Научно-исследовательский институт
«Центрпрограммсистем»

Программные продукты и системы

НАУЧНО-ПРАКТИЧЕСКОЕ ИЗДАНИЕ

№ 1 (97), 2012

Тверь

Вниманию авторов!

Международный журнал «Программные продукты и системы» публикует материалы научного и научно-практического характера по новым информационным технологиям, результаты академических и отраслевых исследований в области использования средств вычислительной техники. Практикуется выпуск тематических номеров по искусственному интеллекту, системам автоматизированного проектирования, по технологии разработки программных средств и системам защиты, а также специализированные выпуски, посвященные научным исследованиям и разработкам отдельных вузов, НИИ, научных организаций.

Решением Президиума Высшей аттестационной комиссии Министерства образования и науки РФ № 616 от 19.02.2010 международный журнал «Программные продукты и системы» внесен в Перечень ведущих рецензируемых научных журналов и изданий, в которых должны быть опубликованы основные научные результаты диссертаций на соискание ученых степеней кандидата и доктора наук.

Информация об опубликованных статьях регулярно по установленной форме представляется в систему Российского индекса научного цитирования (РИНЦ).

Условия публикации

Редакция принимает к рассмотрению материалы, соответствующие тематике журнала (05.13.00 – Информатика, вычислительная техника и управление) и ранее нигде не опубликованные.

Материалы, поступившие в редакцию, оцениваются на соответствие редакционным требованиям.

Работа представляется в электронном виде в формате Word (шрифт Times New Roman, размер 11 пунктов с полуторным межстрочным интервалом). При обилии сложных формул обязательно наличие статьи и в формате pdf. Формулы желательно набирать в редакторе формул Word (Microsoft Equation 3.0). Объем статьи вместе с иллюстрациями должен быть не менее 10 000 знаков. Просьба не присылать цветные, тонированные и не подлежащие дальнейшему редактированию в Word'е рисунки, а также отсканированные формулы и тексты. Заголовок не должен превышать 7 слов; сокращения, а также терминологию узкой тематики желательно в нем не использовать. Количество авторов на одну статью – не более 4, количество статей одного автора в номере, включая соавторство, – не более 2. Список литературы (оформленный в соответствии с ГОСТом Р 7.05–2008), наличие которого обязательно, должен включать не более 5 пунктов.

Вместе со статьей следует прислать отзыв-рекомендацию в произвольной форме и экспертное заключение.

Необходимо также представить сведения об авторах: фамилия, имя, отчество, наименование организации, должность, ученые степень и звание (если есть), контактный телефон, электронный адрес, почтовый адрес для отправки бесплатного авторского экземпляра.

Кроме того, статья должна иметь на русском и английском языках аннотацию (не более 50 слов), ключевые слова (7–10 слов), а также УДК. На английский язык необходимо перевести название статьи, имя и фамилию автора, наименование организации, ученую степень и звание (если есть).

Порядок рецензирования

Все статьи, поступающие в редакцию (соответствующие тематике и оформленные согласно требованиям к публикации), подлежат обязательному рецензированию в течение месяца с момента поступления.

В редакции есть устоявшийся коллектив рецензентов, среди которых члены международной редколлегии журнала, эксперты из числа крупных специалистов в области информатики и вычислительной техники ведущих вузов страны, а также ученые и специалисты НИИ «Центрпрограммсистем» (г. Тверь).

Рецензирование проводится конфиденциально. Автору статьи предоставляется возможность ознакомиться с текстом рецензии. При необходимости статья отправляется на доработку или отклоняется.

Рецензии обсуждаются на заседаниях редакционной коллегии, которая проводится один раз в месяц в НИИ «Центрпрограммсистем» (г. Тверь) или в Главной редакции международного журнала «Проблемы теории и практики управления» (г. Москва).

Решение о целесообразности опубликования статьи после рецензирования принимается редакционным советом.

Материалы, одобренные редакционным советом, публикуются бесплатно в течение года с момента одобрения, а статьи, отправленные на доработку, – с момента поступления после доработки. Если принятая к публикации статья, по мнению автора, является срочной, редакция вправе опубликовать ее в текущем номере на коммерческой основе.

УДК 681.3.06+519.68

ВХОДНОЙ ЯЗЫК ОБЪЕКТНО-ОРИЕНТИРОВАННОЙ БАЗЫ ЗНАНИЙ GRID-СИСТЕМЫ

(Работа выполнена при частичной поддержке РФФИ, грант № 10-07-00146)

Г.А. Опарин, д.т.н.; А.Г. Феоктистов, к.т.н.; Э.К. Вартанян
(Институт динамики систем и теории управления СО РАН, г. Иркутск,
oparin@icc.ru, agf@icc.ru, e.vartanyan@mail.ru)

Рассматривается язык описания знаний об экспериментальной Grid-системе. Описание таких знаний базируется на объектно-ориентированной модели данных.

Ключевые слова: *входной язык, объектно-ориентированная база знаний, Grid-система.*

Организация Grid-систем различного назначения является важным и практически значимым направлением научных исследований в области высокопроизводительных вычислений [1]. Сложность таких систем обусловлена их динамичностью и стохастичностью, наличием большого числа объектов различной природы и связей между ними, распределенностью этих объектов и избыточностью информационно-вычислительных ресурсов. В Grid-систему могут интегрироваться сложные предметно-ориентированные программные комплексы. Для эффективного использования Grid-системы требуются развитые средства описания, хранения и обработки знаний о ее инфраструктуре, а также о процессах планирования и вычислений.

В данной статье рассматривается язык описания объектной модели экспериментальной Grid-системы, созданной на базе вычислительных ресурсов Суперкомпьютерного центра при *Институте динамики систем и теории управления* (ИДСТУ) СО РАН.

Представление знаний. В качестве модели Grid-системы используется объектно-ориентированная модель данных, обладающая рядом важных свойств, необходимых для представления знаний о Grid-системе. В частности, данная модель

– позволяет работать со сложноструктурированными данными, отражать их природу и связи между ними;

– моделирует многомерную структуру данных, с которой приложение в дальнейшем может работать напрямую (доступ, поиск или изменение данных), что увеличивает производительность системы по отношению к реляционной модели;

– обладает гибкими средствами модификации и развития;

– поддерживает комплексирование по данным (представление объектов Grid-системы в универсальном формате, позволяющем работать с ними различным предметно-ориентированным программным комплексам).

База знаний (БЗ) Grid-системы, реализованная на основе такой модели, обеспечивает целостность и безопасность информации в процессе извлечения объектов для совместного с другими пользователями доступа к ним и облегчает работу с объектно-ориентированными приложениями.

В состав базовых элементов модели Grid-системы входят множества классов C, объектов O, полей P и типов T. Приведем ее основные характеристики:

– главное понятие в модели – объект;

– полная идентификация объекта (без привязки к конкретной БЗ) состоит из объекта и имени его типа;

– объект состоит из набора полей, задаваемых классом объекта;

– в качестве поля могут выступать значение какого-либо типа (например, строка или число), ссылка на объект определенного класса или символ неопределенности;

– каждому классу $c \in C$ соответствуют множество его объектов $o_c \in O$ и список его полей $p_c \in P$;

– ссылки на объекты являются значениями специального типа, они служат для доступа к объектам, на которые ссылаются;

– ссылки создаются одновременно с объектами, на которые они ссылаются впоследствии;

– различаются два вида ссылок на объект: простые ссылки, указывающие на один объект, и так называемые множественные ссылки, указывающие сразу на несколько объектов;

– внешнее значение ссылки представляется как имя (или имена) указываемого объекта (или объектов);

– с помощью аппарата ссылок реализуются следующие виды отношений между объектами: «один-к-одному», «многие-к-одному», «один-ко-многим» и «многие-ко-многим»;

– специальное поле в объекте отводится для хранения имени объекта;

– множественные виды отношений (все из перечисленных выше, кроме «один-к-одному») строятся на основе списков;

– списки являются объектами специального класса и служат для объединения совокупности объектов и получения возможности ссылки ко всем этим объектам как к единому целому;

– списки имеют специальное поле, в котором хранится длина списка;

– вновь вводимые классы могут наследовать свойства имеющихся классов;

– возможные изменения модели данных определяются фиксированным набором базовых операторов;

– базовые операторы можно объединять в составной оператор, причем один составной оператор может быть частью другого;

– типы и некоторые классы изначально встроены в модель данных;

– смысловая нагрузка класса объектов находит свое отражение через указание возможности участия объектов данного класса в качестве параметров определенных операторов (базовых или составных) с пояснением роли их участия (входные, выходные);

– знания предметного специалиста об условиях применения объекта того или иного класса в процессе вычислений представляются в виде набора продукций, которые являются и объектами определенного класса модели.

Введем вспомогательные элементы модели: символ неопределенности θ, который используется при создании объекта, выступая в роли значения полей до их инициализации; функцию $g{:}O\to C$, ставящую в соответствие объекту некоторый класс; функцию q: $P\to C\cup T\cup\{\theta\}$: $\forall\, p\in P$

$$q(p)=\begin{cases} c\in C, \\ t\in T, \\ \theta. \end{cases}$$

Состояние модели Grid-системы задается в виде структуры $s=<C, O, P, T>$.

Язык описания модели. Рассматриваемый язык представления знаний о Grid-системе включает синтаксические конструкции вида: `<Имя базового оператора> (<Параметр 1> [, <Параметр 2>, …, <Параметр n>])`.

Данный язык позволяет представить любые действия с моделью Grid-системы как последовательность базовых операторов. В результате выполнения базового оператора происходит переход из исходного состояния модели $s=<C, O, P, T>$ в результирующее $s'=<C', O', P', T'>$.

На выполнение базовых операторов могут быть наложены ограничения двух видов: встроенные в транслятор языка описания модели Grid-системы и дополнительные, определяемые разработчиком модели. К первому виду относятся блокировки дублирования и уничтожения классов или объектов, проверка соответствия типов и другие подобные ограничения целостности модели. Ограничения второго вида определяют специфику объектов модели и взаимосвязей между ними. Они формируются разработчиком с помощью специальной подсистемы транслятора.

Свойства модели (полнота, корректность и целостность) выявляются в процессе трансляции ее описания.

В качестве примера рассмотрим описание фрагмента вычислительной модели распределенного пакета прикладных программ по линейной алгебре. Основными объектами этой модели являются множество параметров Z, множество операций F, множество программных модулей M, реализующих операции из F, и множество вычислительных узлов Grid-системы N.

В таблице 1 приведены примеры создания классов этих объектов.

Таблица 1

Базовый оператор	Описание	Ограничения
`newClass(Prm);`	Создание класса *Prm* (класс параметров)	$Prm\notin C$
`newField(Prm, name, String);`	Создание поля *name* (полное имя параметра) в классе *Prm*. *String* – предопределенный тип	$name\notin P_{Prm}$; $String\in T$; $Prm\in C$
`newField(Prm, value, Link(Object));`	Создание поля *value* (значение параметра). Значением поля *value* является ссылка на объект класса *Object*	$value\notin P_{Prm}$; *Object*, $Prm\in C$
`newClass(Mdl);`	Создание класса *Mdl* (класс модулей)	$Mdl\notin C$
`newField(Mdl, name, String);`	Создание поля *name* (полное имя параметра) в классе *Mdl*	$name\notin P_{Mdl}$; $String\in T$; $Mdl\in C$
`newClass(Opr);`	Создание класса *Opr* (класс операций)	$Opr\notin C$
`newField(Opr, name, String);`	Создание поля *name* (полное имя операции) в классе *Opr*	$name\notin P_{Opr}$; $String\in T$; $Opr\in C$
`newField(Opr, module, Link(Mdl));`	Создание поля *module* (ссылка на программный модуль, реализующий данную операцию) в классе *Opr*	$module\notin P_{Opr}$; *Mdl*, $Opr\in C$
`newField(Opr, input, Link(Array(Prm, null)));`	Создание поля *input* (входные параметры операции) как ссылки на список объектов класса *Prm*	$input\notin P_{Opr}$; *Opr*, $Prm\in C$
`newField(Opr, output, Link(Array(Prm, null)));`	Создание поля *output* (выходные параметры операции)	$output\notin P_{Opr}$; *Opr*, $Prm\in C$
`newClass(Node);`	Создание класса *Node* (класс вычислительных узлов)	$Node\notin C$
`newField(Node, id, String);`	Создание поля *id* (уникальный идентификатор вычислительного узла) в классе *Node*	$id\notin P_{Node}$; $String\in T$; $Node\in C$
`newField(Node, modules, Link(Array(Mdl)));`	Создание поля *modules* (модули, установленные на узле) как ссылки на список объектов класса *Mdl*	$modules\notin P_{Node}$; *Mdl*, $Node\in C$

На основе созданных классов формируются объекты модели. В таблице 2 приводится описание операции масштабирования матриц при вычислении ее собственных значений и векторов.

