерным распределением симплексов в результирующем комплексе.

Важно отметить, что аппаратная поддержка реализации данных алгоритмов значительно ускорила бы вычисления с их использованием. В связи с этим актуальной задачей является выделение и разработка набора соответствующих операций с плавающей арифметикой для последующей реализации в комплексе "Топологический процессор" [11]. Выделение такого набора операций позволит применять модели с плавающей арифметикой в суперкомпьютерных системах на базе комплекса, упомянутого выше.

Список литературы

1. **Frey J. L., George P. L.** Mesh Generation: Applications to Finite Elements. Paris: Hermes, 2000. 817 p.

2. **Teng S. H., Wong C. W., Lee D. T.** Unstructured Mesh Generation: Theory, Practice, and Perspectives // International Journal Computational Geometry and Applications. 2000. 10, N 3 (June). P. 227—266.

3. **Shewchuk J.** What Is a Good Linear Element? Interpolation, Conditioning, and Quality Measure. // In Proc. of 11th Int. Meshing Roundtable. 2002. P. 115—126.

4. **Ruppert J.** A New and Simple Algorithm for Quality 2-Dimensiona1 Mesh Generation. // In Proc. of the 4th ACM/SIAM Symp. on Disc. Algo. (SODA), 1993. P. 83—92.

5. **Cohen-Steiner D., De Verdiere E. C., Yvinec M.** Conforming Delaunay triangulations in 3D. // In Proc. of Symp. on Comp. Geom. 2002. P. 237—246.

6. **Cheng S. W., Poon S. H.** Graded conforming Delaunay tetrahedralization with bounded radius-edge ratio. // In Proc. of the 14th ACM-SIAM Symposium on Discrete algorithms (SODA). 2003. P. 295—304.

7. **Скворцов А. В.** Обзор алгоритмов построения триангуляции Делоне // Вычислительные методы и программирование. 2002. 3, № 1. С. 14—39. URL: http://num-meth.srcc.msu.su/

8. **Галанин М. П., Щеглов И. А.** Разработка и реализация алгоритмов трехмерной триангуляции сложных пространственных областей: итерационные методы. URL: http://www.keldysh.ru/papers/2006/prep09/prep2006_09.html

9. **Li X. Y., Teng S. H., Ungor A.** Biting: Advancing Front Meets Sphere Packing. // Int. J. on Num. Methods in Eng. 2000. 49, 1. P. 61—81.

10. **Cheng S. W., Dey T. K., Edelsbrunner H., Facello M. A. Teng S. H.** Sliver Exudation // In Proc. l5th ACM Symp. Comput. Geom. 1999. P. 1—13.

11. **Рябов Г. Г.** Алгоритмические основы топологического процессора // Методы и средства обработки информации. Труды второй Всероссийской научной конференции. М.: Издательский отдел факультета вычислительной математики и кибернетики МГУ им. М. В. Ломоносова, 2005. С. 53—58. URL: http://lvk.cs.msu.su/old/mco/mso2005-full.doc.

12. **Du Q., Wang D.** Tetrahedral mesh generation and optimization based on centroidal Voronoi tessellations. // International Journal on Numerical Methods in Engineering 56(9). 2003. P. 1355—1373.

13. **Понтрягин Л. С.** Основы комбинаторной топологии. М.: Наука, 1986. 180 с.

14. **Chen L., Xu J.** Optima1 Delaunay triangulations // Journal of Computational Mathematics. 2004. 22, N 2. P. 299—308.

15. **A11iez P., Cohen-Steiner D., Yvinec M., Desbrun M.** Variational Tetrahedral Meshing // ACM Trans. on Graphics (SIGGRAPH'05), 2005. 24(3). P. 617—625.

16. **Long Chen.** Optimal Delaunay Triangulations // Tenth SIAM Conference on Geometric Design and Computing. 2007. P. 1—121.

17. **Рябов Г. Г.** О четверичном кодировании кубических структур // Вычислительные методы и программирование. 2009. 10, № 2. С. 340—347.

УДК 519.633.6

А. В. Васюков, аспирант,
И. Б. Петров, д-р физ.-мат. наук, зав. каф.,
Московский физико-технический институт
(государственный университет)
vasyukov@gmail.com

Компьютерное моделирование последствий механических черепно-мозговых травм

Сформулирована математическая модель механической реакции головы человека на ударные воздействия, описывающая пространственное распределение механических нагрузок на мозг, и приведены некоторые результаты ее численного исследования с применением сеточно-характеристических методов на неструктурированных сетках. Эта задача является актуальной с точки зрения выяснения механизмов повреждаемости тканей мозга при черепно-мозговой травме и различных типах внешней нагрузки.

Ключевые слова: *математические модели, черепно-мозговая травма, сеточно-характеристические численные методы, неструктурированные сетки*

Введение

Численное изучение физиологических и патологических процессов, происходящих в организме человека, позволяет получить новые качественные и количественные характеристики функционирования органов в различных условиях и протекающих в них нормальных и патологических процессов, что необходимо для прогнозирования их развития, предсказания последствий патологий, выдачи медицинских рекомендаций и разработки новых принципов диагностики на ранних стадиях различных заболеваний.

На сегодняшний день медицина является экспериментальной наукой, способной констатировать факты и рекомендовать операционные или медикаментозные средства для ослабления патологических процессов. Проблема построения математических моделей функционирования различных органов остается практически открытой.

Наиболее сложной проблемой при построении моделей органов и процессов в них является экспериментальная верификация расчетных данных, так как соответствующие эксперименты практически отсутствуют. Математическое моделирование как метод исследования в данном случае имеет ряд очевидных достоинств: небольшая стоимость численного эксперимента, доступная широта диапазона изменения основных параметров, полнота получаемой в результате картины протекающих процессов во всем объеме рассматриваемой системы.

Из нейрохирургической практики известно, что области поражения мозга не всегда совпадают с областями, прилежащими к месту удара. Примером

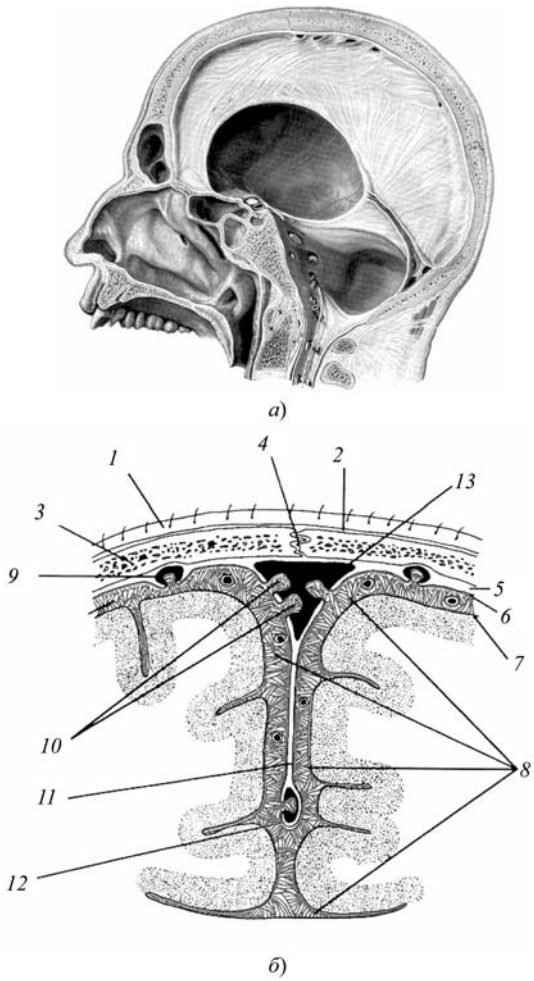


Рис. 1. Черепная коробка человека:
а — кости и твердая оболочка, сагиттальное сечение; *б* — оболочки мозга (фронтальный срез): *1* — кожа; *2* — надкостница; *3* — кость черепа; *4* — продольный шов; *5* — твердая оболочка; *6* — паутинная оболочка; *7* — сосудистая оболочка; *8* — субарахноидальное пространство; *9* — венозная впадина; *10* — арахноидальные грануляции; *11* — серп большого мозга; *12* — нижний сагиттальный синус (расстояния между оболочками преувеличены)

этому является известный феномен "противоудара": при ударе затылком область поражения мозга локализуется в лобной части головы человека.

Из рис. 1 видно, какую сложную неоднородную защитную механическую систему представляет собой черепная коробка головного мозга. Слоистая конструкция ослабляет действие продольных упругих волн (пористый слой черепной коробки), поперечных упругих волн (слой ликвора под костной частью), напряжений, вызванных нормальным (твердая костная часть черепа), а также скользящим (волосяной покров и кожа) ударами.

Объяснение явлениям, наблюдаемым при черепно-мозговой травме, может дать только изучение сложных волновых картин, образующихся в неоднородной биосреде (или биоконструкции), которую представляет собой голова.

Механико-математические модели реакции головы как сложной механической системы на ударные нагрузки также можно использовать как инструмент для исследования качественных зависимостей влияния геометрических параметров (например, размера головы, места приложения удара, возрастной атрофии мозга) на степень риска при черепно-мозговых травмах; изучение напряженно-деформационных динамических картин, возникающих в системе мозг—череп, может быть использовано для предсказания возможных физиологических патологий, которые могут появиться после черепно-мозговой травмы, поскольку карты мозга (т. е. областей, отвечающих за те или иные функции организма человека) уже хорошо известны нейрофизиологам.

1. Уравнения механики деформируемого твердого тела

Для описания поведения биологических тканей как сплошной среды под воздействием механических нагрузок ударного типа использовалась система уравнений линейной теории упругости [2]:

$$\rho \dot{v}_i = \nabla_j \sigma_{ij} \text{ (уравнения движения);}$$

$$\dot{\sigma}_{ij} = q_{ijkl} e_{kl} + F_{ij} \text{ (реологические соотношения).} \quad (1)$$

Здесь ρ — плотность среды; v_i — компоненты скорости смещения; σ_{ij}, e_{ij} — компоненты тензоров напряжений и скоростей деформаций; ∇_j — ковариантная производная по j-й координате; матрица F_{ij} описывает воздействие внешних сил. Тензор 4-го порядка q_{ijkl} определяет реологию среды, в случае линейно-упругого тела его компоненты выражаются через две независимые постоянные — константы Ламе λ и μ:

$$q_{ijkl} = \lambda \delta_{ij} \delta_{kl} + \mu(\delta_{ik}\delta_{jl} + \delta_{il}\delta_{jk}), \quad (2)$$

где δ_{ij} — символ Кронекера.

В расчетах также использовалась модель вязкоупругого тела Максвелла.

Плотность определяется из уравнения состояния $\rho = \rho_0 e^{(p/K)}$, где $p = -\frac{1}{3}\Sigma\sigma_{kk}$ — давление, $K = \lambda + \frac{2}{3}\mu$ — коэффициент всестороннего сжатия.

Уравнения (1), (2) допускают запись в матричной форме:

$$\frac{\partial}{\partial t}\mathbf{u} + \mathbf{A}_1 \frac{\partial}{\partial x_1}\mathbf{u} + \mathbf{A}_2 \frac{\partial}{\partial x_2}\mathbf{u} = \mathbf{f}, \quad (3)$$

где $\mathbf{u} = (v_1, v_2, \sigma_{11}, \sigma_{12}, \sigma_{22}, \sigma_{23})^T$ — вектор искомых функций; $\mathbf{f}$ — вектор правых частей той же размерности; $\mathbf{A}_i$ — матрицы 6×6, явный вид которых приведен в работе [3]; x_1, x_2 — независимые пространственные переменные; t — время.

Если матрицы $\mathbf{A}_i$ в уравнении (3) имеют шесть вещественных собственных чисел, то такая система называется гиперболической, и ее решения соответствуют процессам, которые обычно называют волновыми — распространение возмущений вдоль характеристических конусов в пространстве (x_1, x_2, t).

2. Механико-математическая модель системы череп—мозг

В данной работе представлено несколько механико-математических моделей головы человека. Простейшей из них является двухкомпонентная модель (рис. 2, *а*), в которой ткани кости и мозга описываются однородными изотропными материалами, имеющими усредненные механические свойства; более сложные модели учитывают наличие желудочка (рис. 2, *б*) и мембраны твердой оболочки (рис. 2, *в*).