Таблица 2

Базовый оператор	Описание	Ограничения
`newObject(n, Prm);`	Создание объекта *n* (порядок матрицы) класса *Prm*	*Prm*∈*C*; *n*∉*O*
`newObject(b, Prm);`	Создание объекта *b* (основание системы счисления в машине с плавающей запятой) класса *Prm*	*Prm*∈*C*; *b*∉*O*
`newObject(a, Prm);`	Создание объекта *a* (исходная матрица) класса *Prm*	*Prm*∈*C*; *a*∉*O*
`newObject(low, Prm);`	Создание объекта *low* (параметр операции *balance*) класса *Prm*	*Prm*∈*C*; *low*∉*O*
`newObject(hi, Prm);`	Создание объекта *hi* (параметр операции *balance*) класса *Prm*	*Prm*∈*C*; *hi*∉*O*
`newObject(d, Prm);`	Создание объекта *d* (вектор, содержащий информацию о перестановках и масштабных коэффициентах) класса *Prm*	*Prm*∈*C*; *c*∉*O*
`newObject(arr1, Array(Prm, 3));`	Создание списка из трех объектов класса *Prm* (входные параметры операции масштабирования)	*Prm*∈*C*; *arr*1∉*O*
`newObject(arr2, Array(Prm, 4));`	Создание списка из четырех объектов класса *Prm* (выходные параметры операции масштабирования)	*Prm*∈*C*; *arr*2∉*O*
`initObject(arr1[0], n);`	Инициализация элемента списка *arr*1	*arr*1, *n*∈*O*; *arr*1.*length*>0
`initObject(arr1[1], b);`	Инициализация элемента списка *arr*1	*arr*1, *b*∈*O*; *arr*1.*length*>1
`initObject(arr1[2], a);`	Инициализация элемента списка *arr*1	*arr*1, *a*∈*O*; *arr*1.*length*>2
`initObject(arr2[0], a);`	Инициализация элемента списка *arr*2	*arr*2, *a*∈*O*; *arr*2.*length*>0
`initObject(arr2[1], low);`	Инициализация элемента списка *arr*2	*arr*2, *low*∈*O*; *arr*2.*length*>1
`initObject(arr2[2], hi);`	Инициализация элемента списка *arr*2	*arr*2, *hi*∈*O*; *arr*2.*length*>2
`initObject(arr2[3], d);`	Инициализация элемента списка *arr*2	*arr*2, *d*∈*O*; *arr*2.*length*>3
`newObject(m, Mdl);`	Создание объекта *m* (модуль, реализующий операцию масштабирования) класса *Mdl*	*Mdl*∈*C*; *m*∉*O*
`newObject(bln, Opr);`	Создание объекта *bln* (операция масштабирования) класса *Opr*	*Opr*∈*C*; *bln*∉*O*
`initField(bln, name, «Balance»);`	Инициализация поля *name*. *Balance* – значение типа *String*	*bln*∈*O*; *String*∈*T*; *name*∈P_c; где *c*=*g*(*bln*), *q*(*name*)=*String*
`initField(bln, module, m);`	Инициализация поля *module*	*bln*,*m*∈*O*; *module*∈P_c, где *c*=*g*(*bln*), *q*(*module*)=*g*(*m*)
`initField(bln, input, Link(arr1));`	Инициализация поля *input*	*bln*, *arr*1∈*O*; *q*(*input*)=*g*(*arr*1); *input*∈P_c, где *c*=*g*(*bln*)
`initField(bln, output, Link(arr2));`	Инициализация поля *output*	*bln*, *arr*2∈*O*; *q*(*output*)=*g*(*arr*2); *output*∈P_c, где *c*=*g*(*bln*)

Масштабирование матрицы применяется для понижения ее нормы с целью упрощения процесса нахождения собственных значений матрицы. Схема операции масштабирования имеет следующий вид: *balance*: {*n*, *b*, *a*}→{*a*, *low*, *hi*, *d*}. Обозначения взяты из работы [2]. Как видно из примера, связи между объектами возникают при инициализации полей. В данном случае список параметров связан с операцией масштабирования через поля *input* и *output*. Описание узлов Grid-системы приведено в таблице 3.

Таблица 3

Базовый оператор	Описание	Ограничения
`newObject(r1, Node);`	Создание объекта *r*1 (узел РВС) класса *Node*	*Node*∈*C*; *r*1∉*O*
…	…	…
`newObject(rN, Node);`	Создание объекта *rN* (узел РВС) класса *Node*	*Node*∈*C*; *rN*∉*O*
`newObject(arr3, Array(Mdl,1));`	Создание списка из одного объекта класса *Mdl* (список модулей, установленных на узлах)	*Node*∈*C*; *arr*∉*O*
`initObject(arr[0], m);`	Инициализация элемента списка *arr* (в список установленных модулей заносится модуль *m*)	*arr*, *m*∈*O*; *arr*.*length*>0
`initField(r1, modules, Link(arr));`	Инициализация поля *modules* (установка модулей *arr* на узел *r*1)	*arr*, *r*1∈*O*; *q*(*modules*)= =*g*(*arr*); *modules*∈P_c, где *c*=*g*(*r*1)
…	…	…
`initField(rN, modules, Link(arr));`	Инициализация поля *modules* (установка модулей *arr* на узел *rN*)	*arr*, *rN*∈*O*; *q*(*modules*)= =*g*(*arr*); *modules*∈P_c, где *c*=*g*(*rN*)

Планирование загрузки ресурсов становится необходимым при возникновении избыточности ресурсов Grid-системы в случае, когда существует программный модуль, размещенный в двух или более вычислительных узлах (табл. 3).

База знаний. В качестве основы БЗ экспериментальной Grid-системы ИДСТУ СО РАН используется *объектно-ориентированная БД* (ООБД) NeoDatis [3], дополненная управляющей надстройкой – мультиагентной системой планирования и распределения ресурсов [4].

В целом характеристики БД NeoDatis соответствуют основным положениям стандарта ODMG (Object Database Management Group) [5] и тем самым обеспечивают возможность ее интеграции с различными предметно-ориентированными комплексами, функционирующими в Grid-системе.

Схематично процесс создания БЗ показан на рисунке. Описание предметной области, объектов и ресурсов Grid-системы на входном языке транслируется в текст описания на языке Java. Стандартный Java-компилятор, используя библиотеки Java-классов (JDK), классов взаимодействия с ООБД (драйвер) и классов модели Grid-системы,

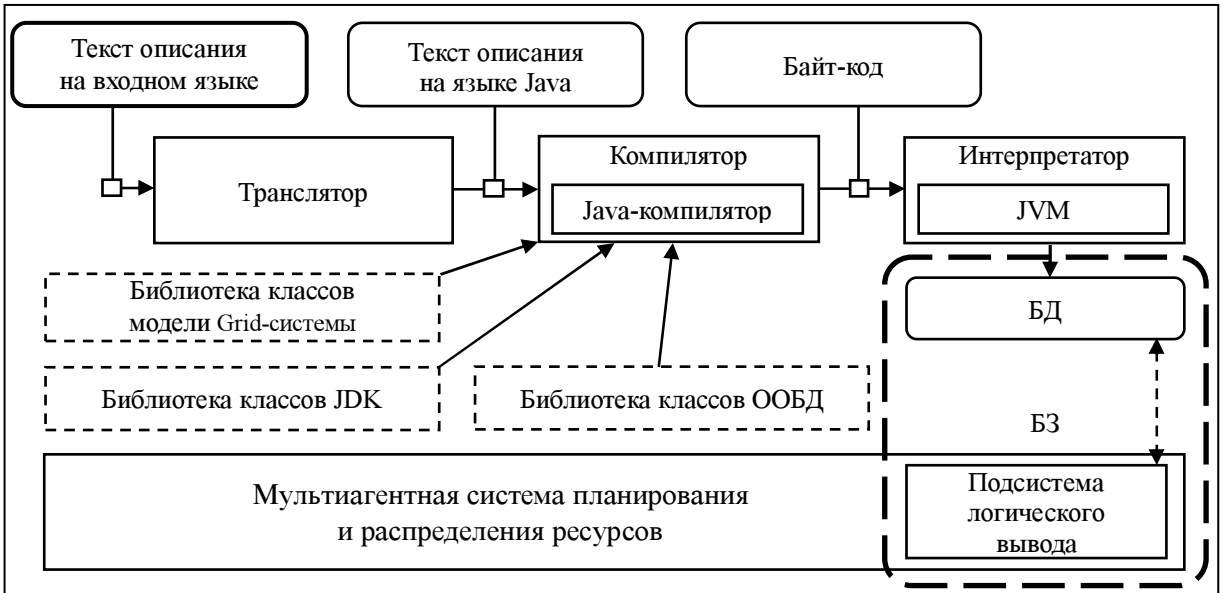


Схема создания базы знаний Grid-системы

компилирует текст на языке Java в исполняемый байт-код. При исполнении байт-кода Java-интерпретатором (Java Virtual Machine) создается БД, которая вместе с подсистемой логического вывода, интерпретирующей правила вывода данных, образует БЗ Grid-системы.

Рассмотренные в данной работе язык и средства его трансляции дают возможность конструировать предметно-ориентированные модели вычислений, включать в них требуемые ограничения и контролировать полноту, корректность и целостность атрибутов и взаимосвязей объектов разрабатываемых моделей. Такие модели обеспечивают детализированное описание инфраструктуры разнородных распределенных вычислительных сред, предоставляя возможность эффективно использовать ресурсы этих сред при решении сложных научных и прикладных задач.

Литература

1. Baker M., Buyya R., Laforenza D. Grids and Grid Technologies for Wide-Area Distributed Computing // Software: Practice and Experience. 2002. Vol. 32. No. 15, pp. 1437–1466.

2. Уилкинсон, Райнш. Справочник алгоритмов на языке АЛГОЛ. Линейная алгебра. М.: Машиностроение, 1976. 389 с.

3. NeoDatis. URL: http://neodatis.wikidot.com (дата обращения: 17.10.2011).

4. Джордан Д. Обработка объектных баз данных в C++. Программирование по стандарту ODMG. М.: Издат. дом «Вильямс», 2001. 384 с.

5. Децентрализованное управление потоками заданий в интегрированной кластерной системе / И.В. Бычков [и др.] // Вестн. НГУ. Сер. Информационные технологии. 2011. Т. 9. Вып. 2. С. 42–54.

УДК 004.75

ИНФОРМАЦИОННАЯ СИСТЕМА ГридННС

(Работа выполнена при финансовой поддержке Минобрнауки РФ, контракт № 07.514.11.4022, и гранта РФФИ № 11-07-00434-а)

А.П. Крюков, к.ф.-м.н.; Л.В. Шамардин, к.ф.-м.н.; Д.О. Патрикеев
(Научно-исследовательский институт им. Д.В. Скобельцына МГУ им. М.В. Ломоносова (НИИЯФ МГУ), kryukov@theory.sinp.msu.ru, shamardin@theory.sinp.msu.ru, patrikeev@theory.sinp.msu.ru)

Описаны архитектура информационной системы ГридННС, построенная на основе RESTful-веб-сервисов, структура публикуемой информации и используемая модель безопасности. Одним из главных отличий ГридННС от других грид-инфраструктур является использование архитектурного стиля REST для реализации грид-сервисов.

Ключевые слова: *высокопроизводительные вычисления, распределенные вычисления, грид, ГридННС, REST, RESTful-веб-сервисы, информационная система, система учета.*

Концепция грида была предложена в 90-х годах 20 века К. Кессельманом и Я. Фостером [1]. За прошедшее время гриды прошли значительный путь развития. На основе успешного опыта ис-

пользования Globus Toolkit версии 2 (GT2) сформулирована открытая архитектура грид-сервисов (Open Grid Services Architecture, OGSA, *http://www.ogf.org/documents/GFD.80.pdf*), которая в дальнейшем была заменена на WSRF (*http://docs.oasis-open.org/wsrf/wsrf-primer-1.2-primer-cd-02.pdf*). В рамках подхода WSRF объединяются веб-сервис и ресурс, доступ к которому он обеспечивает. Канонической реализацией WSRF является Globus Toolkit версии 4 (GT4).

Следуя спецификациям WSRF, можно создавать универсальные сервисы, использующиеся практически для любых гридов. Однако стандарт WSRF оказался исключительно сложным и, как следствие, трудным в реализации. Даже каноническая реализация в виде GT4 не является полной и содержит ряд ошибок. Все это ставит под угрозу достижение основной цели – создание универсального, динамически изменяющегося грида.

В 2000 г. Р. Филдинг предложил новый, простой в использовании и гибкий подход к созданию веб-сервисов – архитектурный стиль REST [2]. Важным достоинством RESTful-сервисов по сравнению с подходом, основанным на WSRF, является несложная семантика запросов, соответствующая интуитивно ясным методам HTTP-протокола. Кроме того, RESTful-сервисы используют протокол HTTP по его прямому назначению – как протокол запросов, а не просто как транспортный протокол, в отличие от WSRF, где процессы сериализации и десериализации создают дополнительные проблемы, в том числе и с совместимостью реализаций.

В процессе работы над проектом *грид-инфраструктуры Национальной нанотехнологической сети* (ГридННС) (*http://ngrid.ru/ngrid, 2008*) авторы использовали архитектурный стиль REST для реализации грид-сервисов в рамках концепции OGSA. В частности, на этой основе был реализован ключевой сервис ГридННС, управляющий выполнением заданий и названный Pilot [3, 4].

В настоящей работе описывается информационная система ГридННС, построенная на базе RESTful-веб-сервисов.

Структура ГридННС

ГридННС предоставляет возможность унифицированного безопасного удаленного доступа к суперкомпьютерным ресурсам ННС.

Структуру ее можно представить в виде трех слоев (рис. 1): пользовательские интерфейсы, грид-шлюзы к ресурсам и общие инфраструктурные сервисы.

Слой пользовательских интерфейсов предназначен для запуска пользователями своих заданий в ГридННС.

Слой общесистемных сервисов отвечает за функционирование и управление ГридННС. Основными общесистемными сервисами являются *информационная система* (ИС), система учета, сервис управления и распределения задач, система регистрации сервисов и ресурсов, служба выдачи и управления цифровыми сертификатами, сервис проверки работоспособности ресурсов.

Слой грид-шлюзов обеспечивает доступ к вычислительным ресурсам, подключенным к ГридННС. Грид-шлюз – это комплекс сервисов, обеспечивающий всю функциональность, необходимую для использования кластера или суперкомпьютера из ГридННС.

Важный принцип, положенный в основу проектирования ГридННС, – отказ от установки дополнительного ПО на вычислительном ресурсе. Все промежуточное ПО устанавливается только на специальном сервере – грид-шлюзе. При таком подходе грид-шлюз взаимодействует с системой пакетной обработки вычислительного ресурса как специфический пользователь.

Одним из компонентов грид-шлюзов является сервис ИС. Другой компонент ИС, агрегатор (hub), является частью общесистемных сервисов. Опи-

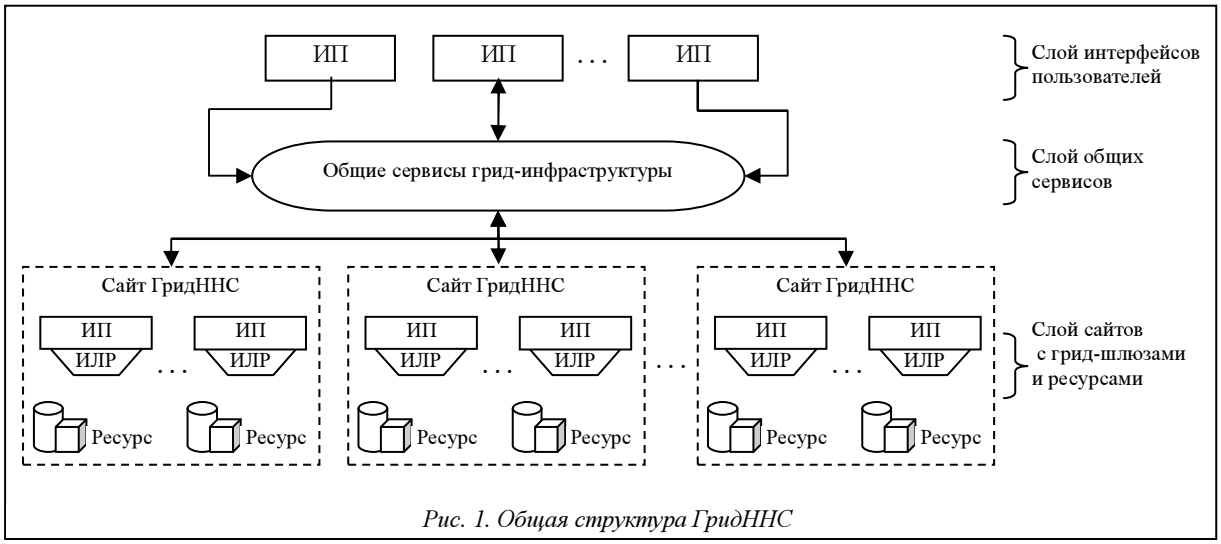


Рис. 1. Общая структура ГридННС

шем подробнее устройство и особенности функционирования ИС ГридННС.

ИС ГридННС

ИС ГридННС первоначально была построена на базе WS-MDS из инструментального набора Globus Toolkit 4. Однако из-за сложности использования и расширения возможностей WSRF-сервисов возникла необходимость перехода на более удобные в использовании RESTful-веб-сервисы. Новая реализация ИС для грида стала еще более актуальной после того, как появилась информация о намерении разработчиков Globus Toolkit отказаться от использования WSRF-сервисов в последней версии Globus Toolkit 5, а также из-за отсутствия в его составе собственной ИС и инструментария для ее создания.

Архитектура ИС ГридННС изображена на рисунке 2. Основная задача ИС в том, чтобы обеспечить пользователей и сервисы ГридННС информацией о текущем состоянии ресурсов. Например, сервису управления выполнением заданий Pilot эта информация позволяет выбрать подходящий ресурс для выполнения задач с учетом требований с их стороны и планирования распределения нагрузки. В качестве источников информации о ресурсе выступают файл с его статическим описанием, а также провайдер динамической информации о нем. Статическая информация включает, например, название ресурса, его местоположение. Первоисточник динамической информации – *локальный менеджер ресурсов* (ЛМР), которым является *система управления пакетной обработкой* (СУПО) вычислительного ресурса.

Особое внимание при построении ИС уделяется схеме данных. В качестве прототипа схемы

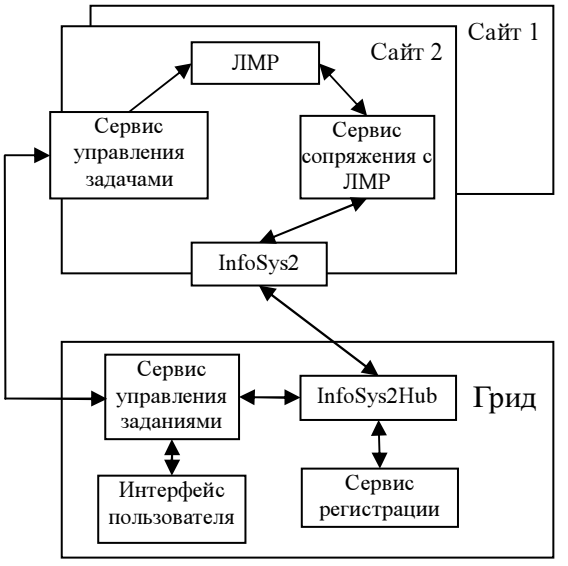


Рис. 2. Архитектура информационной системы ГридННС

данных использовалась GLUE Scheme 2.0 (*http://forge.ogf.org/sf/projects/glue-wg, 2007*), а в качестве формата для обмена данными – JSON [5]. Общая структура публикуемой информации по каждому вычислительному ресурсу имеет следующий вид:

```
{
...
   "Site": {
   ...
      "Cluster": [
      ...
         "SubCluster": [
          ...
          {
           ...
           "Queue": [...]
           "Host": [...]
           "Software": [...]
          }
         ]
      ]
   }
...
}
```

Таким образом, каждый ресурс (сайт) представляет собой список кластеров, которые, в свою очередь, содержат подкластеры. Современные суперкомпьютеры довольно часто являются неоднородными. Например, суперкомпьютер «Ломоносов», занимающий 18-е место в Top500 самых мощных суперкомпьютеров мира, имеет ряд рабочих узлов с расширенной оперативной памятью, другая часть использует графические процессоры. Разбиение кластера на подкластеры позволяет учесть неоднородную структуру вычислительного ресурса. Структура подкластера предполагается однородной.

Статическая часть информации о ресурсе содержит его общее описание (наименование, местоположение, административные контакты и т.д.). Приведем пример статической информации:

```
{
    "__metadata": {
        "infosys2_site_version": "0.5.0"
    },
    "CreationTime": "2011-11-24T09:06:40Z",
    "Validity": 3600,
    "Site": {
        "Cluster": [ ... ],
        "Description": "Resource center",
        "Info": {
            "ServiceEndpoint": {
                "GRAM": [
                    "https://rc.ru:8443/wsrf/services/
ManagedJobFactoryService"
                ],
                "GridFTP": [
                    "gsiftp://rc.ru:2811"
                ],
                "IS2": "https://rc.ru:8443/
services/LIS2",
                "RFT": [
                    "https://rc.ru:8443/wsrf/services/
ReliableFileTransferFactoryService"
                ]
            }
        },
        "Latitude": 55.6,
        "Location": "Moscow, Russia",
        "Longitude": 37.5,
        "Name": "xxx",
        "SecurityContact": "mailto: security@rc.ru",
        "SysAdminContact": "mailto: sysadmin@rc.ru",
        "UniqueID": "xxx.ru",
        "UserSupportContact": "mailto: support@rc.ru",
        "Web": "http://www.xxx.ru",
    }
}
```

Сведения о кластерах, зарегистрированных на одном ресурсном центре, помещаются в массив «Cluster», содержание которого в примере условно обозначено «[...]». Секция «Cluster» включает следующую информацию:

```
{
    "SubCluster": [
         {
             "Name": "rc.ru/subcluster0",
             "PhysicalCPUs": 16,
             "LogicalCPUs": 16,
             "Queue": [ ... ],
             "Host": [ ... ],
             "Software": [ ... ],
         },
         ...
    ] ...
}
```

Структура рабочих узлов описывается в секции «Host»:

```
   "Host": [
       {
           "MainMemory": {
               "RAMSize": 1024,
               "VirtualSize": 8192
           },
           "Processor": {
               "ClockSpeed": 2667,
               "Model": "E5430",
               "Vendor": "Intel Xeon",
               "InstructionSet": "x86"
           },
           "OperatingSystem": {
               "Release": "5.3",
               "Version": "Final",
               "Name": "CentOS"
           },
           "UniqueID": "subcluster0.rc.ru/
host",
           "Architecture": {
               "SMPSize": 1,
               "PlatformType": "i686"
           }
       }
   ] ...
```

Следует еще раз подчеркнуть предположение о том, что подкластер имеет однородную структуру. Вся неоднородность учтена на уровне кластера, который может включать разные подкластеры.

Информация о доступном пользователям ПО публикуется в секции «Software»:

```
    "Software": [
        {
            "LocalID": "hpmpi-2.02.05.01-20070708r",
            "Version": "2.02.05.01-20070708r",
            "Name": "hpmpi",
            "InstalledRoot": "/opt/hpmpi"
        },
        {
            "ModuleName": "lmp",
            "LocalID": "lammps-25Sep11",
            "EnvironmentSetup": [
                {
                    "softenv": "+lammps-25sep11"
                }
            ],
            "ACL": {
                "Rule": [
                    "VOMS:/sysadmin",
                    "VOMS:/gridnnn",
                    "VOMS:/education",
                    "VOMS:/abinit"
                ]
            },
            "Version": "25Sep11",
            "Name": "lammps",
            "InstalledRoot": "/shared/lammps"
        }
    ], ...
```

В секции, описывающей ПО, важную роль играют права доступа («ACL»), с помощью которых системный администратор может сообщить о существовании ограничений на использование тех или иных пакетов прикладных программ членами конкретных *виртуальных организаций* (ВО). Таким образом, публикуется локальная политика использования прикладного ПО в зависимости от принадлежности к ВО. В свою очередь, сервис управления выполнением заданий Pilot при выборе ресурса, на котором может быть выполнена задача, руководствуется этой информацией. Если данный ресурс не публикует сведения о поддержке какой-либо ВО, то он исключается при выборе места выполнения для задач пользователей этой ВО.

Важной частью схемы является секция «EnvironmentSetup», где указываются метки программного окружения «softenv» или другие расширения, по наличию которых ПО грид-шлюза осуществляет специальную настройку среды выполнения для обеспечения доступа к соответствующему ПО некоторым стандартным образом. Например, этой настройкой может быть добавление необходимых путей в переменные PATH и LD_LIBRARY_PATH. Данная необходимость связана с тем, что прикладное ПО на ресурсах может быть установлено различными способами. Простейший пример – использование разных корневых директорий для прикладных пакетов. Так как в момент запуска задания пользователь не знает, на каком ресурсе будет выполнена его задача, необходим механизм ее настройки в момент запуска на конкретном ресурсе. Пользователь указывает названия пакетов, требования к их версиям. Сервис Pilot извлекает из ИС список окружений, необходимых для данных пакетов на выбранном ресурсе, и дополняет описание задачи соответствующими расширениями. Таким образом, пользователю не требуются знания особенностей установки ПО на ресурсах грида, что существенно упрощает запуск задач в такой неоднородной среде.