Реологические свойства биоматериалов подвергались вариации. Так, реология мозгового вещества полагалась как линейно-упругой, так и вязкоупругой. Поведение костного материала моделировалось как изотропной линейно-упругой сплошной средой со средними свойствами пластинчатой и

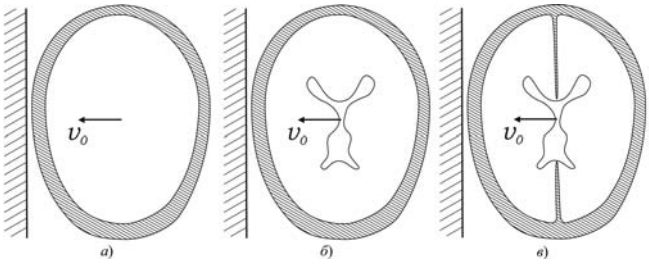


Рис. 2. Модели головы человека:
а — двухкомпонентная; *б* — с желудочками; *в* — с желудочками и мембраной (большим серпом). Промоделировано соударение головы и абсолютно твердой стенки, v_0 — относительная скорость удара (1...3 м/с)

губчатой кости, так и многокомпонентными моделями, явно выделяющими костную ткань с разными механическими характеристиками.

Моделирование взаимодействия между черепом и мозгом является сложной задачей ввиду того, что в действительности мозг имеет большое число различных по механическим свойствам оболочек, складчатых структур, врастающих друг в друга, с полостями, заполненными жидкостью (ликвором). В данной работе применялся метод явного выделения контактного разрыва с контактными условиями, которые варьировались от полного слипания до скольжения с возможностью отслоения.

3. Сеточно-характеристические методы для систем одномерных гиперболических уравнений

Для численного исследования моделей был использован ряд известных конечно-разностных методов, относящихся к классу сеточно-характеристических. Для системы уравнений с одной пространственной переменной

$$\frac{\partial}{\partial t}\mathbf{u} + \mathbf{A}\frac{\partial}{\partial x}\mathbf{u} = \mathbf{f} \qquad (4)$$

решение ищется в виде сеточной функции $\mathbf{u}_m^n$, заданной в узлах расчетной сетки $\{x_m = mh,\ t^n = n\tau\}$, где h и τ — шаги по пространству и по времени.

Монотонная схема. Данная схема строится на основе анализа поведения характеристик системы уравнений (4) и приводит к следующим формулам [3]:

$$\mathbf{u}_m^{n+1} = \mathbf{u}_m^n - \sigma\boldsymbol{\Omega}^{-1}\boldsymbol{\Lambda}^+\boldsymbol{\Omega}(\mathbf{u}_{m+1}^n - \mathbf{u}_m^n) - \sigma\boldsymbol{\Omega}^{-1}\boldsymbol{\Lambda}^-\boldsymbol{\Omega}(\mathbf{u}_m^n - \mathbf{u}_{m-1}^n), \qquad (5)$$

где $\boldsymbol{\Omega}$ — матрица, строки которой являются левыми собственными векторами матрицы $\mathbf{A}$; $\boldsymbol{\Lambda}^{\pm}$ — диагональные матрицы, содержащие соответствующие собственные числа. Схема (5) имеет порядок аппроксимации $O(h, \tau)$, обладает свойством монотонности и минимальной аппроксимационной вязкостью среди монотонных схем первого порядка, что является важным свойством при расчете динамических процессов в неоднородных средах.

Схема второго порядка точности. Единственной центральной схемой второго порядка аппроксимации на трехточечном шаблоне (т. е. использующая значения в точках $\{m-1, m, m+1\}$) является схема Лакса—Вендроффа:

$$\mathbf{u}_m^{n+1} = \mathbf{u}_m^n - \sigma\mathbf{A}\frac{1}{2}(\mathbf{u}_{m+1}^n - \mathbf{u}_{m-1}^n) + \sigma^2\mathbf{A}^2\frac{1}{2}(\mathbf{u}_{m+1}^n - 2\mathbf{u}_m^n + \mathbf{u}_{m-1}^n). \qquad (6)$$

Схема (6) имеет минимальное размазывание волнового фронта, но не является монотонной. Это проявляется в виде нефизичных осцилляций вблизи разрывов точного решения.

Гибридные разностные схемы. Если составить линейную комбинацию двух предыдущих схем, то гибридную схему можно записать в виде

$$\mathbf{u}_m^{n+1} = \mathbf{u}_m^n - \sigma\mathbf{A}\frac{1}{2}(\mathbf{u}_{m+1}^n - \mathbf{u}_{m-1}^n) + ((1-\alpha)\sigma\boldsymbol{\Omega}^{-1}|\boldsymbol{\Lambda}|\boldsymbol{\Omega} + \alpha\sigma^2\mathbf{A}^2) \times \frac{1}{2}(\mathbf{u}_{m+1}^n - 2\mathbf{u}_m^n + \mathbf{u}_{m-1}^n), \qquad (7)$$

где $|\boldsymbol{\Lambda}|$ обозначает диагональную матрицу, составленную из модулей собственных значений матрицы $\mathbf{A}$. При $\alpha = 0$ получаем схему первого порядка (5), при $\alpha = 1$ — схему второго порядка (6). Схема (7) называется гибридной, если коэффициент α выбирается в соответствии с локальными свойствами решения [4], и гибридизированной, если α имеет фиксированное значение [5], подбираемое экспериментально. В данной работе локальная гладкость решения определялась из условия, предложенного Р. П. Федоренко [9]:

$$\frac{1}{2}(\mathbf{u}_{m+1}^n - 2\mathbf{u}_m^n + \mathbf{u}_{m-1}^n) \leqslant K\frac{1}{2}(\mathbf{u}_{m+1}^n - \mathbf{u}_{m-1}^n). \qquad (8)$$

В расчетах полагалось $K = 0{,}5$. В случае выполнения условия (8) применялась схема второго порядка (6), если решение имело разрывный характер, т. е. условие (8) не выполнялось, использовалась схема первого порядка (5).

4. Разностные схемы для двумерных гиперболических уравнений

В случае двумерных гиперболических уравнений разностные схемы на регулярной расчетной сетке $\{x_l = lh,\ y_m = mh,\ t^n = n\tau\}$ могут быть получены из перечисленных одномерных разностных схем следующим образом. Каждую явную схему на трехточечном шаблоне можно представить в виде действия оператора

$$\mathbf{u}_m^{n+1} = \mathbf{u}_m^n + G(\mathbf{A}, \tau, h)\{\mathbf{u}_{m-1}^n, \mathbf{u}_m^n, \mathbf{u}_{m+1}^n\}. \qquad (9)$$

Тогда соответствующая двумерная схема на пятиточечном шаблоне будет иметь вид

$$\mathbf{u}_{l,m}^{n+1} = \mathbf{u}_{l,m}^n + \frac{1}{2}G(2\mathbf{A}_x, \tau, h) \times \{\mathbf{u}_{l-1,m}^n, \mathbf{u}_{l,m}^n, \mathbf{u}_{l+1,m}^n\} + \frac{1}{2}G(2\mathbf{A}_y, \tau, h)\{\mathbf{u}_{l,m-1}^n, \mathbf{u}_{l,m}^n, \mathbf{u}_{l,m+1}^n\}. \qquad (10)$$

Идея другого метода — расщепления по пространственным направлениям заключается в замене двумерной системы (3) тремя одномерными системами:

$$\frac{\partial}{\partial t}\mathbf{u} + \mathbf{A}_1 \frac{\partial}{\partial x_1}\mathbf{u} = 0; \qquad (11)$$

$$\frac{\partial}{\partial t}\mathbf{u} + \mathbf{A}_2 \frac{\partial}{\partial x_2}\mathbf{u} = 0; \qquad (12)$$

$$\frac{\partial}{\partial t}\mathbf{u} = f. \qquad (13)$$

Решение их выполняется с помощью некоторой одномерной разностной схемы поэтапно: сначала во всей области интегрирования решается система (11), затем результат используется для решения системы (12), и, наконец, конечное решение получается в результате прибавления правой части (13).

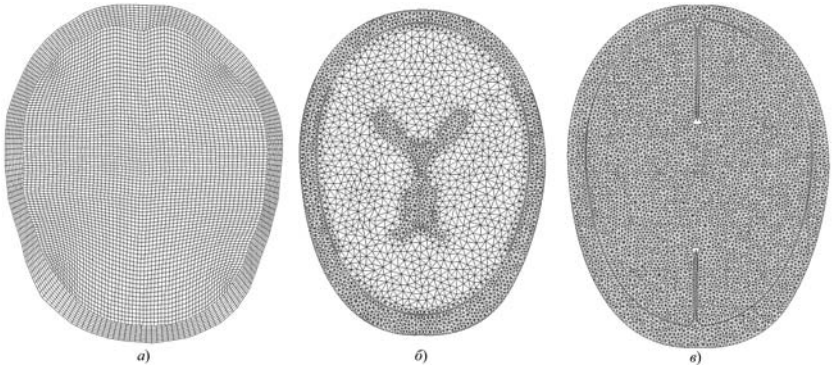


Рис. 3. Расчетные сетки для разных моделей головы человека:
а — двухкомпонентной; *б* — с желудочками; *в* — с желудочками и мембраной

5. Расчет контактных границ

Расчеты, в которых область интегрирования распадается на участки различных реологий и механических свойств, возможно перемещающиеся друг относительно друга, требуют аккуратного решения задачи контактного разрыва на поверхностях соприкосновения тел. Сеточно-характеристические схемы позволяют формулировать условия на контактной границе двух областей интегрирования в явном виде. Как показывают результаты работ [7, 8], явное выделение контактных границ имеет ряд преимуществ над их сквозным расчетом.

Полный алгоритм расчета контактных границ для двумерного случая приведен в работах [7, 8]. Данный алгоритм позволяет явно выделять контактные границы между телами разной реологии и рассчитывать их, используя различные контактные условия.

В данной работе использовались:

условие полного слипания

$$\begin{aligned}&\mathbf{v}[1] = \mathbf{v}[2]\\ &\text{(равенство скоростей);}\\ &\sigma_n[1] = \sigma_n[2]\\ &\text{(равенство нормальных напряжений);}\\ &\sigma_\tau[1] = \sigma_\tau[2]\\ &\text{(равенство касательных напряжений);}\end{aligned} \qquad (14)$$

условие свободного скольжения

$$\begin{aligned}&\mathbf{v}_n[1] = \mathbf{v}_n[2]\\ &\text{(равенство нормальных скоростей),}\\ &\sigma_n[1] = \sigma_n[2]\\ &\text{(равенство нормальных напряжений),}\\ &\sigma_\tau[1] = \sigma_\tau[2] = 0\\ &\text{(отсутствие касательных напряжений).}\end{aligned} \qquad (15)$$

6. Обобщение на неструктурированные треугольные сетки

В данной работе был реализован алгоритм, позволяющий применить описанные двумерные разностные схемы для расчетов на неструктурированных треугольных сетках. При этом значения в узлах пятиточечного шаблона определялись линейной интерполяцией внутри прилежащих треугольников.

Отметим, что уже имеются программные инструменты промышленного уровня для построения треугольных сеток в областях произвольной формы и связности. Например, в данной работе использовалась программа Triangle (автор J. R. Shewchuk). Треугольная сетка позволяет управлять размерами ячеек, чтобы добиться либо полной однородности, либо, при необходимости, сгущения или разряжения в определенных зонах.

Таким образом, можно сделать вывод, что использование гибридных схем с расщеплением на неструктурированных треугольных сетках вполне оправдано, так как позволяет получать численные решения удовлетворительного качества в областях интегрирования произвольной формы (рис. 3).

7. Результаты расчетов и сравнение с клиническими данными

На рис. 4 приведены интегральные характеристики механического воздействия на мозг при боковом ударе, полученные с помощью двухкомпонентной модели с условием свободного скольжения на границе череп—мозг. Внешняя нагрузка задается как соударение системы череп—мозг с абсолютно жесткой неподвижной преградой с заданной начальной скоростью (1...3 м/с).