Не менее важным атрибутом ресурса, публикуемым в ИС, является информация о свойствах очереди, в частности, об обслуживаемых ВО. Секция «Queue» содержит динамическую информацию о состоянии очередей СУПО каждого подкластера вычислительного ресурса. Публикуемая динамическая информация получается от СУПО.

Структура секции следующая:

```
"Queue": [
    {
         "CEInfo": "example.com/batch-A",
         "Feature": [ "mpi", "single" ],
         "ACL": {
             "Rule": [
                 "VOMS:/nnn-vo-0",
                 "VOMS:/nnn-vo-1",
                 "VOMS:/nnn-vo-2/group1",
                 "VOMS:/nnn-vo-3/Role=VO-Admin"
             ]
         }
         "MaxWallTime": 6000,
         "MaxTotalJobs": 100,
         "MaxRunningJobs": 50,
         "RunningJobs": 30,
    }
] ...
```

В приведенном примере динамической информацией является количество запущенных заданий («RunningJobs»).

Следует обратить внимание на то, что данная секция, как и секция «Software», имеет подсекцию «ACL» и позволяет указать права доступа к очереди членам ВО. Возможна и более детальная спецификация доступа на основе ролей и групп пользователей в данной ВО. Параметры очередей СУПО устанавливают политику администрации вычислительного ресурса по отношению к потребленным компьютерным ресурсам, таким как время счета и максимальное количество доступных вычислительных ядер. Таким образом, данная секция позволяет информировать пользователей об ограничениях на возможное количество потребляемых ресурсов.

Секции «Software» и «Queue» совместно обеспечивают эффективный и гибкий механизм управления доступом к вычислительным ресурсам.

Информация с каждого ресурса агрегируется специальным RESTful-сервисом InfoSys2Hub, собирающим информацию с сайтов, опубликованных в сервисе регистрации грид-сервисов ГридННС, с учетом их режима работы («работает» или «тестируется»). Такая структура позволяет потребителям информационного сервиса иметь дело с единственной точкой входа в инфраструктуру. Регистрация нового ресурса автоматически приводит к появлению информации о нем в ИС, изменения состояния сайтов также отражаются в общей ИС.

Безопасность ИС построена на использовании цифровых сертификатов стандарта X.509 инфраструктуры публичных ключей (PKI). Доступ к сервису предоставляется конечному набору потребителей в соответствии с конфигурацией сервиса.

В заключение необходимо отметить, что в процессе разработки ГридННС был решен ряд принципиальных вопросов создания грид-инфраструктур на основе RESTful-грид-сервисов.

В частности, реализована ИС как RESTful-сервис, обеспечивающий публикацию информации о состоянии ресурсов, ее агрегацию со всех доступных для пользователей ресурсов, предоставление ее потребителям.

В качестве формата обмена информацией был использован JSON. Его гибкость позволила реализовать подмножество публикуемых параметров, которые описаны в GLUE Scheme 2 – стандарте для построения ИС в грид-инфраструктурах.

Использование RESTful-сервисов дало возможность применять HTTP-протокол не только как транспортный, но и по его прямому назначению – как протокол запросов с ясной семантикой. Это обусловило значительное упрощение сериализации и десериализации информации, что повысило надежность системы.

В ближайшее время планируется внедрение данной версии ИС в инфраструктуру ГридННС в качестве основной.

Литература

1. Foster I. and Kesselman C. The Globus Project: A Status Report // In Proc. Heterogeneous Computing Workshop, IEEE Press, 1998, pp. 4–18.

2. Fielding R.T. Architectural styles and the design of network-based software architectures // Doctoral dissertation, University of California, Irvine, 2000.

3. Реализация программного интерфейса грид-сервиса Pilot на основе архитектурного стиля REST / Демичев А. [и др.] // Вычислительные методы и программирование. 2010. Т. 11. С. 62–65.

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

5. K.Zyp: A JSON Media Type for Describing the Structure and Meaning of JSON Documents // Technical report, IETF Network Workong Group, draft-zyp-json-schema-02, March 2010.

УДК 519.688

ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА АВТОМАТИЗАЦИИ ПАРАЛЛЕЛЬНОГО РЕШЕНИЯ БУЛЕВЫХ УРАВНЕНИЙ НА МНОГОЯДЕРНЫХ ПРОЦЕССОРАХ

Г.А. Опарин, д.т.н.; В.Г. Богданова, к.т.н.

(Институт динамики систем и теории управления Сибирского отделения РАН, г. Иркутск, oparin@icc.ru, bvg@icc.ru)

Рассматриваются принципы организации, основные функции и структурные компоненты инструментального комплекса для автоматизации параллельного решения булевых уравнений на многоядерных процессорах, в частности, технология автоматизации булева моделирования.

Ключевые слова: *параллельные вычисления, булево моделирование, решение булевых уравнений.*

Современный этап развития вычислительной техники характеризуется массовым распространением параллельных компьютеров. Расширяется круг предметных областей, в которых для реше-

ния задач применяются параллельные технологии. С появлением многоядерных и многопоточных процессоров возникла возможность более глубокого распараллеливания вычислений. Для реализации двухуровневого параллелизма на кластере, имеющем в вычислительных узлах многоядерные процессоры, на верхнем уровне можно организовать взаимодействие между подзадачами путем передачи сообщений, а на нижнем использовать технологии работы с общей памятью, применив к подзадаче модель параллелизма по управлению. В настоящее время повышается интерес к использованию высокопроизводительных кластеров для решения задачи удовлетворения ограничений.

К важному классу ограничений относятся булевы ограничения, представляемые в виде систем булевых уравнений. Постановка задачи формулируется следующим образом: для нахождения решения системы булевых уравнений, которую без потери общности можно представить в виде одного булева уравнения $f(x_1, \ldots, x_n)=0$, где $x_i \in B_2$ для всех $i = \overline{1,n}$, $B_2=\{0, 1\}$, $f : B_2^n \to B_2$ – заданная функция своих аргументов, требуется найти одно или все решения уравнения либо сделать вывод, что решений не существует. Такие задачи, как правило, имеют тяжелые комбинаторные характеристики с высокой оценкой сложности, что заставляет вести поиск методов, наиболее эффективно действующих для отдельных (интересных с практической точки зрения) классов булевых систем, и одновременно повышать производительность решения систем булевых уравнений путем использования многопроцессорной техники.

Для параллельного решения задач булевой выполнимости применяются различные эвристические методы расщепления булевой функции, а также разработаны параллельные SAT-решатели дискретных задач. В качестве входного языка таких программ используется фрагмент международного формата DIMACS представления булевых ограничений в виде конъюнктивной нормальной формы пропозициональной логики. Следует отметить, что средства автоматизации построения булевых моделей по содержательной (ориентированной на конечного пользователя) постановке задачи в этих программах отсутствуют. В большинстве существующих программ-решателей, использующих метод расщепления для крупноблочного распараллеливания булевых моделей, выбор тактики расщепления зависит от задачи и возлагается на пользователя.

В статье рассматривается *инструментальный комплекс* (ИК) РЕБУС, применяющий технологию синтеза булевых ограничений по содержательному описанию дискретной задачи, технологию распараллеливания решения системы булевых уравнений как на уровне узлов кластера, так и на уровне процессорных ядер, содержащей методы и средства автоматизации представления, накопления, модификации и обработки знаний (в частности, алгоритмы полного поиска) для решения задач в булевых ограничениях, и ориентированной на использование в фундаментальных и прикладных исследованиях, где естественным образом возникают дискретные модели в виде систем булевых уравнений.

Известно, что с точки зрения человеческого восприятия булевы системы имеют сложную специфическую природу. В связи с этим актуальной, практически значимой и весьма нетривиальной проблемой остается построение средств, позволяющих человеку моделировать задачи на языке булевых ограничений. К сожалению, работ, содержащих описание программных средств автоматизации построения булевой модели, относительно немного, в частности [1]. До сих пор остается открытым вопрос и о параллельном решении систем булевых уравнений с левой частью общего вида.

Технология решения комбинаторной задачи в ИК РЕБУС соответственно включает четыре этапа обработки информации: моделирование, декомпозиция булевой модели, параллельные вычисления и обработка результатов. Возможны три варианта проведения вычислений: ПЭВМ–кластер, только ПЭВМ, только кластер. В варианте ПЭВМ–кластер первый, второй и четвертый этапы выполняются на рабочей станции, на третьем этапе строится план решения задачи, генерируется задание для проведения параллельных вычислений, которое передается в *систему управления заданиями* (СУПЗ) на кластер. Вычисления проводятся в узлах кластера, после чего результаты передаются на рабочую станцию. Во втором и третьем вариантах все этапы выполняются соответственно на ПЭВМ или кластере.

Базовая технология автоматического синтеза булевых ограничений по содержательному описанию дискретной задачи, применяемая в ИК РЕБУС, была разработана в процессе исследования построения булевых моделей для ряда классических и практических дискретных задач. Анализ этого процесса позволил выделить характерные особенности, классифицирующие дискретные задачи по ряду признаков.

Первый признак – зависимость булева вектора X от дискретного времени $t=0, 1, 2, \ldots, k$. С этой точки зрения различаются статические (или одношаговые) задачи, вектор состояния которых не зависит от времени t, и динамические (или многошаговые) задачи, вектор состояния которых зависит от времени t. В первом случае булевы ограничения (в векторной форме) имеют вид $F(X)=0$, во втором рассматривается уравнение вида $F(X_0, X_1, \ldots, X_k)=0$, где $X_i=X(t)$ при $t=i$.

Второй признак – структуризация множества X булевых переменных задачи. Здесь рассматриваются варианты линейного упорядочения множест-

ва X (в данном случае X – вектор независимых переменных) и вариант упорядочения множества X в виде многомерного массива булевых переменных.

Приведенная классификация является в определенной степени условной, и существуют дискретные задачи, занимающие в ней промежуточное положение. В общем случае размерность динамических дискретных задач (при прочих равных условиях) существенно выше размерности статических задач, так как для динамической (многошаговой) задачи необходимо определить значение вектора состояния для каждого момента времени t=0, 1, 2, …, k.

Построение компьютерной модели включает следующие этапы:

1) задание множества булевых переменных $X=\{x_1, x_2, \ldots, x_n\}$;

2) задание системы булевых ограничений (множество $C=\{C_1, C_2, \ldots, C_m\}$);

3) представление системы ограничений в одном из форматов входных данных программ-решателей ИК РЕБУС.

Для автоматизации второго и третьего этапов была разработана специальная методика моделирования [2]. Методика булева моделирования предоставляет два различных способа построения компьютерной модели: автоматический синтез булевой модели в виде текста программ на языке программирования (Фортран, C++) и синтез системы булевых ограничений в международном формате DIMACS.

Для автоматического синтеза применяется генератор процедурного описания булевой модели – подсистема, разработанная с целью организации параллелизма на нижнем уровне и предназначенная для автоматического построения вычислителя булевой функции, использующегося решателем в процессе поиска в пространстве возможных решений [3]. Эта подсистема позволяет из формата представления булевой функции на языке математического редактора формул получить параллельный алгоритм ее вычисления на заданном наборе переменных.

При втором способе построения компьютерной модели применяется конструктор булевой модели – подсистема, обеспечивающая генерацию булевых ограничений по декларативным описаниям дискретных задач. Декларативное описание задачи задается при помощи специального *языка содержательных формулировок* (ЯСФОР) [2]. Встроенный в ИК РЕБУС конвертор осуществляет синтаксический анализ программ на данном языке и генерирует булевы уравнения во внутреннем формате, предусмотренном языком. Далее следует этап построения по полученной системе уравнений набора ограничений в DIMACS-формате.

Решение булева уравнения $f(x_1, \ldots, x_n)=0$ на многоядерных процессорах с общей памятью дает возможность применять двухуровневое распараллеливание вычислительной нагрузки. На верхнем уровне, уровне узлов кластера, используется параллелизм по данным. На нижнем – уровне процессорных ядер – параллелизм по задачам.

Для распараллеливания булевой модели наиболее предпочтительным с точки зрения решения данного уравнения является метод расщепления, позволяющий выполнить декомпозицию этого уравнения на произвольное число независимых уравнений, объединение решений которых даст решение исходного уравнения. Каждое независимое уравнение может решаться в своем вычислительном узле. Анализатор булевой модели выбирает тактику расщепления и генерирует схему расщепления, используемую для последующей декомпозиции этой модели. Для проведения вычислений в узлах кластера используются решатели – вычислительные модули ИК РЕБУС. Три алгоритма решения булевых уравнений, лежащие в основе работы этих модулей, показали сравнительно высокую эффективность на ряде классических и практических дискретных задач. Специфика этих методов заключается в ориентации на решение больших разреженных систем булевых уравнений, а в качестве преимущества первых двух методов следует отметить возможность представления левой части булева уравнения в общем виде в отличие от существующих SAT-решателей задач удовлетворения булевых ограничений, требующих соответствующего представления в конъюнктивной нормальной форме. Третий алгоритм предполагает знание структуры булевой формулы приведенного уравнения, которая должна быть представлена в дизъюнктивной нормальной форме. Этот алгоритм использует булево распространение ограничений и интеллектуальный возврат, а также специальную внутреннюю структуру данных для представления булевой функции, позволяющую особенно эффективно решать задачи, когда булева формула содержит только бинарные и p-арные конъюнкты с положительным или отрицательным вхождением булевых переменных соответственно.

Для распараллеливания на нижнем уровне разработана параллельная версия этого алгоритма, использующая стандарт OpenMP для работы с общей памятью.

Параллельный исполнитель – подсистема вычислительного эксперимента ИК РЕБУС – отвечает за проведение вычислений на кластере. ИК РЕБУС допускает две схемы проведения параллельных вычислений: 1) генерация параллельной программы, использующей библиотеку MPI с последующим сохранением, компиляцией и выполнением этой программы на кластере; 2) интеграция с ИК DISCOMP [4]. Обе схемы требуют средства интерактивного взаимодействия с кластером. В первом случае это взаимодействие обеспечивает ИК РЕБУС, использующий штатные средства

СУПЗ кластера и утилиты удаленного доступа. Во втором случае взаимодействие с кластером осуществляется средствами доступа ИК РЕБУС к программному интерфейсу XML-RPC ИК DISCOMP. При этом перед проведением вычислительного эксперимента требуется обеспечить формирование в рабочей области пользователя на кластере области расчетных данных, содержащей спецификации объектов предметной области и входные данные.

Затем средствами интерфейса XML-RPC производится запуск вычислительного процесса. Мониторинг выполнения этого процесса, выявление его завершения и считывание результатов вычислений из области расчетных данных кластера в ИК РЕБУС также отслеживаются при помощи средств доступа к XML-RPC-сервису. В этом случае на верхнем уровне абстракции вычислительная модель [5] ИК РЕБУС может быть представлена в виде двудольного графа $\langle P, O, E\rangle$, где доля P содержит множество параметров, доля O – множество операций ИК РЕБУС, а множество ребер E – направленные связи между параметрами и операциями. Списки параметров, операций и модулей хранятся в БЗ ИК РЕБУС. В соответствии со спецификой решаемой задачи множество T допустимых типов параметров наряду с традиционными типами языков программирования дополнено специальным типом *par* – «параллельный список». Интерпретация операции $f{:}x{\to}y$, где x, y – параметры типа *par*, выполняется следующим образом: 1) в БЗ осуществляется поиск всех вычислительных ресурсов (модуль + узел), реализующих операцию f; 2) организуется параллельное применение i-го параллельного ресурса к i-му элементу списка x; 3) результат применения присваивается i-му элементу.

В инструментальной среде ИК РЕБУС можно выделить следующие основные системы: моделирования, проведения вычислительного эксперимента и работы с БЗ. Система моделирования включает инструментальные средства автоматизации построения булевых ограничений и расщепления булевой модели. Система вычислительного эксперимента содержит инструментарий для построения плана решения задачи, организации процесса решения (последовательного или параллельного проведения вычислений), визуализации результатов. Информация, необходимая для работы этих систем, содержится в БЗ ИК РЕБУС. Система работы с БЗ включает три компонента: справочник (предоставление справочной информации по БЗ в соответствии с правами пользователя в результате выполнения запроса или составления отчета); администратор (администрирование пользователей, работа с нормативно-справочной информацией и мониторинг системы); пользователь (просмотр разрешенных таблиц БЗ и таблиц собственной БД, сохранение результатов).

Рассмотрим функциональные возможности компонентов ИК РЕБУС, обеспечивающих организацию параллельных вычислений, а именно: подсистемы моделирования, включающей модули конвертирования, анализа и расщепления модели; подсистемы параллельного исполнителя, организующего интерактивное взаимодействие с кластером; подсистемы обработки результатов решения, содержащей модули постпроцессора, слияния и визуализации результатов работы (см. рис.). Эти программные средства обеспечивают: анализ структуры булевой модели; выбор тактики расщепления; формирование схемы расщепления; генерацию спецификаций параметров и операций предметной области и схем решения задачи; генерацию паспорта задания для кластера; получение остаточной функции в узле в результате применения схемы расщепления к исходной функции; упрощение остаточной функции; параллельное решение полученных в результате расщепления подсистем булевых уравнений решателями вычислительного ядра ИК РЕБУС или внешними SAT-решателями, установленными в узлах кластера; слияние и визуализацию результатов решения; интерактивное взаимодействие с кластером.

Интерактивное взаимодействие ИК РЕБУС с кластером (см. рис.) в случае и генерации параллельной программы, и использования ИК DISCOMP требует, во-первых, предварительного сохранения в рабочей области пользователя либо текста параллельной программы, либо спецификаций параметров, операций и схемы решения задачи в области расчетных данных для первого и второго случаев соответственно; во-вторых, формирования паспорта задания, содержащего перечень исполняемых программ, названия файлов с исходными данными и результатами работы, инструкции по выполнению программ, требования к ресурсам. Средства, поддерживающие эти функциональные требования, реализованы в исполнителе, являющемся компонентом системы проведения вычислительного эксперимента ИК РЕБУС.

Представленная в данной статье инструментальная среда разработана на основе принципов создания многоплатформенных приложений, обеспечивающих переносимость на уровне исходного кода и использование приложений, разработанных для ОС Windows и Linux, в качестве модулей. Технология булева моделирования применялась для решения ряда дискретных задач, например, для поиска гамильтонова цикла, нахождения покрытия конечного множества системой его подмножеств, составления плана ремонта электропоездов при равномерной загрузке локомотивного депо, составления оперативного плана отправления локомотивных бригад в условиях жесткого расписания грузовых поездов, генерации туннельных маршрутов (подзадача задачи о равномерной загрузке каналов связи СПД), построе-

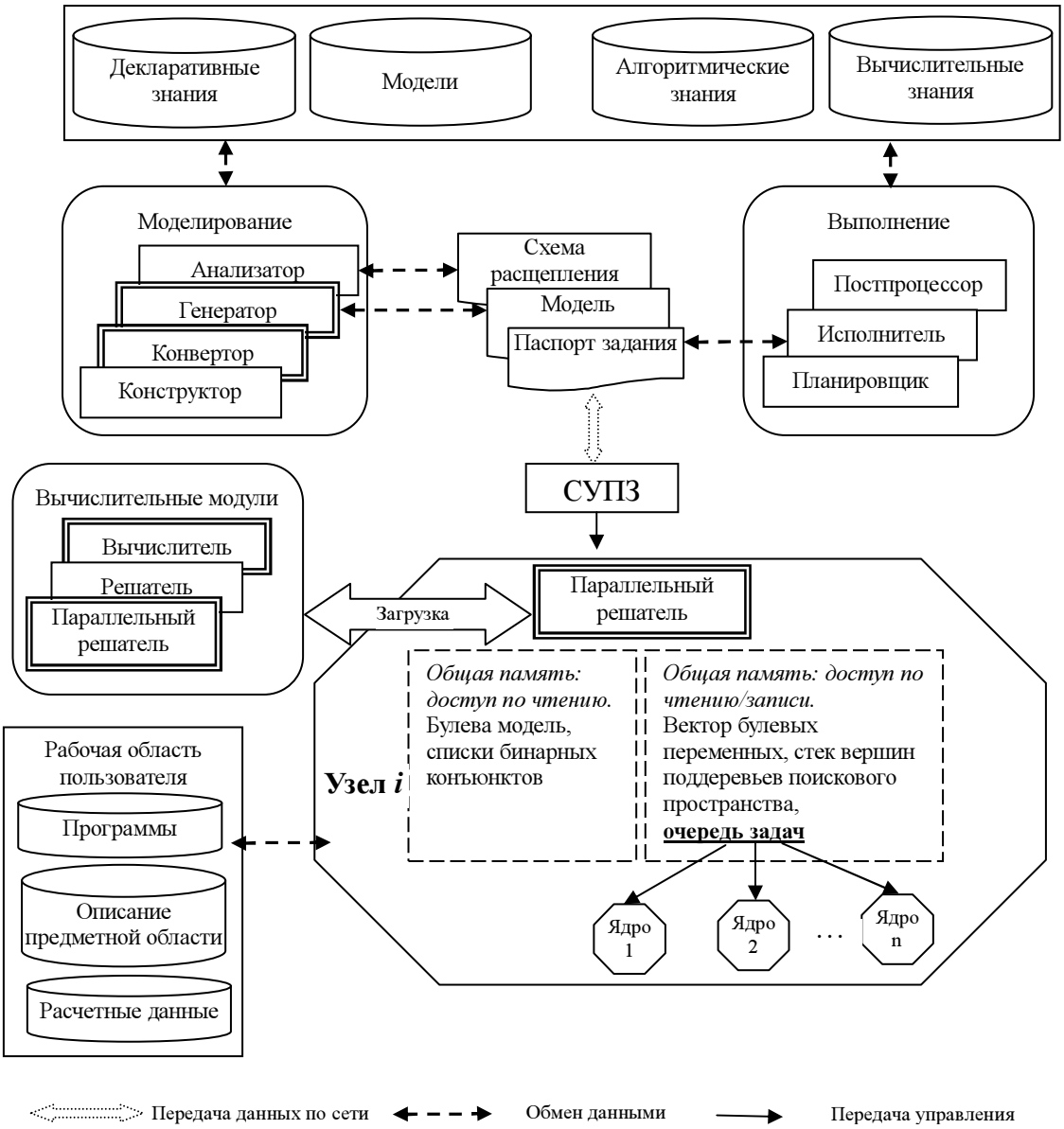


Схема проведения параллельных вычислений в ИК РЕБУС

ния параллельного плана требуемой длины в распределенных вычислительных системах. Вычислительные эксперименты проводились в ИДСТУ СО РАН (г. Иркутск) на кластере выделенных рабочих станций с процессором Intel Core 2 Duo, на кластере Blackford Multicore, имеющем в каждом вычислительном узле два четырехъядерных процессора Intel Xeon 5345, и сервере NVidia Tesla, имеющем два шестиядерных процессора Intel Xeon X5670, под управлением соответственно ОС Windows и Linux и показали эффективность рассматриваемого подхода.

Дальнейшее направление исследований связано с разработкой решателя для графического ускорителя и проведением экспериментов на сервере NVidia Tesla, имеющем 4 GPU NVidia Tesla C1060 с общим числом ядер 960.

Литература

1. Отпущенников И.В., Семенов А.А. Технология трансляции комбинаторных проблем в булевы уравнения // Прикладная дискретная математика. 2011. № 1. С. 96–115.

2. Опарин Г.А., Богданова В.Г. РЕБУС – интеллектуальный решатель комбинаторных задач в булевых ограничениях // Вестн. НГУ. Сер. Информационные технологии. 2008. Т. 6. Вып. 1. С. 60–68.

3. Богданова В.Г., Попов Е.В. Автоматизация параллельного вычисления булевой функции общего вида на многоядерных процессорах // Винеровские чтения: матер. IV Всерос. конф. Иркутск: Изд-во ИрГТУ, 2011. Т. 3. С. 5–10.

4. Интеллектуальные технологии и инструментальные средства создания вычислительной инфраструктуры в сети Интернет / С.Н. Васильев [и др.] // Вычислительные технологии. 2006. Т. 11. С. 34–44 (Спец. вып.).

5. Решение булевых уравнений большой размерности в распределенной вычислительной среде / Г.А. Опарин [и др.] // Распределенные вычисления и Грид-технологии в науке и образовании: тр. Междунар. конф. Дубна: ОИЯИ, 2004.

УДК 004

ПРОГРАММНЫЙ КОМПЛЕКС ДЛЯ ИССЛЕДОВАНИЯ УСТОЙЧИВОСТИ НЕЛИНЕЙНЫХ ДИНАМИЧЕСКИХ СИСТЕМ

А.Ю. Кузнецов
(Тверской государственный университет, ferus.tigris@gmail.com)

Представлен комплекс программ для исследования устойчивости автономных нелинейных динамических систем без использования функций Ляпунова. Описаны его структура и основные особенности реализации на ПЭВМ. Приведены структуры алгоритмов основных вычислительных модулей комплекса.

Ключевые слова: *динамическая система, нелинейность, устойчивость, бифуркации, распределенные системы.*

Результаты основополагающих работ по исследованию сложных динамических систем, описываемых нелинейными системами дифференциальных уравнений, получены на основе введения и анализа функций Ляпунова, то есть качественными методами. Однако общего алгоритма построения функции Ляпунова нет [1], эвристические приемы реализуются в частных случаях, так что, универсальный программируемый алгоритм исследования устойчивости нелинейных динамических систем пока не создан.

В данной статье представлен комплекс программ, реализующий алгоритм исследования нелинейных автономных динамических систем, основанный на разработанном универсальном методе исследования автономных нелинейных динамических систем без использования функций Ляпунова [2–4].