Наиболее опасными представляются концентрации максимальных растягивающих (положительных) (рис. 4, *б*) и сдвиговых напряжений (рис. 4, *в*). В частности, упомянутое выше явление противоудара продемонстрировано на рис. 4, *б*: наиболее

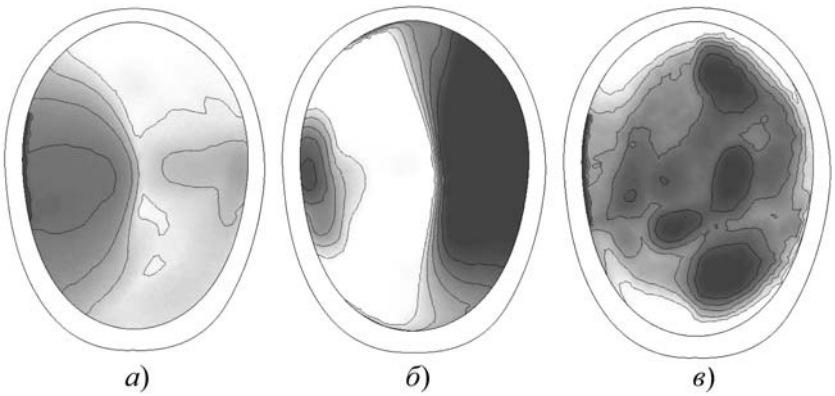


Рис. 4. Результаты расчета двухкомпонентной модели:
а — максимальное сжатие; *б* — максимальное растяжение; *в* — максимальные сдвиговые напряжения

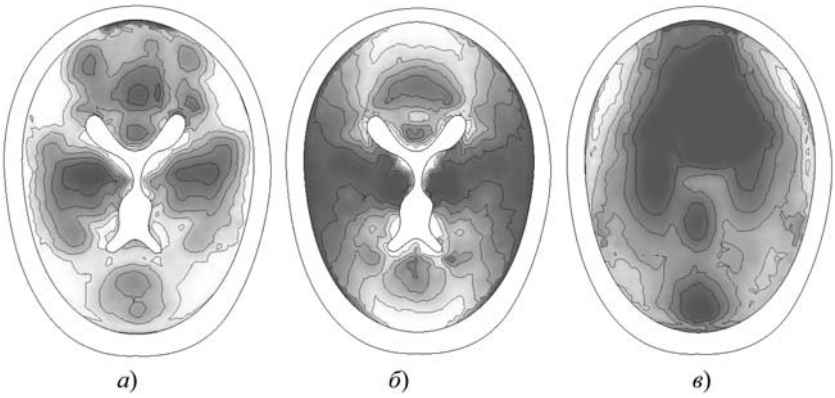


Рис. 5. Распределение сдвиговых напряжений:
а — полное слипание; *б* — скольжение; *в* — двухкомпонентная модель

опасные повреждения мозгового вещества локализуются в области положительных напряжений.

На рис. 5 приведены распределения сдвиговых напряжений при ударе снизу, полученные с помощью разных моделей. Использование условия полного слипания на границе череп—мозг приводит к концентрации сдвиговых напряжений вдоль контактной границы на боковых поверхностях, в то время как скользящий контакт полностью их снимает.

Учет наличия желудочков оказывает слабое влияние на распределение областей максимального сжатия и растяжения, но существенно влияет на распределение сдвиговых нагрузок. Наличие мембраны является более существенным для локализации областей сжатия—растяжения при боковых ударах.

На рис. 6, *а* приведен пример КТ-снимка пациента с ушибом головного мозга тяжелой степени, полученным от удара слева при ДТП (данные предоставлены Главным военным госпиталем им. Н. Н. Бурденко). Стрелками на томограмме указаны места поражения мозгового вещества. На рис. 6, *б* приведено соответствующее распределение максимальных сдвиговых нагрузок.

По результатам сравнения расчетных данных с клиническими данными по 18 пациентам, получившим черепно-мозговые травмы различной тяжести, а также с имеющимися в медицинской литературе качественными описаниями биомеханики черепно-мозговой травмы, сделан вывод об удовлетворительной описательной способности модели.

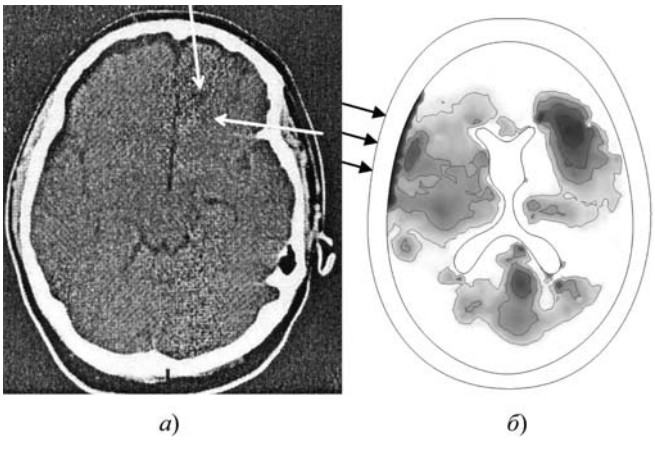


Рис. 6. КТ-снимок пациента с ушибом головного мозга тяжелой степени, удар слева (*а*) и соответствующее распределение максимальных сдвиговых нагрузок (*б*)

Список литературы

1. **Белоцерковский О. М.** Численное моделирование в механике сплошных сред. М.: Физико-математическая литература, 1994. 442 с.

2. **Новацкий В. К.** Теория упругости. М.: Мир, 1975. 872 с.

3. **Магомедов К. М., Холодов А. С.** Сеточно-характеристические численные методы. М.: Наука. 1988. 288 с.

4. **Петров И. Б., Холодов А. С.** О регуляризации разрывных численных решений уравнений гиперболического типа // Журнал вычисл. матем. и матем. физ. 1984. Т. 24, № 8. С. 1172—1188.

5. **Петров И. Б., Тормасов. А. Г., Холодов А. С.** Об использовании гибридизированных сеточно-характеристических схем для численного решения трехмерных задач динамики деформируемого твердого тела // Журнал вычисл. матем. и матем. физ. 1990. Т. 30, № 8. С. 1237—1244.

6. **Агапов П. И., Петров И. Б., Челноков Ф. Б.** Численное исследование задач механики деформируемого твердого тела в неоднородных областях интегрирования // Обработка информации и моделирование: Сб. ст. М.: Моск. физ.-тех. ин-т, 2002. С. 148—157.

7. **Белоцерковский О. М., Агапов П. И., Петров И. Б.** Моделирование последствий черепно-мозговой травмы // Медицина в зеркале информатики. М.: Наука, 2008. С. 113—124.

8 **Петров И. Б., Тормасов А. Г, Холодов А. С.** О численном изучении нестационарных процессов в деформируемых средах многослойной структуры // Изв. АН СССР. Механика твердого тела. 1989. № 4. С. 89—95.

9. **Федоренко Р. П.** Введение в вычислительную физику. М.: Изд-во Моск. физ.-техн. ин-та, 1994. 528 с.

НЕЙРОСЕТЕВЫЕ ТЕХНОЛОГИИ

№ 5
МАЙ
2011

УДК 004.032.26:622.2

А. И. Галушкин, д-р техн. наук, проф.,
Центр информационных технологий
и систем органов исполнительной власти
Минобрнауки России,
e-mail: neurocomputer@yandex.ru,
П. А. Казанцев, канд. техн. наук, вед. науч. сотр.,
ООО "Павлин-Технология"

Нейросетевое распознавание гранулометрического состава шарообразных тел применительно к горно-рудному производству

Представлены описание и сравнительный анализ нейросетевых и других алгоритммов оценки гранулометрического состава сыпучих материалов.

Ключевые слова: *нейронные сети, распознавание образов, гранулометрия, горнообрабатывающая промышленность*

Контроль и управление качеством продукции всегда являлись важнейшими задачами в любом производственном процессе. Ввиду удобства последующей обработки в некоторых производственных процессах сырьевому материалу придают шарообразную форму [1]. Размеры получающихся "шариков", или "окатышей", сильно влияют на качество получаемого конечного сырья. Распределение шарообразных тел по размерам называют гранулометрическим составом (или сокращенно — грансоставом). Отслеживание и поддержание грансостава в определенных пределах является важной задачей при контроле и управлении качеством сырья. Способы решения подобных сложных задач управления предлагает теория нейроуправления.

В данной статье исследуются возможность и точность нейросетевого распознавания гранулометрического состава шарообразных тел по данным измерения поверхности набора этих тел. Эксперименты проводились с трехмерной моделью набора, диаметры тел которого лежали в диапазонах, принятых в качестве эталонных на горнообогатительных комбинатах. Описаны нейросетевой алгоритм распознавания, а также метод формирования презентативных обучающей и тестовой выборок, требуемых для эффективного обучения и верификации нейросетевой системы. В статье приведено сравнение нейросетевого подхода к решению данной задачи с линейными методами.

Нейросетевой алгоритм распознавания гранулометрического состава шарообразных тел

Модель набора шарообразных тел. Диапазон варьирования диаметров тел был взят исходя из опыта разработки гранулометра для распознавания грансостава сырых окатышей сырой руды на одном из горнообогатительных комбинатов.

Гранулометрический состав тел в массовых долях — $clasm[i]$, где $i = 0, ..., 5$, представлен в виде доли массы тел данного класса в общем объеме. Тела из классов, используемые при генерации, имеют диаметры:

- менее 5 мм;
- от 5 до 10 мм;
- от 10 до 14 мм;
- от 14 до 16 мм;
- от 16 до 18 мм;
- более 18 мм.

Физическая постановка задачи. Пусть имеется трехмерная модель набора шарообразных тел, генерируемых в соответствии с изложенной выше моделью набора шарообразных тел. По информации, взятой с поверхности данного набора, требуется определить весовое содержание шарообразных тел в каждой из шести категорий — гранулометрический состав (грансостав).

Нейросетевая постановка задачи. По признакам, выбранным с огибающей поверхности набора, с помощью нейронной сети требуется получить шестимерный выходной вектор, отвечающий весовому содержанию тел в каждой категории.

Обучающая и тестовая выборки. Формирование наборов шарообразных тел осуществлялось исходя из заданных грансоставов и линейных размеров основания прямоугольного параллелепипеда, охватывающего набор тел. Так, в данном эксперименте ширина основания последнего составляла 300 мм, длина — 500 мм.

Весовое одержание в каждой категории генерировалось в соответствии со следующей функцией распределения

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma}\,\mathbf{e}^{\frac{-(x-M)^2}{2\sigma^2}},$$

где M и σ — среднее значение и среднеквадратическое отклонение размеров шаров.

Путем варьирования параметров M и σ был получен широкий диапазон грансоставов. Процент по каждой категории в обучающей выборке изменялся от 0 до 90 %. Таким образом, на этапе обучения обеспечивалось представление нейронной сети примеров, значительно удаленных друг от друга в пространстве входных признаков. Другими

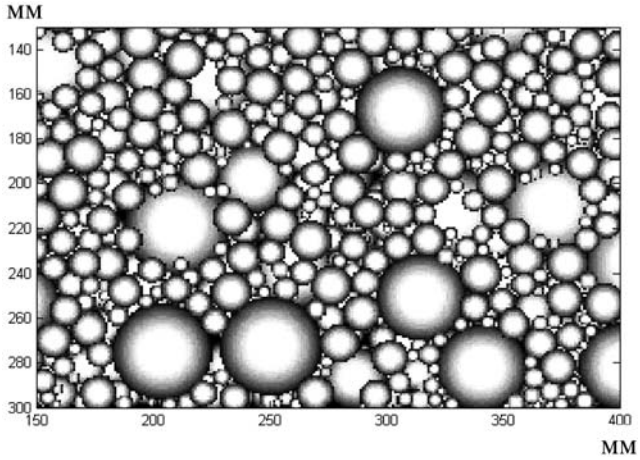


Рис. 1. Яркостное изображение фрагмента поверхности

словами, если рассматривать нейронную сеть как интерполятор функции, то значения точек функции, которую требуется интерполировать, лежат в широком диапазоне.

В каждой сотне примеров из обучающей выборки форсировалось изменение определенных категорий. Так, например, первая сотня примеров была сгенерирована по грансоставам, в которых весовой процент в категории тел + 18 мм превалировал, во второй сотне преобладал весовой процент в категории — 5 мм и т. п. Обучающая выборка состояла из 1800 примеров, тестовая — из 2000.

После генерации грансоставов по ним строились наборы шарообразных тел и их огибающие, по которым затем формировались модели фотоснимков. Пример фотоснимка фрагмента одного из наборов приведен на рис. 1.

Пространство признаков. Метод проекций. Входной вектор формировался следующим образом. Создавался нулевой вектор размерностью $M = 255$. Для каждой точки огибающей поверхности вычислялось ее значение яркости, после чего значение p-й компоненты входного вектора инкрементировалось на единицу. После обработки всей огибающей получался вектор, каждая p-я компонента которого содержала число точек с яркостью p.

Проще говоря, такой вектор несет информацию о том, сколько раз на огибающей встречается каждое значение y. Так, число раз, которое каждая горизонтальная прямая пересекает огибающую, записывается в ячейку массива с номером, равным значению яркости огибающей.

Такой метод формирования пространства признаков имеет два преимущества. Первое из них заключается в сокращении размерности входного пространства, что уменьшает размер сети. Вторым преимуществом является то, что во входной вектор помещается интегральная информация обо всей огибающей, а не об отдельно выбранных ее участках.

Структура сети, алгоритм обучения и функционал вторичной оптимизации. Эксперименты проводились с двухслойными нейронными сетями с прямыми последовательными связями. Функция активации первого слоя — сигмоидная, второго слоя — положительно-линейная. Сети обучались по правилу обратного распространения ошибки. Функционал вторичной оптимизации — среднеквадратичная ошибка выходов сети:

$$E = \frac{1}{6} \sum_{i=1}^{6} (y_{\text{ж}i} - y_{\text{д}i})^2,$$

где $y_{\text{ж}i}$ — желаемое значение i-го выхода; $y_{\text{д}i}$ — действительное значение i-го выхода. Обучение проводилось на протяжении 300 эпох.

Экспериментальные результаты

Было проведено несколько экспериментов с сетями с различным числом нейронов в 1-м слое (H_1) (см. таблицу).

На рис. 2 (см. третью сторону обложки) представлен график изменения весовых процентов в категории +18 мм (самая большая ошибка), совмещенный с графиком весовых процентов, которые выдает нейронная сеть 500 × 6 по этой тестовой выборке (темные — выход нейронной сети по тестовой выборке, светлые — значения в тестовой выборке). Из графиков и значения вектора ошибок видно, что сеть не только отслеживает тенденцию изменения грансостава, но и достаточно точно определяет его.

Гипотетический линейный алгоритм оценки. К сожалению, все коммерческие модели гранулометров являются закрытыми системами. Поэтому оценить точность алгоритмов, лежащих в их основе, можно лишь исходя из некоторых предположений, основывающихся на анализе коммерческих презентаций продуктов и технической документации. Предположим, что существует некий алгоритм, с помощью которого можно идеально точно определить линейный размер тела, если $X\%$ половины площади его поверхности видимы. Так,

Результаты экспериментов (ошибки оценки гранулометрического состава) с моделями черно-белых фотоснимков поверхностей (в градации серого 0...255)

Число нейронов первого слоя H_1	Категория					
	+18 мм	−18 + 16 мм	−16 + 14 мм	−14 + 10 мм	−10 + 5 мм	−5 мм
100	8,82	4,58	5,08	4,46	3,9	4,89
300	3,88	1,82	1,95	1,96	1,55	1,6
500	3,77	1,58	1,74	1,96	1,6	1,47
800	3,93	1,93	1,67	1,71	1,52	1,31
1000	4,25	2,02	1,73	1,72	1,59	1,31

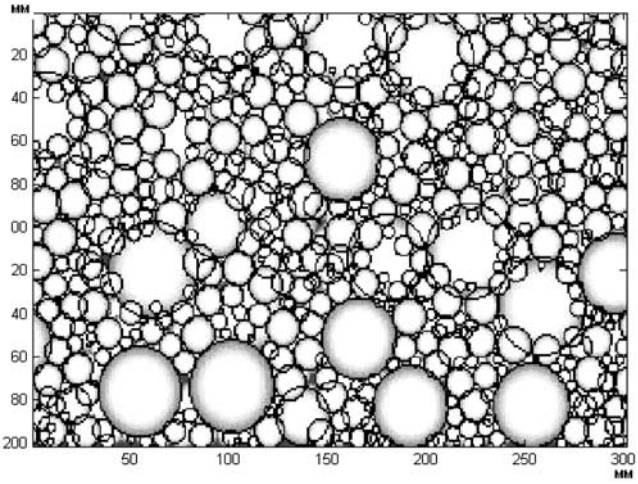


Рис. 3. Восстановление реальных размеров тел по их видимой поверхности

если видимыми являются менее X %, но более $X/2$ % половины площади, то линейный размер (в нашем случае — диаметр) устанавливается произвольным в диапазоне $d \pm e$, где d — реальный диаметр тела, а e — ошибка, не превышающая x % от d. Диаметры всех тел набора, в том числе и скрытых, считаются известными. Далее, для тел, диаметры которых были определены точно или с некоей ошибкой, высчитывается гранулометрический состав. Определяя таким образом линейные размеры тел верхнего слоя, мы сможем оценить ошибку определения грансостава, связанной с косвенностью данного алгоритма, который не учитывает физики распределения тел. На рис. 3 показано, как частично видимые тела достраиваются до полного диаметра, что отображено прорисовкой округлых контуров.

Эксперименты проведены с яркостной моделью, описанной выше, для различных значений X и x.

На рис. 4 (см. третью сторону обложки) приведен график зависимости ошибки в категории 18+ мм от величин e и X, который наглядно показывает, насколько сильно линейные классические алгоритмы определения грансостава зависят от точности определения линейных размеров тел набора.

Как можно видеть из сравнения таблицы и рис. 4, результаты оценки грансостава нейронной сетью с $H_1 = 500$ имеют примерно такую же точность, как и результаты оценки с помощью гипотетического алгоритма, требующего для точного определения диаметра тела всего лишь 10 % половины площади тела. Маловероятно, что существуют настолько точные алгоритмы, поэтому оценку возможностей алгоритмов определения грансостава по линейным размерам следует проводить исходя из строк таблицы.

Нельзя не заметить, что быстродействие нейросетевого алгоритма в несколько раз больше быстродействия алгоритма, определяющего грансостав по линейным размерам, так как последний использует в своей основе попиксельную обработку изображения, Так, обработка нейросетевым алгоритмом 2000 тестовых изображений при $H_1 = 500$, включая предобработку изображения для формирования входов нейронной сети, занимает около $t_1 = 238{,}31$ с. При этом предобработка занимает 238 с, а прямой проход нейронной сети — 0,31 с. Обработка же гипотетическим алгоритмом с выделением границы тел одного изображения будет занимать около 30 с. Следует отметить, что гипотетический алгоритм, на самом деле, включает в себя довольно простые попиксельные операции перезаписи, т. е. ни о каких вычислениях градиентов яркости речь даже не идет, и, скорее, полученное время вычисления следует рассматривать как время предобработки. Время же вычисления радиусов, как ожидается, значительно повысит указанное время обработки линейным алгоритмом. Нейросетевой же алгоритм, отработанный на данной модели, является конечным инструментом распознавания. Вычисления проводились в среде Matlab 7.0 на машине с процессором Pentium 4 с 1 Гбайт RAM.

Список литературы

1. **Галуза Ю. П., Галушкин А. И., Казанцев П. А., Коробкова С. В., Лодягин А. М., Остапенко Г. П., Пантюхин Д. В., Пантелеев С. В.** и др. Разработка системы нейроуправления чашевыми окомкователями. Отчет по НИР. М.; Научный центр нейрокомпьютеров. 2004.

2. **Казанцев П. А., Коробкова С. В., Лодягин А. М.** Нейросетевое распознавание гранулометрического состава шарообразных тел и его сравнение с линейными методами // Матер. Международной научно-технической конференции "Интеллектуальные машины и системы". Дивноморск, 2005. Т. 2. С. 251—257.

3. **Казанцев П. А., Коробкова С. В., Лодягин А. М.** Распознавание гранулометрического состава потока округлых тел по огибающей поверхности нейросетевым методом // Матер. конф. "Нейросетевые технологии и их применение" ("НСТиП-2005"), г. Краматорск, 2005.

4. **Galushkin A. I., Kazantsev P. A., Korobkova S. V., Lodyagin A. M., Panteleev S. V.** Neural Network Recognition of Spherical Bodies Set Grain-Size Distribution Using Envelope of Surface // Proceedings of IEEE WCCI'2006.

5. **Галушкин А. И.** Теория нейронных сетей. М.: ИПРЖР, 2000.

6. **Нейроматематика** / Под ред. А. И. Галушкина. Т. 6 серии "Нейрокомпьютеры и их применение". М.: Радиотехника, 2002.

УДК 004.043

А. Р. Исхаков, преподаватель,
e-mail: intellab@mail.ru,
Р. М. Асадуллин,
д-р физ.-мат. наук, проф., зав. каф.,
e-mail: asadullin54@mail.ru,
М. Р. Богданов,
канд. биолог. наук, доц. каф.,
e-mail: bogdanov_marat@mail.ru,
Башкирский государственный педагогический университет им. М. Акмуллы, г. Уфа
Н. И. Федоров, д-р биолог. наук. зав. лаб.,
e-mail: fedorov@anrb.ru,
Институт биологии УНЦ РАН, г. Уфа

Автоматизация предварительной обработки картографического материала, содержащего цветные контуры, в целях их дальнейшей векторизации

Обсуждается методика фильтрации пикселей в полноцветном представлении сканированного картографического материала с помощью нейронной сети. Использована программа MATLAB с модулями расширения Neural Network Toolbox и Image Processing Toolbox. Предлагаемый подход позволяет экономить машинное время и помогает избежать потерь данных.

Ключевые слова: *обработка изображений, фильтрация пикселей, ГИС, искусственная нейронная сеть, технология предобработки цветных изображений, кластеризация, краевая сегментация, нейронный слой Кохонена, среда разработки MATLAB, модуль Image Processing Toolbox*

Введение

Географические информационные системы (ГИС) в настоящее время переживают период бурного развития. ГИС работают с объектами, имеющими географическую привязку. Для описания объектов земной поверхности в ГИС широко применяются векторные объекты. В частности, для моделирования небольших природных объектов используются точки. Для описания незамкнутых протяженных объектов используются ломаные, полигонами обозначают замкнутые природные объекты. Объекты, относящиеся к одной и той же предметной области, помещают в один слой. В ГИС может быть много слоев, например, слой транспортной сети, слой ареалов редких растений и т. д.

Современные ГИС-программы имеют развитые средства, предназначенные для создания и редактирования векторных объектов, но создание таких объектов вручную очень трудоемко. Автоматизированная векторизация отсканированной топографической основы может избавить разработчика от таких проблем и трудностей. Подобная возможность заложена в некоторых современных ГИС. Так, популярный пакет ESRI ArcGIS содержит в своем составе модуль векторизации ArcScan, на который мы ориентировались в своей работе.

Программам векторизации растровых изображений присущи определенные ограничения, они требуют предварительной обработки изображений. В частности, ArcScan работает только с однобитными изображениями. ArcScan при векторизации распознает ломаные и полигоны. Точки добавляются вручную. Ломаные распознаются относительно хорошо. С полигонами дело обстоит сложнее. Для успешной векторизации полигоны должны иметь четкие непрерывные края. Если же из-за наличия разрывов векторизовать их не удается, то приходится восстанавливать границы вручную или использовать полигоны, опираясь на отсканированную карту в качестве топографической основы.

Очевидным недостатком описанных подходов является их трудоемкость в векторизации полигонов. Кроме того, при создании карт для ГИС часто возникает необходимость группировки полигонов одного цвета, имеющих однотипные контуры, в отдельные слои. Это не является сложной задачей, но ситуация усугубляется тем, что отсканированные карты часто бывают раскрашены вручную, в связи с чем однотипные контуры имеют неоднородную закраску.

Целью данной статьи является обсуждение подхода автоматизации предварительной обработки растрового картографического материала, содержащего цветные контуры для их последующей векторизации в полигоны в ГИС с использованием технологии нейросетевой предобработки цветных изображений.

Описание технологии предварительной обработки картографического материала

Для сравнения рассмотрим векторизатор ArcScan, который работает с монохромными изображениями, в то время как обычно исходные карты являются цветными и имеют неоднородную окраску однотипных контуров (рис. 1). При создании соответствующих ГИС возникает необходимость получить из областей определенного оттенка замкнутые монохромные контуры с четкими краями.