Комплекс предназначен для

– исследования динамической устойчивости решения автономных нелинейных систем обыкновенных дифференциальных уравнений вида

$$\dot{x} = X(x,r), \qquad (1)$$

где $x = (x_1, \ldots, x_n)$ – вектор фазовых координат, $-\infty<x_i<\infty$; $r = (r_1, \ldots, r_m)$ – вектор управляющих параметров; $X(x,r)$ – нелинейная вектор-функция;

– исследования структурной устойчивости нелинейных автономных динамических систем и построения их бифуркационных множеств;

– прогнозирования поведения динамических систем с целью построения функций, сохраняющих устойчивость этих систем.

Программы комплекса могут выполняться в автоматическом и интерактивном режимах.

Структура и особенности реализации программного комплекса

Комплекс имеет модульную структуру (рис. 1) и состоит из следующих независимых модулей: управляющий модуль, СУБД, транспортный модуль, модуль исследования динамической устойчивости (ИДУ), модуль исследования структурной устойчивости (ИСУ), модуль формирования управляющих воздействий (ФУВ), модуль графического пользовательского интерфейса (ГИ).

Модули комплекса взаимодействуют между собой по сети через транспортную программу. Транспортная программа и протокол передаваемых через нее сообщений образуют транспортную систему, являющуюся надстройкой над стеком протоколов TCP/IP с дополнительной системой авторизации и шифрования и реализованную на прикладном уровне модели OSI (ГОСТ Р ИСО 7498-2-99).

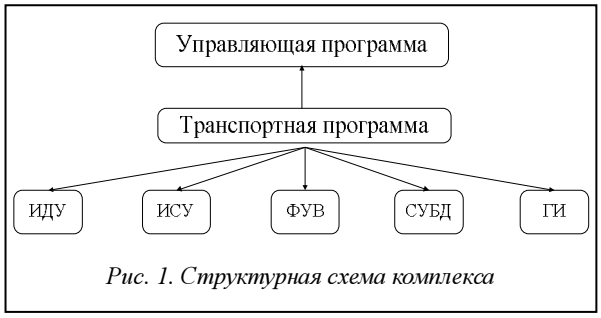


Рис. 1. Структурная схема комплекса

Модули комплекса могут выполняться одновременно на разных ПЭВМ, распределенных территориально и объединенных в сеть (Интернет или ЛВС). Такая распределенность многократно повышает быстродействие комплекса, так как ресурсоемкие расчеты могут выполняться параллельно (рис. 2), а передача данных между программами в составе комплекса может производиться на скоростях свыше 1 Гбит/с. Благодаря распределенности возможно использование комплекса на облачных платформах и в многопроцессорных системах.

Процесс работы комплекса организует управляющий модуль, координирующий запуск, завершение и взаимодействие других модулей. Управляющие модули могут подключаться друг к другу (рис. 3), образуя древовидные структуры для одновременного исследования независимых или последовательно соединенных динамических систем. Вычислительные модули могут подключаться динамически «в горячем режиме», не прерывая работу всего комплекса и других модулей. Путем динамического подключения вычислительных модулей и других комплексов осуществляется масштабируемость. Так как комплекс является распределенной системой, то его масштабируемость не ограничена.

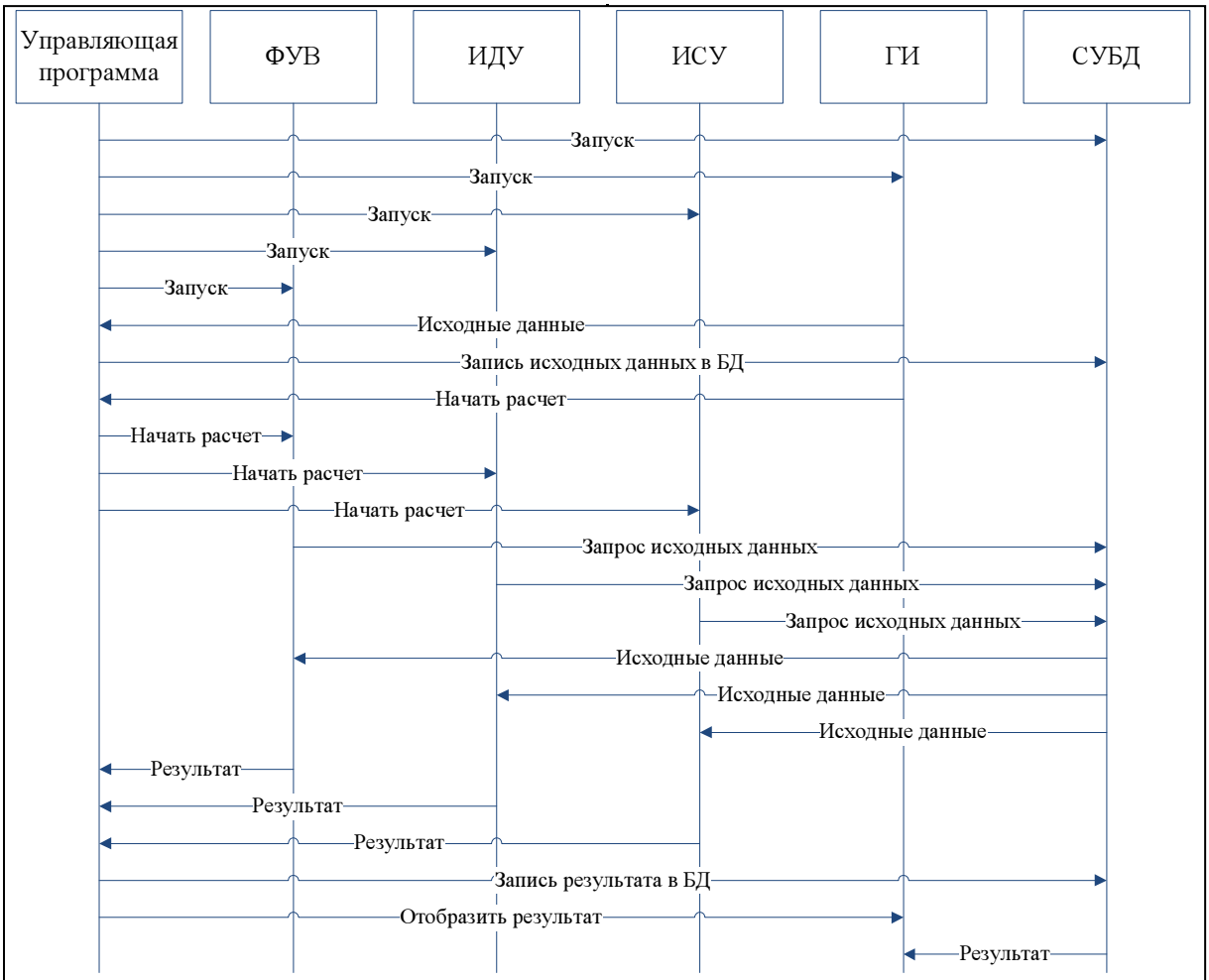


Рис. 2. Временная диаграмма работы комплекса

В комплексе используется нереляционная БД, обладающая высоким быстродействием и расширяемостью. Данные в ней представлены структурой ключ-значение для быстрого и легкого доступа к ним. Такая структура СУБД, в отличие от традиционных SQL-решений, не ограничивает быстродействие комплекса и его масштабируемость.

Интерактивное взаимодействие с оператором осуществляется через модуль графического пользовательского интерфейса. Для ввода исходных данных – размерности фазового пространства, размерности пространства управляющих параметров, нелинейной системы дифференциальных уравнений, описывающей исследуемую систему, – используется форма ввода исходных данных. Дифференциальные уравнения вводятся в символьной мнемонической форме.

После ввода исходных данных оператор дает команду начать расчет, который может быть прерван по запросу оператора.

После окончания расчета оператору выводится сообщение «КАТАСТРОФЫ», если динамическая система имеет катастрофы; «НЕУСТОЙЧИВА», если исследуемая система неустойчива, и сообщение «УСТОЙЧИВА» в обратном случае.

Если система неустойчива динамически, оператор может просмотреть области фазового про-

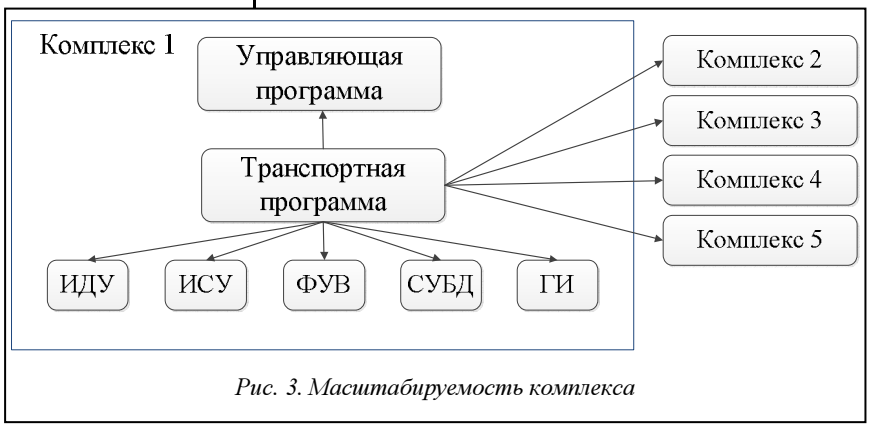


Рис. 3. Масштабируемость комплекса

странства, в которых проявляется неустойчивость. Если система неустойчива структурно, оператор может просмотреть области пространства управляющих параметров, в которых система имеет катастрофы. Оператор имеет возможность просмотреть графики функций управления, сохраняющих систему в устойчивом состоянии и построенных модулем ФУВ.

Все модули комплекса разработаны на высокоуровневом императивном языке C++ и высокоуровневом функциональном языке lisp с использованием библиотек Qt версии 4.6. В вычислительных модулях комплекса задействована система символьной математики Maxima версии 5.0.

Модуль ИДУ

На входе модуль получает исходное дифференциальное уравнение (1) в символьном представлении, вектор управляющих параметров $r_0 = (r_1^0, \ldots, r_m^0)$ и требуемую точность.

На выходе модуль выдает область U устойчивости исходной динамической системы (1) в пространстве фазовых координат.

Алгоритм преобразования входных данных в выходные [2] следующий.

1. Регистрация в транспортной системе.
2. Получение исходных данных из БД.
3. Построение сопряженной гамильтоновой системы уравнений по отношению к системе (1):

$$\dot{p}_i(t) = -\partial\left(\sum_{j=1}^{n} X_j(x_1, x_2, \ldots, x_n, r_0)\right)/\partial x_i,$$

i=1, ..., n. (2)

4. Построение характеристического уравнения сопряженной системы:

$$\lambda^n + S_1(-1)(-\lambda)^{n-1} + S_2(-1)^n(-\lambda)^{n-2} + \cdots$$
$$\ldots + S_{n-1}(-1)^n(-\lambda) + (-1)^n S_n = 0, \quad (3)$$

где $S_0 = 1$,

$$S_1 = (-1)^{n-1}\frac{\partial X_1}{\partial x_1}\ldots\frac{\partial X_{n-1}}{\partial x_{n-1}} + (-1)^{n-1}\frac{\partial X_1}{\partial x_1}\ldots\frac{\partial X_{n-2}}{\partial x_{n-2}}\frac{\partial X_n}{\partial x_n},$$

…,

$S_n = (-1)^n\frac{\partial X_1}{\partial x_1}\ldots\frac{\partial X_n}{\partial x_n} - \ldots - \frac{\partial X_1}{\partial x_n}\frac{\partial X_n}{\partial x_1}$ – суммы главных миноров n-го порядка матрицы сопряженной к (1) системы $\left(\frac{-\partial X_i}{\partial x_k} - \lambda_{i=k}\right)i = 1, \ldots, n\ k = 1, \ldots, n = 0$.

5. Построение из коэффициентов-функций характеристического уравнения системы (2) функциональных миноров Гурвица:

$$\Delta_0 = 1,$$

$$\Delta_1 = \begin{pmatrix} (-1)^{2n} & S_1(-1)^{n(n-1)} \\ S_3(-1)^{n(n-3)} & S_2(-1)^{n(n-2)} \end{pmatrix},$$

$$\Delta_n = \begin{pmatrix} (-1)^{2n} & \ldots & 0 \\ 0 & \ldots & S_n(-1)^n \end{pmatrix}. \quad (4)$$

6. Выявление методом градиентного спуска такой области U в пространстве фазовых координат, в которой функциональные миноры Гурвица (4) отрицательны.
7. Запись выявленной области U в БД.

Модуль ИСУ

На входе модуль получает исходное дифференциальное уравнение (1) в символьном представлении и требуемую точность.

На выходе модуль выдает бифуркационное множество R исходной динамической системы (1) в пространстве управляющих параметров.

Алгоритм преобразования входных данных в выходные [4] выглядит следующим образом.