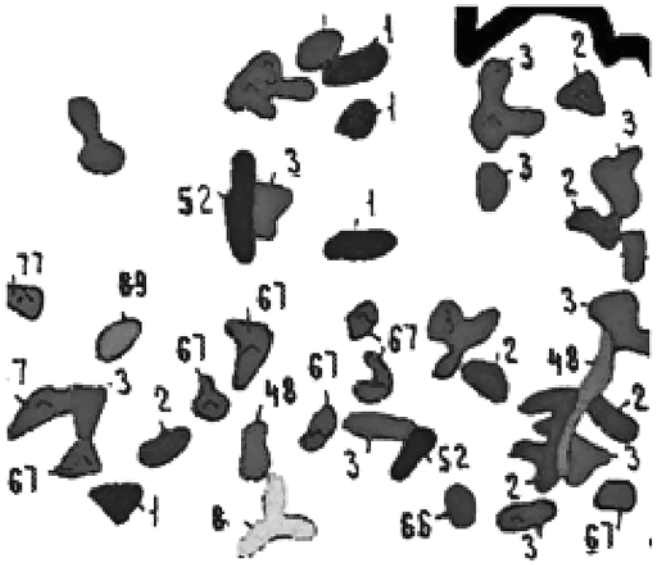


Рис. 1. Фрагмент геоботанической карты сенокосов и пастбищ

Предлагается решать задачу в два приема. На первом этапе осуществляется ***предварительная обработка исходного изображения***, в ходе которой выполняются:

1) перевод исходного полноцветного изображения в монохромное;

2) фильтрация шумов, полученных при бинаризации;

3) восстановление пикселей, потерянных при бинаризации и фильтрации;

4) выделение пикселей векторизуемых областей из исходного изображения с помощью маски.

На втором этапе происходит ***разработка нейронной сети***, используемой в дальнейшем для выделения из исходного изображения интересующих областей. Разработка нейронной сети осуществляется в приводимой ниже последовательности:

1. Выбор пикселей, полученных на четвертом шаге предварительной обработки, и создание генеральной совокупности.

2. Вычисление объема генеральной совокупности и числа пикселей в обучающем множестве.

3. Определение параметров нормального закона распределения и генерация пикселей обучающего множества.

4. Оценка качества обучающего множества на репрезентативность к генеральной совокупности. В случае, если обучающее множество достаточно хорошо представляет главное множество, можно перейти к шагу 5, в противном случае — перейти к шагу 3.

5. Создание нейронной сети и ее обучение.

6. Создание тестового множества.

7. Проверка качества нейронной сети на тестовом множестве.

8. При получении удовлетворительного результата на шаге 7 переход к шагу 9, иначе переход к шагу 5.

9. Разработка алгоритма применения нейронной сети по отношению к исходному изображению и его апробация.

Рассмотрим оба этапа работы подробнее.

Предварительная обработка исходного изображения

Как уже говорилось выше, предварительная обработка изображения начинается с перевода цветного изображения в монохромное с помощью экспериментально подобранного порога. Выбор порога определяется цветом и яркостью паттернов [2, 6]. При низком значении порога остаются лишние паттерны, высокий порог может привести к потере важных данных. Опытным путем было установлено, что оптимальным является значение порога 0,75.

Обработку изображения осуществляли с помощью программы MATLAB (команда im2bw) [2, 6]. Результат пороговой обработки исходного изображения (рис. 1) представлен на рис. 2. Анализ полученного результата выявил некоторые проблемы, вызванные пороговой обработкой изображения. В частности, исчезла область, закрашенная желтым цветом, в областях 77 и 89 (рис. 2) наблюдается потеря внутренних значащих точек, появились шумы в виде небольших областей.

Для улучшения качества изображения мы применили медианный фильтр (команда medfilt2 программы MATLAB применялась с апертурой 5 × 5) [2, 6]. Фильтрация помогла восстановить внутренние пиксели областей 77 и 89, а также сгладить края этих областей. Область желтого цвета восстановить не удалось (рис. 3).

Рис. 2. Результат пороговой обработки командой im2bw среды MATLAB

Рис. 3. Результат медианной фильтрации командой medfilt2 среды MATLAB

Рис. 4. Результат морфологической операции bwmorph среды MATLAB

Мы посчитали, что результаты медианной фильтрации требуют доработки (на изображении остались одиночные точки), поэтому в качестве дополнительного метода восстановления информации была выбрана морфологическая операция "закрытие" (bwmorph), которую мы применили 2000 раз [6]. При этом границы интересующих нас областей существенно не изменились (рис. 4).

Морфологическая операция "закрытие" завершает предобработку изображения [2, 6]. Полученный результат можно применить в дальнейшем в качестве маски для выделения из исходного изображения точек областей интереса.

Разработка нейронной сети

Второй этап направлен на создание, обучение и применение нейронной сети, предназначенной для кластеризации пикселей исходного изображения по их цвету. Множество, полученное на шаге 4 первого этапа, мы использовали в качестве маски для выделения генеральной совокупности. Для заданного изображения с размерами 500 × 400 (см. рис. 1) размер этой совокупности составил 154 944 пикселей. Множество с такой мощностью не приемлемо для обучения нейронной сети, так как для этого потребуется слишком много времени [3, 5]. Необходимо подобрать подмножество, которое бы достаточно хорошо описывало генеральную совокупность. В генеральной совокупности имеются пиксели различных цветов, образующих разные кластеры. Пиксели, выражающие оттенки этих цветов, будут достаточно далеко удалены от центров кластеров. Поэтому в выборочной совокупности все цвета должны быть представлены равномерно [3].

Для обучения нейронной сети возможности различать не только разные цвета, но и их оттенки, нужно будет разделить генеральную совокупность на обучающее, контрольное и тестовое множества [3]. Для создания обучающего множества необходимо сгенерировать последовательность из N равномерно распределенных случайных чисел из отрезка [1; M], где M — число доступных пикселей. Эти случайные числа будут представлять индексы выбираемых из обучающего множества пикселей. Таким образом, будет создано обучающее множество.

В среде MATLAB можно создать такое множество следующим образом. Определившись со значениями концов отрезка a и b для генерации случайных чисел, а также математическим ожиданием μ и дисперсией σ для нормального закона распределения, нужно набрать следующие строки, чтобы сгенерировать набор индексов в количестве N единиц [4]:

```
[p, q] = size(gnrl);
p = p/2;
N = 10 000;
z = floor(normrnd(p, 1000, 1, N));
```

В таком случае в массиве z будут находиться необходимые индексы. Выбирая по этим индексам пиксели из набора gnrl, полученного в результате прохождения первого этапа, придем к искомому обучающему множеству tr:

```
tr = gnrl(z(:),:);
```

Численное представление обучающего множества неудобно для восприятия. Множество будет гораздо нагляднее, если отобразить его пиксели (рис. 5).

Из рис. 5 видно, что в обучающем множестве преобладают пиксели, относящиеся к кластеру "зеленого" цвета. Согласно [3] у нейронной сети воз-

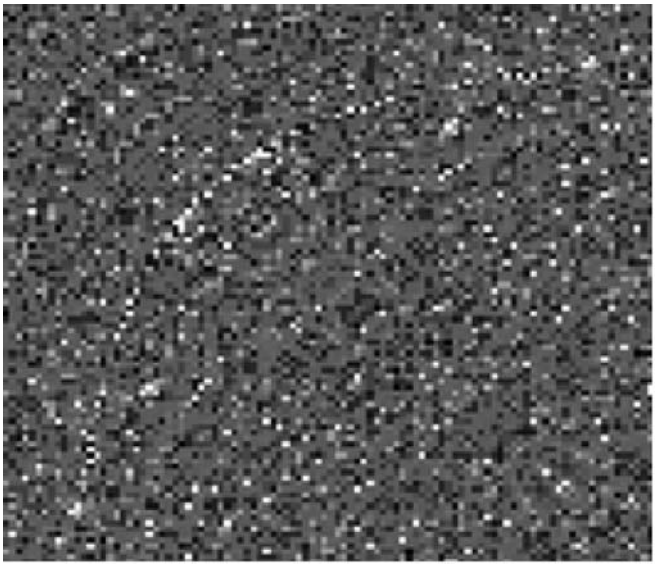


Рис. 5. Обучающее множество в объеме 10 000 пикселей

можно проявление свойства обобщения в пользу этого кластера. Это связано с тем, что пиксели остальных цветов, за исключением зеленого, достаточно равномерно представлены в фоне точек "зеленого" кластера. Такое множество может претендовать на роль обучающего.

В качестве нейронной сети для решения задачи кластеризации был выбран нейронный слой Кохонена [3]. Это связано с тем, что для заданных точек из первого этапа неизвестна их принадлежность к кластерам и необходимо, чтобы нейронная сеть без информации об их принадлежности провела бы процесс кластеризации. Процесс самообучения нейронного слоя позволит ему самостоятельно сгруппировать пиксели в кластеры. Нейронный слой Кохонена является разновидностью нейронных сетей и относится к классу самообучающихся вычислительных систем [3]. Сеть состоит из единственного слоя нейронов. Число нейронов в слое предполагается известным априори. Нейроны играют роль моделей центроидов для искомых кластеров [1], а весовые коэффициенты между всеми входными элементами и текущим нейроном — его координат в пространстве входных данных.

Данный метод обучения называется "победитель забирает все" [3]. Суть метода заключается в корректировке весовых коэффициентов нейрона, который наиболее близок входным данным. Последнее означает, что найден нейрон-победитель, являющийся центроидом. После самообучения сети каждый нейрон занимает позицию, сумма расстояний от которого до всех точек соответствующего ему кластера будет минимальной [1]. Предположив, что число кластеров равно числу разных цветов (см. рис. 1), образующих графические контуры, число нейронов в слое выберем равным 10 единицам. Изображение представлено в полноцветном формате и поэтому в ячейках массивов данные записаны числами из диапазона [0; 255]. Так как в массиве tr пиксели из генеральной совокупности gnrl сохраняют свое представление, то приходится их преобразовывать в матрицу полноцветного изображения и полученную матрицу транспонировать. Графическое представление обучающего множества было приведено на рис. 5.

Особенностью нейронной сети является число эпох для каждого образа, равное 20. Посредством проведенных опытов было установлено, что этого достаточно для того, чтобы нейронный слой смог начать выделять 10 кластеров. Ниже приводится код создания и настройки сети:

```
net=newc([0 255; 0 255; 0 255],10);
net.trainParam.epochs=20;
```

Далее идет процесс обучения, который происходит согласно алгоритму trainc:

```
for i = 1:mbuf;
net = train(net,buf(:,i));
end;
```

Следует заметить, что процесс обучения нейронной сети протекает в 10 000 эпох. В каждой из них сеть обучается запоминанию одного набора данных, представляющего один пиксель из обучающего множества. Каждый пиксель запоминается, в свою очередь, в течение 20 эпох обучения. Весь процесс занимает около 12 мин процессорного времени при его низкой степени загрузки на четырехядерной машине. Обучение нейронной сети было завершено успешно и получены следующие весовые коэффициенты. Результаты представлены в таблице в формате float для вещественных чисел и в формате uint8 для однобайтовых целых чисел (строки матрицы соответствуют весовым коэффициентам отдельных нейронов).

Согласно определению тестовое множество используется для оценки конечной архитектуры нейронной сети и только один раз [3]. Забегая вперед, отметим, что в той нейронной сети, которая была использована для решения задачи, известно число

Весовые коэффициенты нейронного слоя Кохонена

Формат float			Формат uint8		
Red	Green	Blue	Red	Green	Blue
38,0216	128,0837	9,4798	38	128	9
58,9693	181,0423	193,5376	58	181	193
60,5176	128,3270	120,9728	60	128	120
70,9440	107,2786	67,0622	70	107	67
85,6713	107,5601	115,5388	85	107	115
184,3834	162,1068	211,2053	184	162	211
57,8055	123,9877	38,0343	57	123	38
26,5413	64,2450	21,8369	26	64	21
48,4041	60,6533	69,8395	48	60	69
46,3844	131,2323	22,7827	46	131	22

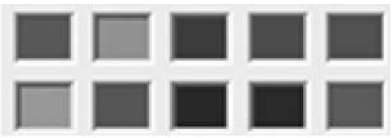


Рис. 6. Цветовая интерпретация весовых коэффициентов нейронного слоя Кохонена после обучения его на 10 000 пикселей по 20 эпох

кластеров, и поэтому не требуется подбор его архитектуры [3]. Следовательно, нет необходимости в использовании различных контрольных множеств. Оно будет единственным и в то же время будет исполнять роль тестового множества. Качество обучения можно оценить и по изображению весовых коэффициентов нейронного слоя, что и было выбрано в роли метода оценки (рис. 6). Суть его заключается в визуальном сравнении цветового представления весовых коэффициентов с цветами областей интереса в исходном изображении.