1. Регистрация в транспортной системе.
2. Получение исходных данных из БД.
3. Построение сопряженной гамильтоновой системы уравнений (2) по отношению к системе (1).
4. Построение характеристического уравнения (3) сопряженной системы.
5. Построение из коэффициентов-функций характеристического уравнения системы (2) функциональных миноров Гурвица (4).
6. Выявление такой области R в пространстве управляющих параметров, где каждая точка является границей между областями, в одной из которых все функциональные миноры (4) положительны, а в другой нет.
7. Запись выявленной области R в БД.

На основании изложенного можно сделать следующие выводы. Комплекс является распределенной масштабируемой вычислительной системой с собственными механизмами хранения и обработки данных.

База вычислительных модулей комплекса использует мощный математический аппарат и реализует уникальные алгоритмы исследования динамических систем и процессов. Предусмотрена возможность быстрого динамического наращивания базы вычислительных модулей комплекса.

Разработанный комплекс программ обеспечивает исследование задач проектирования сложных динамических систем и процессов с нелинейными элементами.

Литература

1. Матросов В.М. Метод векторных функций Ляпунова: анализ динамических свойств нелинейных систем. М.: Наука, 2001.

2. Катулев А.Н., Кузнецов А.Ю. Алгоритм исследования устойчивости решений нелинейных автономных систем дифференциальных уравнений // Нелинейный мир. 2010. № 10. Т. 8. С. 616–620.

3. Кудинов А.Н., Катулев А.Н., Кузнецов А.Ю. Исследование устойчивости автономных нелинейных динамических систем // Динамические и технологические проблемы механики конструкций и сплошных сред: матер. XVI Междунар. симпоз. им. А.Г. Горшкова. М.: МАИ, 2010. Т. 1. С. 110–112.

4. Катулев А.Н., Кузнецов А.Ю. Исследование устойчивости автономных нелинейных динамических систем // Труды МАИ: электрон. журн. 2010. № 40. URL: www.mai.ru/science/trudy/.

УДК 681.3

ПРОГРАММНАЯ СИСТЕМА ДЛЯ ПОДДЕРЖКИ ПРИНЯТИЯ ОПЕРАТИВНЫХ ПЛАНОВЫХ РЕШЕНИЙ

А.В. Колесников, д.т.н. (Балтийский федеральный университет им. Иммануила Канта, г. Калининград, avkolesnikov@yandex.ru);
С.А. Солдатов (ООО «ГИМАС+», г. Калининград, soldatov@l-on.ru)

Рассматривается программная система для поддержки принятия оперативных плановых решений. Дается ее состав, описываются элементы. Приводятся результаты экспериментов на примере машиностроительного предприятия «Калининградгазавтоматика».

Ключевые слова: *система поддержки принятия решений, оперативное планирование, машиностроительное предприятие.*

Неотъемлемой частью производства является система планирования, предусматривающая планирование выполнения заказов, обеспечение поставки комплектации, равномерную загрузку участков предприятия, контроль перемещения деталей между производственными операциями. При применении задач *оперативно-производственного планирования* (ОПП) на реальных промышленных предприятиях было выявлено, что они характеризуются большой размерностью, сжатыми сроками выполнения и высокой ценой ошибки (до 15 % от ежемесячной прибыли предприятия), что делает их сложными для построения релевантной компьютерной модели и снижает достоверность результатов работы.

Многие отечественные и зарубежные ученые – Л.В. Канторович, К.Г. Татевосов, С.Н. Петраков, Д. Тейлор, Н. Рэйден и др. – в своих работах при рассмотрении задач ОПП используют различные аналитические, статистические, имитационные, структурные и интеллектуальные модели и методы. Но, как показал анализ, в известных методах и моделях ОПП отсутствует или имеет ограниченное применение важный механизм взаимодействия (координации) подзадач в процессе решения сложной задачи. Это обусловлено главным образом тем, что представление о задаче, выработанное в исследовании операций, теории принятия решения, искусственном интеллекте и системном анализе, для современного индустриального общества устарело.

Авторы данной статьи в своих исследованиях поставили цель разработать новый метод решения сложной задачи ОПП, учитывающий важный механизм взаимодействия (координации) подзадач, для повышения релевантности компьютерной модели реальной задаче. В результате были созданы модели сложной задачи с координацией, модели задачи координации и системы поддержки принятия решений с координацией, алгоритм координации в *системах поддержки принятия решений* (СППР), методика разработки функциональной гибридной интеллектуальной системы с координацией для ОПП [1, 2]. Внедрение программной

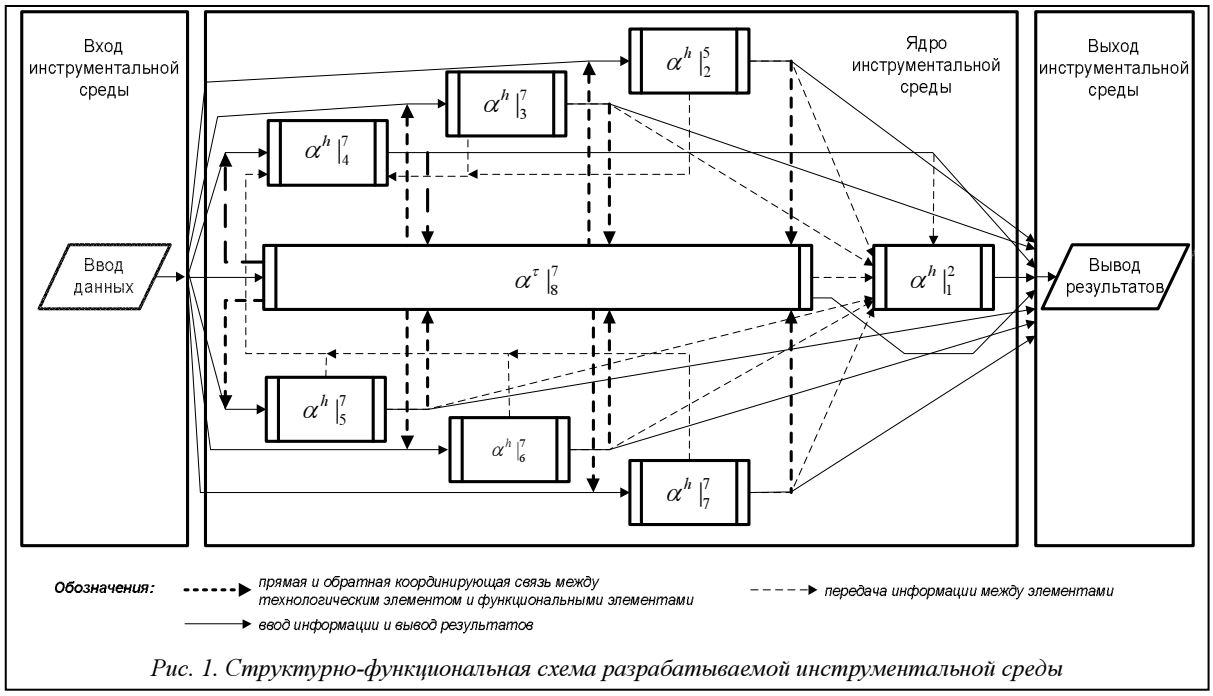


Рис. 1. Структурно-функциональная схема разрабатываемой инструментальной среды

системы, построенной по новому методу решения задачи ОПП, на реальном предприятии показало практическую ценность этой научной работы.

Опишем программную систему, теоретическим базисом которой является «Гибридная система планирования» (рис. 1).

Прямые и обратные связи (жирный пунктир) между технологическим и функциональными элементами обозначают передачу промежуточных результатов решения подзадач $\pi^h \in \Pi^h$ и получение функциональным элементом $\alpha^h|_j$, j=1, …, 7, координирующего воздействия. Порядок работы функциональных элементов определяется декомпозицией $\hat{\pi}^u$ сложной задачи π^u, а также рекомендациями экспертов.

В таблице приведены функциональные/технологические элементы инструментальной среды и методы их представления [3]. Верхняя цифра индекса обозначает базовый класс методов функциональных *гибридных интеллектуальных систем* (ГиИС) [4], нижняя – порядковый номер подзадачи.

Обозначение элемента	**Назначение элемента и методы представления**	**Код модели**	**Код подзадачи**
$\alpha^h\|_1^2$	Подзадача 1: «Прогнозирование результатов выполнения оперативного графика»; решается НС	НС	ПРГ1
$\alpha^h\|_2^5$	Подзадача 2: «Оптимизация количества выпускаемой продукции»; решается ГА	ГА	ОПТ1
$\alpha^h\|_3^7$	Подзадача 3: «Анализ соответствия имеющихся ресурсов требованиям КД по обеспечению выполнения плана производства»; решается ЭС	ЭС1	АНЛ1
$\alpha^h\|_4^7$	Подзадача 4: «Анализ соответствия имеющихся ресурсов требованиям ТД по обеспечению выполнения плана производства»; решается ЭС	ЭС2	АНЛ2
$\alpha^h\|_5^7$	Подзадача 5: «Учет, контроль и регулирование обеспечения материальными ресурсами»; решается ЭС	ЭС3	АНЛ3
$\alpha^h\|_6^7$	Подзадача 6: «Планирование и контроль выполнения ремонтных работ»; решается ЭС	ЭС4	АНЛ4
$\alpha^h\|_7^7$	Подзадача 7: «Планирование и контроль оплаты заказов»; решается ЭС	ЭС5	АНЛ5
$\alpha^\tau\|_8^7$	Подзадача 8: Технологический элемент решает *k*-задачу методами ЭС	ЭС6	КОР1

Примечание: НС – нейронная сеть; ГА – генетический алгоритм; ЭС – экспертная система; КД – конструкторская документация; ТД – технологическая документация.

Для передачи промежуточных результатов решения подзадач и получения функциональным элементом координирующего воздействия в ГиИС используются глобальные переменные – переменные-результаты и переменные-команды. Порядок работы функциональных элементов определяется декомпозицией $\hat{\pi}^u$ сложной задачи π^u, а также рекомендациями экспертов. Опишем детально автономные модели.

Функциональный элемент $\alpha^h|_1^2$. Предназначен для решения подзадачи ПРГ1 с помощью искусственной *нейронной сети* (НС). НС генерируется заново для каждого разработанного плана. Это связано с тем, что для экспертов наиболее важным аспектом прогноза является совместный выпуск указанных в плане изделий (а каждый план уникален!). Для обучения НС используются обучающие примеры, которые состоят из пары «требуемое количество изделия»–«было ли это количество изготовлено», вход и выход соответственно. Количество обучающих примеров и итераций обучения задается пользователем.

Функциональный элемент $\alpha^h|_2^5$. Предназначен для решения подзадачи ОПТ1 с помощью *генетического алгоритма* (ГА). Для ГА функция приспособленности представляет собой сумму единиц ресурса, затраченных на производство деталей каждого вида. Приспособленность рассчитывается как отношение затраченного ресурса к имеющемуся на складе. Так как для одного изделия можно задействовать несколько ресурсов, после определения максимально возможного количества изделия, которое можно изготовить с использованием только одного ресурса, выбирается минимальное значение количества изделия среди максимумов, полученных для каждого ресурса. Это позволяет получить плановый перечень с указанием максимального количества изделий, которые могут быть выпущены исходя из всех имеющихся материальных ресурсов.

Функциональные элементы $\alpha^h|_3^7$ – $\alpha^h|_7^7$ и *технологический элемент* $\alpha^\tau|_8^7$ реализованы по технологии *экспертных систем* (ЭС). ЭС содержат продукционные БЗ следующих экспертов из СППР (см. табл.): ЭС1 – главный конструктор, ЭС2 – главный технолог, ЭС3 – начальник отдела материально-технического снабжения, ЭС4 – начальник электромеханического отдела, ЭС5 – начальник отдела продаж, ЭС6 – начальник производственного центра. В системе используется метод рассуждений в прямом направлении.

Для ввода данных разработан модуль ввода и трансляции входных данных в язык системы, выполняющий первичный анализ полученных данных и в зависимости от их типа направляющий информацию соответствующему функциональному элементу. Модуль вывода результатов пред-

ставляет полученные результаты работы программной системы в виде графиков и текстового описания. Данные могут вводиться вручную или импортироваться из БД [3].

Результат работы системы – рекомендуемый оперативный график работы производства, прогноз его выполнения, а также количество использованных/необходимых для его выполнения ресурсов. В программе реализована возможность сравнения результатов решения для задачи с координацией и без, а также выдачи объяснений, отображающих динамику процесса формирования оперативного графика.

ПО написано на VB.NET; общий объем кода – более 5 000 строк. БД программы создана на основе СУБД MS SQL Express. Программа работает в среде Windows XP/Windows 2003, требует не менее 1 Гб на жестком диске, 1 Гб оперативной памяти, процессор с тактовой частотой не ниже 2,2 ГГц, предпочтительнее многоядерный.

Гибридная система планирования апробирована на данных завода «Калининградгазавтоматика». Объемы обработанной информации – более 5 000 заказов, содержащих в сумме более 13 000 позиций; количество инструментов и производственного оборудования – свыше 1 500; номенклатура производимых изделий и применяемых материалов – около 89 000. Размеры БЗ функциональных элементов – от 8 до 40, а технологического элемента (*k*-задача) – 15 продукций.