Сравнивая эти цвета с областями исходного изображения (см. рис. 1), приходим к следующим выводам:

- выделены отдельные кластеры для областей 1, 2 и 3 в количестве 5 нейронов. Данный факт подтверждает свойство обобщения нейронной сети в пользу "зеленого цвета", которое преобладало в обучающем множестве. Последнее утверждение означает, что нейронная сеть смогла уловить наличие оттенков на изображении;
- для областей 48 и 52 система смогла выделить только один кластер. Возможно, при достаточной продолжительности этого процесса этот кластер был бы разделен;
- аналогично, для областей 66 и 67 был создан только один кластер. Это объясняется их схожестью;
- для областей 89 и 77 нейронная сеть смогла выделить отдельные нейроны несмотря на то, что данные этих областей занимают малую долю в обучающем множестве. Данное обстоятельство можно объяснить тем, что они представляют оттенки других цветов, которые весьма удалены от остальных, присутствующих на изображении;
- так как пикселей "черного" цвета тоже достаточно много, нейронная сеть выделила отдельный нейрон в пользу этого кластера;
- к сожалению, кластера "желтого" цвета не оказалось. Это связано с тем, что в обучающем множестве отсутствовали данные об этих пикселях.

Результат распознавания зависит от размера обучающего множества и от его содержимого [1]. Недостаток данных может привести к получению очень грубой модели, а избыток данных — увеличить время обучения нейронной сети [3]. Другая ситуация заключается в представлении данных, образующих обучающее множество, с точки зрения их достаточности. Любое преобладание одного типа данных может стать причиной переобучения или искажения результата в их пользу [3]. Избыток нейронов в архитектуре нейронной сети может привести к огрублению вычислительной системы. Но даже при таком числе ограничений правильно обученная нейронная сеть может эффективно решать проблему классификации.

Для применения обученной нейронной сети к нашему изображению мы использовали следующий алгоритм:

1. Выбрать пиксель из исходного изображения.
2. Параметры текущего пикселя подать на вход нейронной сети.
3. Из отклика нейронной сети определить номер нейрона-победителя.
4. Для текущего пикселя определить число смежных пикселей одинакового цвета и выбрать цвет с преобладающей численностью.
5. По номеру нейрона-победителя и номеру строки и столбца пикселя заменить его весовыми коэффициентами этого нейрона, если весовые коэффициенты описывают "преобладающий цвет", в противном случае отклонить эти весовые коэффициенты и принять в качестве цвета пикселя цвет смежных пикселей.
6. Если текущий пиксель является последним, то остановиться, иначе — повторить шаги 1—5 для следующего пикселя.

Применяя выше описанный алгоритм, удалось получить достаточно хороший результат (рис. 7).

Полученное после обработки нейронной сетью изображение является промежуточным результатом. Для проведения дальнейших операций нужно параллельно с нейронной сетью для обрабатываемого изображения создавать матрицу меток. Суть матрицы меток заключается в том, что для одного цвета пикселей соответствующие ячейки

Рис. 7. Результат обработки исходного изображения

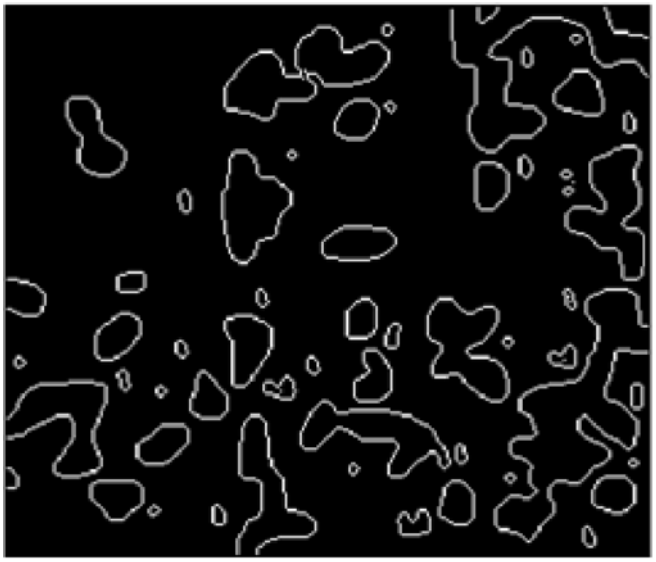


Рис. 8. Результат операции выделения границ командой edge среды MATLAB

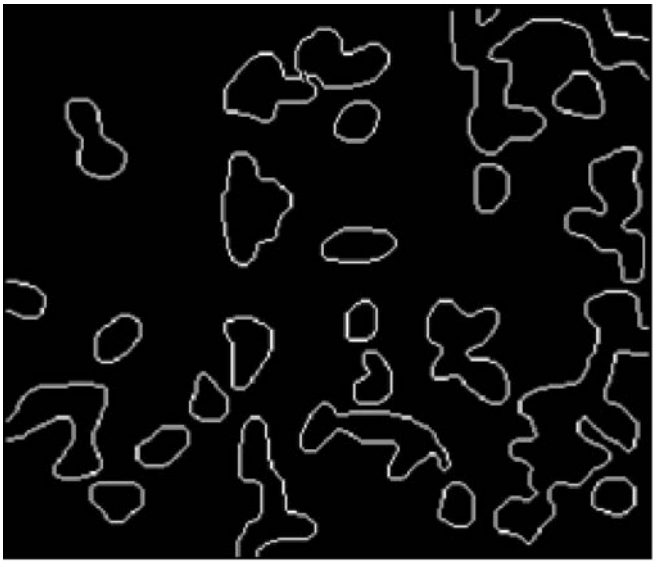


Рис. 9. Пороговая обработка контуров по периметру

этой матрицы получают определенные числовые метки [2, 6]. По указанным меткам можно исключать из исходного изображения ненужные паттерны и оставлять интересующие области. Таким образом, можно добиться "расслоения" первоначального изображения, где каждый слой будет содержать области только одного цвета. В дальнейшем, обработав каждый из этих слоев командой edge [2, 6], можно для них получить требуемые границы (рис. 8).

Результат (рис. 8) содержит контуры не только интересующих областей, но и шумовых областей. Далее применили пороговую операцию с учетом периметров границ этих паттернов в целях очистки от мелких областей. Таким образом, мы получили контуры интересующих паттернов (рис. 9). Применение этого метода возможно и для каждого слоя по отдельности.

Заключение

Разработан эффективный алгоритм фильтрации цветных пикселей на основе искусственной нейронной сети. Алгоритм использует замену компонента пикселя весовыми коэффициентами нейрона-победителя. Данный подход можно использовать для предварительной обработки картографических данных и их подготовки к векторизации контуров.

Список литературы

1. **Горелик А. Л., Скрипкин В. А.** Методы распознавания: учеб. пособие для вузов. 4-е изд., исправ. М.: Высш. шк., 2004. 261 с.
2. **Дьяконов В.** MATLAB. Обработка сигналов и изображений. Специальный справочник. СПб.: Питер, 2002. 608 с.
3. **Медведев В. С., Потемкин В. Г.** Нейронные сети. MATLAB 6 / Под общ. ред. В. Г. Потемкина. М.: ДИАЛОГ-МИФИ, 2002. 496 с.
4. **Мещеряков В. В.** Задачи по статистике и регрессионному анализу с MATLAB. М.: Диалог-МИФИ, 2009. 448 с.
5. **Форсайт Д., Понс Ж.** Компьютерное зрение: современный подход / Пер. с англ. М.: Издат. дом "Вильмяс", 2004. 928 с.
6. http://matlab.exponenta.ru/imageprocess/index.php

ИНФОРМАЦИЯ

УДК 004.322

Н. М. Новикова, д-р техн. наук, проф.,
В. Г. Ляликова, аспирант,
e-mail: vikalg@yandex.ru,
Воронежский государственный университет

Математические модели параметрических статистических и нейросетевых обнаружителей сигналов при наличии шума и импульсной помехи

Рассматривается математическое моделирование обнаружителей сигналов, использующих алгоритм Байеса, Неймана—Пирсона, двухслойного персептрона и нейронной сети РБФ. Рассмотрены методы обучения представленных нейронных сетей. Приведены результаты вычислительного эксперимента. Проведен сравнительный анализ качества обнаружения сигналов рассмотренными методами при наличии гауссовского шума и импульсной помехи.

Ключевые слова: *обнаружение сигналов, статистические алгоритмы, нейросетевые алгоритмы, моделирование, сравнительный анализ*

Введение

Информационные технологии широко применяют при выполнении функций сбора и обработки радиолокационной информации. Обработка радиолокационной информации состоит из нескольких этапов, важнейшим из которых является обнаружение сигналов. На этом этапе проявляются наибольшие трудности автоматизации процессов обработки радиолокационной информации. В настоящее время используют основной подход к построению автоматических обнаружителей — это синтез параметрических обнаружителей. При синтезе параметрических обнаружителей предполагается, что, имея набор некоторого семейства вероятностных распределений, возможно построить оптимальные обнаружители на основе статистической теории принятия решений. Однако сделать обоснованный выбор распределений обычно не удается, что приводит к снижению качества обнаружителей в реальных условиях. Вместо рассмотренного классического подхода при синтезе обнаружителей представляется более целесообразным использовать нейросетевую технологию.

Данная технология дает возможность компьютерной системе обучаться на примерах. Искусственные нейронные сети позволяют получать решения многих проблем, ранее считавшихся неразрешимыми. При этом достигается гибкость и адаптивность работы, робастность, способность к обобщению.

В настоящее время известно достаточно большое число исследований [1], в которых рассматривается возможность создания нейросетевых обнаружителей. В статье [2] представлены результаты работы устройств обнаружения и оценки параметров сигнала нейросетевыми методами. Особенностью этой и многих других работ является отсутствие оценок уровня ложных тревог, что не позволяет судить о качестве полученных обнаружителей. Кроме того, исследуется обнаружение сигнала только на фоне гауссовского шума, хотя на практике обнаружитель часто работает в условиях воздействия не только шума, но и импульсных помех.

Целью статьи является создание математических моделей статистических и нейросетевых обнаружителей сигнала и проведение сравнительного анализа вероятностных характеристик этих обнаружителей.

Постановка задачи. Задачу обнаружения сигнала можно представить следующим образом. Пусть на входе системы наблюдается реализация случайного процесса $x(t)$, которая может быть только шумом $x(t) = n(t)$, или суммой сигнала и шума $x(t) = s(t) + n(t)$. Необходимо определить алгоритм обработки наблюдаемого процесса $x(t)$ и характеристики алгоритма. Посредством этого алгоритма выносится решение о наличии или отсутствии сигнала в наблюдаемом процессе. Решение рассмотренной задачи может быть осуществлено как статистическими методами, так и с помощью нейронных сетей.

Теоретический анализ

Математическая модель обнаружения сигнала на фоне шума формулируется как задача проверки статистических гипотез. Проверке подлежит гипотеза H_0: $x(t) = n(t)$ против альтернативы H_1: $x(t) = s(t) + n(t)$. Теперь определение алгоритма обнаружения сводится к отысканию правила выбора решения по наблюдаемым данным $x(t)$ в пользу одной из гипотез H_0 или H_1. При известных распределениях вероятностей случайных процессов при гипотезах H_0 и H_1 теорема Неймана—Пирсона дает оптимальный обнаружитель [3]. Этот обнаружитель основан на сравнении функционала отношения правдоподобия с некоторым порогом. Выбор порога зависит от наличия априорной информации о характеристиках случайных процессов, наблюдаемых на входе обнаружителя.

Нерандомизированное решающее правило можно описать как некоторую критическую область G в пространстве векторов входных сигналов. При

попадании входного сигнала в эту область принимается решение о наличии искомого объекта. В противном случае принимается решение об его отсутствии [4].

Алгоритм принятия решений о выборе между двумя гипотезами H_0 и H_1 оценивается следующими вероятностными характеристиками [5]:

- вероятностью ошибки ложного обнаружения α (шум превосходит порог C и принимается решение о наличии сигнала, хотя его нет):

$$\alpha = \int_C^{\infty} \omega_n(u)du,$$

где u — напряжение на входе; $\omega_n(u)$ — плотность распределения вероятностей распределения шума;

- вероятностью ошибки пропуска сигнала β (сигнал присутствует, но не превышает порог C)

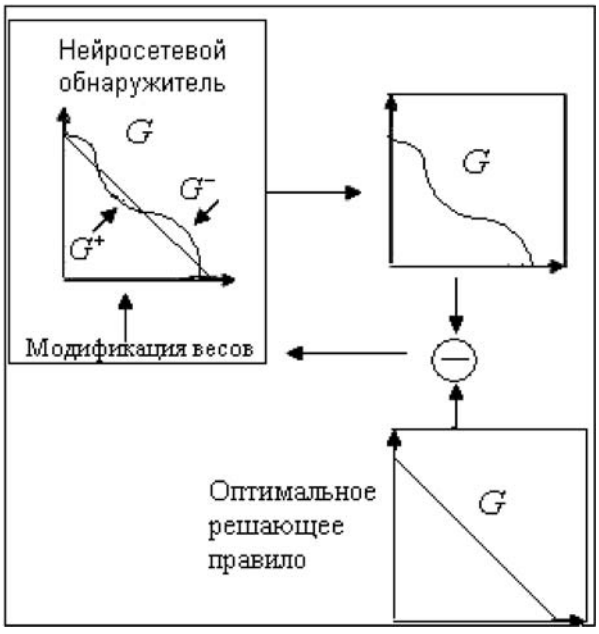
$$\beta = 1 - \int_C^{\infty} \omega_{sn}(u)du,$$

где $\omega_{sn}(u)$ — плотность распределения вероятностей распределения смеси сигнала и шума;

- вероятностью правильного обнаружения D

$$D = 1 - \beta = \int_C^{\infty} \omega_{sn}(u)du.$$

Отсюда следует, что необходимо найти такой алгоритм обработки принятого сигнала, который минимизировал бы ошибку ложного обнаружения и максимизировал вероятность правильного обнаружения.



Обучение нейросетевого обнаружителя

В практических задачах вид плотностей распределений для случаев наличия сигнала и его отсутствия заранее неизвестен. При этом одним из вариантов построения процесса оптимизации является создание системы, обучающейся на примерах, представляющих собой реализации реальных случайных величин из заданных классов.

На ввод системы последовательно подаются обучающие примеры, для каждого из которых сообщается, к какому классу он принадлежит. Решения, принимаемые системой обнаружения по обучающим примерам, могут быть как правильными, так и неправильными. В случае неправильных решений внутренние параметры системы модифицируются таким образом, чтобы приблизиться к идеальной системе, т. е. когда все ее решения оказываются верными. Такой механизм реализуется в нейронных сетях [8]. Обучение нейросетевого обнаружителя можно представить с помощью схемы, показанной на рисунке, где G — область принятия решения.

Статистические алгоритмы

Байесовский алгоритм. При наличии полной априорной информации о сигнале и шуме используют критерий среднего риска (байесовский критерий). Оптимальное байесовское правило обнаружения основывается на минимизации среднего риска [6], определяемого формулой

$$R = \sum_{j=0}^{1} \sum_{i=0}^{1} \Pi_{ji} p_j \int_{x_i} W_n(x|H_j)dx,$$

где x_i — область принятия решения;

$$\text{матрица потерь } \Pi = \begin{pmatrix} \Pi_{00} & \Pi_{01} \\ \Pi_{10} & \Pi_{11} \end{pmatrix}; \qquad (1)$$

$p_1 = 1 - p_0$ — априорная вероятность наличия сигнала; p_0 — априорная вероятность отсутствия сигнала;

$W_n(x|H_j)$ — условная плотность распределения вероятностей (функция правдоподобия) наблюдаемой выборки в предположении, что верна гипотеза H_j.

Байесовский алгоритм обнаружения сводится к сравнению с порогом функционала отношения правдоподобия

$$L(x) = \frac{W_n(x|H_1)}{W_n(x|H_0)}. \qquad (2)$$

Значение порога определяется формулой

$$C = \frac{\Pi_{01} - \Pi_{00}}{\Pi_{10} - \Pi_{11}} \frac{p_0}{p_1}. \qquad (3)$$

Принимается решение γ_1 (отклоняется гипотеза H_0, т. е. сигнал присутствует), если $L(x) \geqslant C$, и принимается решение γ_0 (принимается гипотеза H_0, т. е. наблюдается только шум), если $L(x) < C$.

Критерий Неймана—Пирсона. При использовании данного критерия фиксируется на определенном уровне вероятность ложного обнаружения и выбирается такое правило решения, при котором вероятность пропуска сигнала имеет минимальное значение.

Таким образом,

$$\alpha = \int_C^{\infty} \omega_n(u)du \leqslant \text{const};$$

$$\beta = \beta_{\min} = \min\left[1 - \int_C^{\infty} \omega_{sn}(u)du\right].$$

Для использования критерия Неймана—Пирсона должна быть задана вероятность ложных тревог α и известны плотности распределения вероятностей $\omega_n(n)$, $\omega_{sn}(n)$, а следовательно, и соответствующие законы распределения [5].

Нейросетевые алгоритмы

Двухслойный персептрон. Персептрон представляет собой двухслойную нейронную сеть без обратных связей (входной слой является первым слоем и выходной — вторым). Данный персептрон с одним выходом работает следующим образом. На его входы поступают входные сигналы, проходящие по синапсам и образующие выходные сигналы. Данный процесс, происходящий в нейронной сети, может быть записан в следующей матричной форме:

$$Y = F(F(XW)V),$$

где X и Y — соответственно входной и выходной сигнальные векторы; W — весовые коэффициенты первого слоя; V — весовые коэффициенты второго (выходного) слоя; $F(s)$ — активационная функция, применяемая поэлементно к компонентам вектора S, $S = XW$.

Обучение персептрона происходит по правилу обучения Хебба и сводится к формированию весов связей между первым и вторым слоями по формуле [7]

$$w_{ij}(p + 1) = w_{ij}(p) + \eta y_j^{(1)} y_i^{(2)}, \qquad (4)$$

где $w_{ij}(p + 1)$ — значение веса от нейрона i к нейрону j после подстройки; $w_{ij}(p)$ — значение веса от нейрона i к нейрону j до подстройки; $y_j^{(1)}$ — выходное значение j-го нейрона слоя 1; $y_i^{(2)}$ — выходное значение i-го нейрона слоя 2; w_{ij} — весовой коэффициент синапса, соединяющего эти нейроны; η — коэффициент скорости обучения; p — шаг итерации.

Нейронная сеть РБФ. Под нейронной сетью с радиально-базисными функциями (РБФ) активации, или РБФ-сетью, понимается двухслойная сеть без обратных связей, которая содержит слой скрытых нейронов с радиально симметричной активационной функцией, каждый из которых предназначен для хранения отдельного эталонного вектора (в виде вектора весов) [8]. Процесс обучения сети РБФ с учетом выбранного типа радиально-базисной функции сводится:

- во-первых, к подбору эталонов, представленных в виде весовых векторов нейронов скрытого слоя;
- во-вторых, к подбору центров r_i по формуле

$$r_i(t + 1) = r_i(t) + \alpha_t(x_j(t) - r_i(t)), \qquad (5)$$

где $r_i(t + 1)$ — значение центра i в момент времени $t + 1$; $r_i(t)$ — значение центра i в момент времени t; α_t — темп обучения,
и параметров отклонений σ_i от формы базисных функций:

$$\sigma_i = \sqrt{\left(\sum_{k=0}^{L-1} r_i - r_k\right)^2}; \qquad (6)$$

- в-третьих, к подбору весов нейронов выходного слоя w_{ij} по формуле (4).

Экспериментальная часть

Моделирование на компьютере. Для реализации математических моделей обнаружителей на компьютере задача может быть поставлена следующим образом. Пусть в течение некоторого времени $[0, T]$ на вход системы поступает реализация случайного процесса

$$x = s + \xi, \qquad (7)$$

которая может быть либо шумом, либо суммой полезного сигнала и шума. Здесь ξ — либо реализация аддитивной гауссовской помехи с нулевым математическим ожиданием и единичной дисперсией, либо смесь этой же помехи и хаотической импульсной помехи.

Необходимо определить алгоритм обработки наблюдаемого процесса $x(t)$ и характеристики алгоритма. В теории статистического обнаружения доказано, что характеристики обнаружителя не зависят от формы сигнала, т. е. от вида функции $s(t)$, а определяются отношением сигнал/шум [2]. Используя это положение, для сокращения вычислительных затрат и большей наглядности ре-

зультатов будем рассматривать квазидетерминированный сигнал — функцию вида

$$s(t) = A\mathbf{e}^{\frac{-(t-t_0)}{2}},$$

где A — амплитуда сигнала; t_0 — время прихода сигнала — неизвестный параметр, распределенный по равномерному закону на интервале $[0, T]$; t — текущий момент времени.

Моделирование сигнала и шума. В качестве шума использовался нормальный случайный процесс, который моделировался методом скользящего суммирования [9].

Импульсный случайный процесс можно описать следующим образом [10]:

$$\xi(t) = \sum_{t=-\infty}^{\infty} A_i f_i(t - t_i),$$

где $f_i(t - t_i) = \begin{cases} 1, & t_i \leqslant t < t_i + \tau_i; \\ 0, & \text{при других } t; \end{cases}$

τ_i — длительность импульса.

Для моделирования хаотической импульсной помехи сначала формировалась последовательность с равномерным распределением на интервале $[-1, 1]$. Далее преобразуем эту последовательность в $y(t)$ таким образом, чтобы обеспечить постоянство значений на протяжении всей длительности импульса τ. На основе $y(t)$ по правилу

$$\xi(t) = \begin{cases} 1, & y(t) > 0; \\ 0, & y(t) < 0 \end{cases}$$

формировали импульсную последовательность со случайным генерированием единичных и нулевых посылок. Затем, применяя амплитудную модуляцию, получили амплитудно-манипулированную хаотическую импульсную помеху.

Сигнал моделировался как случайная величина с равномерным распределением времени появления и квазидетерминированной амплитудой для 100 предъявлений. Процесс (7) получался аддитивным сложением реализации сигнала $s(t)$ и реализации шума $\xi(t)$.

Моделирование статистических обнаружителей. Моделирование байесовского обнаружителя проводили при равных априорных вероятностях наличия и отсутствия сигнала, а платы за принятые решения имели следующие значения: $\Pi_{00} = 0{,}4$, $\Pi_{01} = 1{,}8$, $\Pi_{10} = 3$. $\Pi_{11} = 1$. Формировались функции:

$L_s(t) = s(t) + \xi(t)$ — при наличии сигнала;

$L_\xi(t) = \xi(t)$ — при отсутствии сигнала.

Реализации $L_s(t)$ и $L_\xi(t)$ сравнивались с порогом (3). Подсчитывались вероятности ложной тревоги $\alpha = \mathrm{P}[L_\xi(t) > C]$ и пропуска сигнала $\beta = P[L_s(t) < C$, а также вероятности правильного обнаружения $P_{\text{обн}} = P[L_s(t) > C]$ при различных отношениях сигнал/шум. Эти вероятности подсчитываются как частота рассмотренных событий по числу предъявлений.

Моделирование обнаружителя Неймана—Пирсона проводилось таким же образом, как и обнаружителя Байеса, за исключением выбора порога C. Для его определения сначала необходимо выбрать уровень ложных тревог α, а затем по значению α, используя условие

$$\alpha = P[L_\xi(t) \leqslant C] = \int_C^\infty L_\xi(t)dt \leqslant C,$$

определить порог C. Значения ложных тревог устанавливались на уровне 10^{-8} [4].

Моделирование нейросетевых обнаружителей. Рассмотренные нейронные сети имели число входов, равное числу предъявляемой выборки (100), т. е. числу значений в наблюдаемой реализации. Нейронная сеть РБФ имела два выхода (один выход отвечает за присутствие сигнала, второй — за присутствие только помехи), а двухслойный персептрон имел один выход, который затем сравнивался с порогом ($T = 1$) для определения результата работы сети.

Обучение нейронной сети РБФ и двухслойного персептрона проходило следующим образом. Данным сетям на вход случайным образом предъявлялись обучающие выборки, содержащие либо гауссовский шум, либо смесь гауссовского шума и сигнала, т. е. обучающие выборки не содержали информации о хаотической импульсной помехе. Выборка состояла из 100 обучающих примеров. В процессе предъявления на вход сети обучающих примеров определялся выход сети. В зависимости от полученного значения выхода весовые коэффициенты либо изменялись, либо нет. Изменение весовых коэффициентов происходило в соответствии с алгоритмами обучения. Если веса перестали изменяться, значит для данного примера сеть обучена. Обучение проводится до тех пор, пока весовые коэффициенты не застабилизируются с точностью до 10^{-10}.

Обучение двухслойного персептрона сводилось к расчету 10 000 весовых коэффициентов первого слоя и 100 значений коэффициентов второго слоя по формуле (4). В общей сложности потребовалось настроить 10 100 значений весовых коэффициентов. В качестве активационной функции использовалась сигмоидальная логистическая функция $F(x) = 1/(1 + \mathbf{e}^{-x})$. Данная функция обладает свойством усиливать слабые сигналы и предотвращает от насыщения при больших сигналах.

Обучение нейронной сети РБФ проводилось следующим образом. Скрытый слой обучался с помощью алгоритма Кохонена [7]. Весовые коэффици-

енты выходного слоя сети оптимизировались по методу обучения Хебба, которым обучался и двухслойный персептрон. В качестве эталонных значений в первый образец записывались значения сигнала, а во второй — значения гауссовского шума. Для обучения сети потребовался подбор 200 значений эталонов, настройка двух значений центров r_i по формуле (5), одного значения отклонения σ_i с использованием евклидова расстояния по формуле (6) и четырех значений весовых коэффициентов второго выходного слоя по формуле (4).

Результаты

В результате проведения вычислительного эксперимента получены вероятности правильного обнаружения, вероятности ложных тревог и вероятности пропуска сигнала в зависимости от отношения сигнал/шум. Эти вероятности рассчитывались как частота события по 1000 реализациям для каждого значения сигнал/шум. Основные результаты эксперимента представлены в табл. 1—4. В таблицах приведены значения, округленные до второго порядка.

Рассматривая вероятности правильного обнаружения относительно гауссовского шума, можно заметить, что наличие хаотической импульсной помехи для всех обнаружителей привело к ухудшению результатов. В табл. 4 представлены результаты процентного уменьшения вероятности правильного обнаружения при наличии хаотической импульсной помехи, а также при наличии аддитивной смеси гауссовского шума и импульсной помехи. Сравнение проведено с вероятностями правильного обнаружения, полученными при наличии только гауссовского шума.

Относительно уровня ложных тревог можно заметить, что для оптимальных статистических алгоритмов и для алгоритма сети РБФ он остается на фиксированном низком уровне 10^{-8}. У алгоритма двухслойного персептрона уровень ложных тревог также находится на фиксированном низком уровне 10^{-4}.

Таблица 1

Зависимость вероятности правильного обнаружения от отношения сигнал/шум при гауссовском шуме

Алгоритм	Отношение сигнал/шум								Среднее
	0,001	0,20	0,60	1,00	1,40	2,00	3,40	4,00	
Статистический алгоритм									
Байеса	0,00	0,46	0,63	0,76	0,81	0,90	0,99	0,99	0,846
Неймана—Пирсона	0,00	0,21	0,43	0,47	0,58	0,62	0,78	0,81	0,632
Нейросетевые алгоритмы									
Двухслойный персептрон	0,004	0,96	0,99	0,99	0,99	0,99	0,99	0,99	0,963
Сеть РБФ	0,00	0,67	0,92	0,98	0,99	0,99	0,99	0,99	0,937

Таблица 2

Зависимость вероятности правильного обнаружения от отношения сигнал/шум при наличии хаотической импульсной помехи

Алгоритм	Отношение сигнал/шум								Среднее
	0,001	0,20	0,60	1,00	1,40	2,00	3,40	4,00	
Статистический алгоритм									
Байеса	0,00	0,40	0,66	0,73	0,80	0,89	0,99	0,99	0,843
Неймана—Пирсона	0,00	0,17	0,40	0,54	0,62	0,67	0,72	0,80	0,628
Нейросетевые алгоритмы									
Двухслойный персептрон	0,06	0,78	0,95	0,99	0,99	0,99	0,99	0,99	0,96
Сеть РБФ	0,00	0,57	0,82	0,91	0,96	0,99	0,99	0,99	0,915

Таблица 3

Зависимость вероятности правильного обнаружения от отношения сигнал/шум при наличии смеси гауссовского шума и импульсной помехи

Алгоритм	Отношение сигнал/шум								Среднее
	0,001	0,20	0,60	1,00	1,40	2,00	3,40	4,00	
Статистический алгоритм									
Байеса	0,00	0,43	0,59	0,74	0,82	0,87	0,99	0,99	0,842
Неймана—Пирсона	0,00	0,17	0,39	0,50	0,58	0,64	0,79	0,78	0,622
Нейросетевые алгоритмы									
Двухслойный персептрон	0,00	0,72	0,88	0,95	0,99	0,99	0,99	0,99	0,952
Сеть РБФ	0,00	0,54	0,82	0,88	0,95	0,99	0,99	0,99	0,906

Таблица 4

Результаты процентного уменьшения вероятности правильного обнаружения

Алгоритм	Хаотическая импульсная помеха	Смесь гауссовского шума и хаотической импульсной помехи
Статистический алгоритм		
Байеса	0,3 %	0,4 %
Неймана—Пирсона	0,4 %	1,0 %
Нейросетевые алгоритмы		
Двухслойный персептрон	1,2 %	2,8 %
Сеть РБФ	2,2 %	3,1 %

Рассмотрев результаты работы представленных алгоритмов, можно увидеть, что оптимальным образом работают алгоритм Байеса и алгоритм нейронной сети РБФ, причем у алгоритма сети РБФ значения вероятности правильного обнаружения выше, чем у Байеса в среднем на 7,6 %. Необходимо выявить, значима ли разница между вероятностями правильного обнаружения, полученными в вычислительном эксперименте с моделями Байеса и нейронной сетью РБФ. Выдвигается гипотеза, что вероятности получены по выборкам, извлеченным из одной генеральной совокупности. Для проверки этой гипотезы используем критерий однородности двух выборок χ^2. Применение этого статистического критерия подтверждает, что алгоритм нейронной сети РБФ работает как оптимальный алгоритм Байеса при уровне значимости 0,05.

Заключение

В данной статье представлены результаты математического моделирования алгоритмов Байеса, критерия Неймана—Пирсона, а также обнаружителей сигнала на основе нейронных сетей: двухслойного персептрона и сети РБФ. Проведен сравнительный анализ работы этих обнаружителей при гауссовском шуме, хаотической импульсной помехе и смеси гауссовского шума и хаотической импульсной помехи. Анализ показал, что из рассмотренных нейросетевых алгоритмов только нейронная сеть РБФ может оптимальным образом обнаруживать сигнал как при наличии гауссовского шума, так и при наличии импульсной помехи, так же как алгоритм Байеса. Полученные результаты могут быть использованы в современных технологиях обработки радиолокационной информации.

Список литературы

1. **Галушкин А. Н.** Нейрокомпьютеры в разработках военной техники США (обзор по материалам открытой печати) // Зарубежная радиоэлектроника. 1995. № 5.
2. **Перов А. И., Соколов Г. Г.** Особенности синтеза устройств обнаружения и оценки параметров сигнала нейросетевыми методами // Радиотехника. 2001. № 7. С. 22—29.
3. **Лагутин М. Б.** Наглядная математическая статистика. М.: БИНОМ. Лаборатория знаний. 2007. 472 с.
4. **Татузов А. Л.** Методы обучения нейронных сетей для решения задач обнаружения целей // Нейрокомпьютеры: разработка и применение. 2004. № 4. С. 56—67.
5. **Акимов П. С., Евстратов Ф. Ф., Захаров С. И.** и др. Обнаружение радиосигналов / Под. ред. А. А. Колосова. М.: Радио и связь, 1989. 288 с.
6. **Леви Б. Р.** Теоретические основы статистической радиотехники. М.: Радио и связь, 1989. 656 с.
7. **Круглов В. В., Дли М. И., Голунов Р. Ю.** Нечеткая логика и искусственные нейронные сети. М.: Физматлит, 2001. 224 с.
8. **Круглов В. В., Борисов В. В.** Искусственные нейронные сети. Теория и практика. М.: Горячая линия—Телеком, 2002. 352 с.
9. **Быков В. В.** Цифровое моделирование в статистической радиотехнике. М.: Советское радио, 1971. 328 с.
10. **Тихонов В. И.** Статистическая радиотехника. М.: Советское радио, 1982. 624 с.

CONTENTS

The package of methods and programs is proposed that provides minimization of Boolean functions in disjunctive normal form (DNF) according to criterions of complexity and power dissipation of corresponding two-level AND-OR CMOS-circuits. The methods are extensions of known methods of Boolean function minimization by adding heuristics that turn the minimization process towards lowering the power dissipation in the sought CMOS-circuits. The results of computer experiments are given, which allow to evaluate power driven minimization influence on power dissipation of resulting CMOS-circuits.

Keywords: logical design, minimization of Boolean functions, power consumption, CMOS-technology

Questions of introduction of Web-technologies in systems of the automated designing are considered. The comparative estimation of possible ways of construction of the software of systems of modeling with use of technology of Web-services is resulted. The description of realization of the software distributed system of circuitry designing is given.

Keywords: systems of the automated designing, system of modeling, circuitry designing, Web-technologies, Web-services

In this paper a method of job-flow scheduling in distributed computing environment is proposed and examined. The method involves an economical model to increase overall resource distribution efficiency.

Keywords: distributed computing, metascheduler, resource distribution, job scheduling, time and cost optimization

The appropriateness of introduction of context-dependent elements providing interactive access of the user to commands and properties of the chosen object into graphical interfaces of computer programs is proved. Variants of development and expansion of application sphere of the given elements on an example of MS Word text editor are offered, the review of existing analogues is performed, ergonomics and efficiency of performed workings out are substantiated.

Keywords: graphical interface, optimization of graphical interfaces, graphical interfaces design, software ergonomics, human-computer interaction, contextual elements of the interface

The developed architectural solution of distributed system aimed for digital graphic and animation creation is considered.

Keywords: distributed system, graphics, animation

In the article presents structure of system of recognition and control of chaotic processes, which ideological basis is nonlinear recurrence analysis. Paradigm of control of process use Chirikov, Lozi and Ikeda maps. The recurrence diagrams of new chaotic systems Chirikov+Ikeda and Lozi+Ikeda is derived.

Keywords: chaotic processes, nonlinear recurrence analysis, Chirikov, Lozi and Ikeda maps

The given work presents the method of model generation of complex system analysis. The complex system is a researching object with characterized collection factors. This method is based on combination of the multifactor nonlinear regression model, the group method of data handling, the parallel genetic algorithm of significant factors selection and the method of artificial intelligence. This approach is a good way for receiving most qualitative set of factors. This method is considered for model generation analysis of anindustrial organization.

Keywords: complex systems, non-linear regression analysis, self-organization, recursive process, parallel genetic algorithm, feature selection

The paper discusses the main provisions of algorithms and optimization of simplicial complexes on the quasi-stationary grids. These algorithms can be used in FPU for the software-hardware complex "Topological processor". The proposed algorithms include the methods of constructing and optimizing the triangulation on the grids. The method of quality assurance and optimization based on minimizing the integral of a quadratic norm on the simplexes, and as a criterion to be considered the value of solid angle at the vertices of simplexes of the subdivision.

Keywords: triangulation, grid, optimization, delaney, simplex

The papers describes some results of numerical modeling of mechanical factors of cranial-cerebral injury caused be percussive stress. The usage of grid-characteristic method on unstructured grid is discussed. The authors introduce mechanical-mathematical model of human skull and brain and compare results of numerical modeling based on this model and results observed in clinics. The problem is topical for detailed analysis of mechanisms of brain damage during cranial-cerebral injury.

Keywords: numerical models, cranial-cerebral injury, grid-characteristic method, unstructured grid

It is presented the description and comparative analysis of neural network algorithms of evaluation of granulometry properties friable materials in Mining Industry.

Keywords: neural networks, pattern recognition, granulometry, mining

Color image pixels filtering technique using a neural network is discussed. The program MATLAB containing Neural Network Toolbox and Image Processing Toolbox was used. The proposed approach allows us to save computing time and helps to avoid data loss under working with scanned maps.

Keywords: image processing, pixel filtering, GIS, artificial neural network, preliminary processing of the color images, clustering, edge segmentation, MATLAB the program, image processing toolbox

The computer simulation of signals detectors have been considered in this paper. Bayesian method, Neyman—Pearson criterion, RBF neural network and two-layer perseptron have been considered. The methods of education presents neural network have been examined. The computing experiment results have been produced. The comparative analysis of consideration methods work with gaussian noise and chaotic pulse noise has been realized.

Keywords: signals detection, statistical algorithms, neural network algorithms, simulation, comparative analysis

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 *М. Г. Джавадян*.
Сдано в набор 10.03.2011. Подписано в печать 18.04.2011. Формат 60×88 1/8. Бумага офсетная. Печать офсетная.
Усл. печ. л. 9,8. Уч.-изд. л. 11,10. Заказ 281. 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