В ходе экспериментов выбирался месяц, задавалось количество планерок (от 2 до 15), выполнялся запуск системы для имитации работы СППР как последовательности планерок, прогнозировалась выполнимость плана НС, сравнивалась работа ГиИС в режиме без координации и с координацией, при изменении перечня заказов запускался ГА для оптимизации количества изготавливаемых изделий в зависимости от текущего количества ресурсов.

Результаты решения сложной задачи ОПП с учетом координации с участием всех экспертов и ЛПР на всех пятнадцати планерках приведены на рисунке 2.

В среднем по всем одиннадцати экспериментам относительная погрешность результатов решения задачи ОПП с учетом координации не превысила 1 %. Это позволяет утверждать, что разработанная программная система релевантно отображает моделируемые процессы и явления. Можно отметить, что учет координации при моделировании не приводит к улучшению или ухудшению результатов решения задачи ОПП, а только делает модель релевантнее реальной задаче. Без учета координации результаты моделирования решения задачи ОПП не соответствуют фактическим – относительная погрешность результатов достигает 36 %.

По мнению экспертов, рекомендации, полученные в ходе экспериментов по решению сложной задачи ОПП с учетом координации, могут рассматриваться как руководство к действию при отсутствии серьезных изменений во внешней среде предприятия – задержках расчетов, поставок и т.д.

Кроме того, экспертами отмечено существенное сокращение времени (не менее чем в два раза) на подготовку данных для планерок, поскольку часть необходимых данных можно оперативно получать из разработанной автоматизированной системы для решения задачи ОПП на заводе «Калининградгазавтоматика».

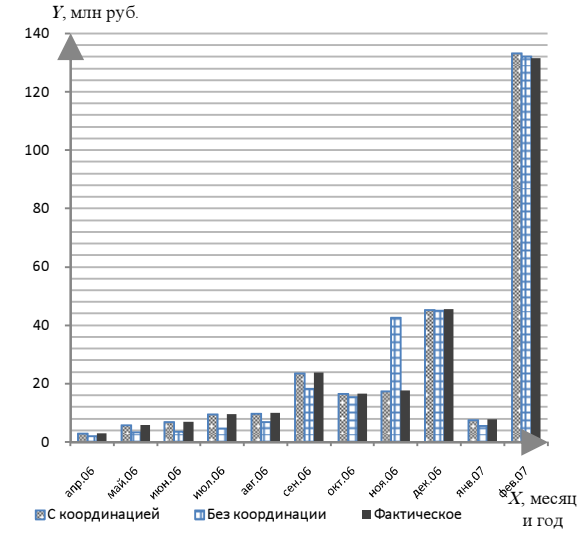


Рис. 2. График результатов решения задачи

Для оценки экономического эффекта от внедрения разработанной ГиИС с координацией для решения задачи ОПП на «Калининградгазавтоматике» проведен сравнительный анализ с мелкосерийным машиностроительным предприятием «Вест-Автоматика» (г. Калининград), на котором была внедрена аналогичная система. Экономическая эффективность от внедрения составила за год 9–10 % (5,4–6,5 млн рублей) к ожидаемой прибыли предприятия.

Полученные экспериментальные данные подтвердили теоретическое обоснование необходимости учета координации при решении сложной задачи ОПП, практическую значимость учета координации при решении сложной задачи ОПП, а также состоятельность разработанных моделей и релевантность полученной программной системы моделируемым процессам и явлениям.

Литература

1. Колесников А.В., Солдатов С.А. Теоретические основы решения сложной задачи оперативно-производственного планирования с учетом координации // Вестн. РГУ им. Иммануила Канта. Вып. 10: Сер. Физ.-мат. науки. Калининград: Изд-во РГУ им. И. Канта, 2009. С. 82–98.

2. Колесников А.В., Солдатов С.А. Алгоритм координации для гибридной интеллектуальной системы решения сложной задачи оперативно-производственного планирования // Информатика и ее применение. М.: ИПИ РАН. 2010. Т. 4.

3. Солдатов С.А. Гибридная интеллектуальная система поддержки принятия оперативных плановых решений для машиностроительных предприятий с мелкосерийным заказным производством. Инновации в науке и образовании-2007: докл. V Междунар. науч. конф. в номинации «Участник молодежного научно-инновационного конкурса У.М.Н.И.К.». Калининград: КГТУ, 2007. С. 58–61.

4. Колесников А.В., Кириков И.А. Методология и технология решения сложных задач методами функциональных гибридных систем. М.: ИПИ РАН, 2007. 387 с.

УДК 004.9

МНОГОАГЕНТНЫЙ ПОДХОД В СИСТЕМАХ ИНФОРМАЦИОННОЙ ПОДДЕРЖКИ УПРАВЛЕНЧЕСКИХ РЕШЕНИЙ

Н.М. Козьминых; А.А. Голованов, к.т.н.

(Вятский государственный университет, г. Киров, kzmnhn@yahoo.com, kzmnhn@gmail.com)

Изложен подход к созданию приложения информационной поддержки управленческих решений, основа которого представлена модулем многоагентной системы. Определены общая модель многоагентной системы и общие модели ее агентов, выделены основные типы агентов, участвующих в принятии решения.

Ключевые слова: *агент, модель агента, многоагентный подход, информационная поддержка управленческих решений.*

Многоагентный подход широко используется в создании программных модулей интеллектуальных информационных систем – от выполнения задач поиска информации и распознавания образов до принятия решений, в том числе управленческих. Исследования в области применения многоагентного подхода ведутся довольно давно, но задача построения универсального модуля *многоагентной системы* (МАС) в полной мере так и не решена. Причиной этого является узкая направленность исследуемых предметных областей применения.

В настоящей статье обсуждается обобщенная структура модуля МАС, предназначенного для информационной поддержки управленческих решений и обеспечивающая выполнение следующих функциональных возможностей:

– извлечение и объединение информации для дальнейшей ее обработки из разрозненных источников данных для получения достоверной и объективной информации;

– сохранение шаблонов пользовательских запросов, необходимых для получения часто используемых данных;

– использование настраиваемых помощников для отслеживания конкретных ситуаций;

– расширение функциональных возможностей системы путем увеличения способов анализа и интерпретации данных, то есть возможность дополнения определений отдельных показателей и их взаимосвязей, а также увеличения их количества в процессе эксплуатации системы;

– проведение автоматизированных исследований статистических данных, не требующих от пользователей специальных знаний, что также минимизирует число ошибок, обусловленных человеческим фактором;

– реализация универсального средства для решения схожих задач, то есть проведение аналитических исследований количественных показателей при помощи математико-статистического аппарата, что позволяет использовать одни и те же агенты вычислений для решения схожих задач в разных предметных областях.

Модуль рассматриваемой МАС состоит из автономных агентов, способных воспринимать ситуацию, принимать решения и взаимодействовать с себе подобными. Знания, необходимые для такой системы, отделены от ее программного кода и хранятся в онтологии, представляющей собой сеть понятий и отношений предметной области. Функционирование агентов осуществляется в рамках агентной платформы – среды, в которой могут существовать и взаимодействовать агенты [1, 2].

Модуль МАС обладает следующими свойствами:

– структура сообщества агентов является динамической относительно типов и количества членов сообщества;

– сообщество агентов основывается на принципах кооперации;

– структура сообщества агентов подразумевает распределенность (размещение агентов на различных компьютерах), что позволяет эффективно организовать доступ к распределенным источникам данных;

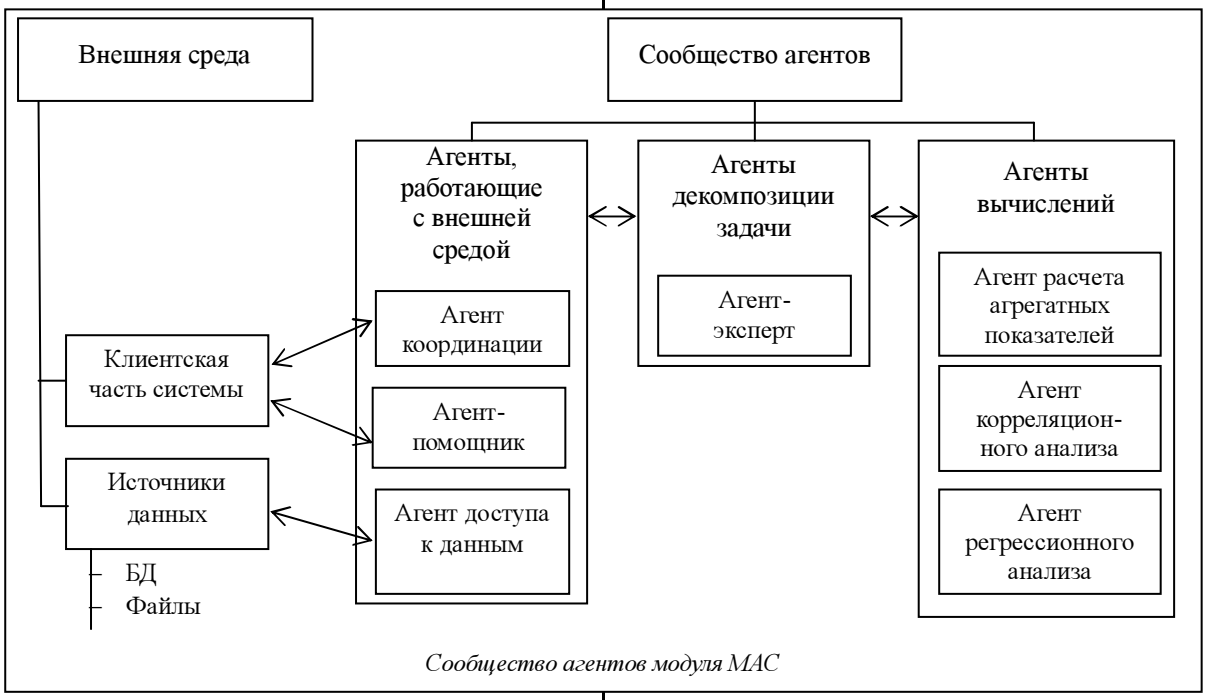
– агенты используют определенную предметную область для решения поставленных задач;

– агенты обеспечивают работу в асинхронном режиме;

– появление новых членов сообщества агентов или изменение функций некоторых агентов не требует перезагрузки всей информационной системы;

– структура сообщества базируется на требованиях FIPA (Federation of Intelligent Physical Agents) [3].

Модуль МАС представлен сообществом агентов, разработанных в соответствии с рекомендациями FIPA. В состав модуля МАС входят семь типов агентов (см. рис.), которые разделены на три группы в зависимости от основного назначения.



Сообщество агентов модуля МАС

Первая группа агентов взаимодействует с внешней средой и представлена координатором, помощником, а также агентом доступа к данным.

Вторая группа занимается декомпозицией поставленной задачи и включает экспертов предметной области. Агент-эксперт вводится в систему для решения различных специфичных задач.

Третья группа занимается непосредственно вычислениями и объединяет агента расчета агрегированных показателей, агентов корреляционного и регрессионного анализа.

В качестве теоретической основы для построения модели МАС была использована абстрактная модель FIPA: MAS=⟨Agent, Interaction⟩, где Agent – множество всех классов агентов; Interaction – множество всех взаимодействий между классами агентов.

Класс агента представляет общую структуру агента или коллекцию всех агентов с одними и теми же характеристиками внутренних состояний. Модель класса агента определена следующим образом: Agent=⟨LA_1, CM, BM, O⟩, где LA_1 – множество информационных атрибутов (идентификатор, имя, местоположение и т.д.); CM – коммуникационная модель (язык и методы для общения); BM – поведенческая модель (способы обработки сообщений); O – множество онтологий.

Согласованное взаимодействие двух или более агентов – это ядро многоагентной технологии. Агенты взаимодействуют друг с другом для достижения общей цели. Класс взаимодействий определен следующим образом: Interaction=⟨LA_2, CA, TA⟩, где LA_2 – множество фактических атрибутов, например, тип взаимодействия (кооперация, сотрудничество, конкуренция и т.д.); CA – множество характеристик соединения, таких как формат обмена, речевой акт, протокол взаимодействия, знания, язык и т.д.; TA – множество транспортных характеристик, таких как подход (клиент-серверный, точка-точка и т.д.), режим (синхронный, асинхронный) и др.

Для разрабатываемой системы модель МАС и модель класса взаимодействий оставим без изменения, модель класса агента адаптируем к структуре сообщества МАС и к кругу решаемых задач. Модель класса агента определим следующим образом: Agent=⟨LA_1, CM, BM, O, AD, SA⟩, где CM=⟨Z, W⟩ (Z={Z_i}, i=1÷I – множество входящих сообщений; W={W_s, W_a}, s=1÷S, a=1÷A – множество синхронных W_s и асинхронных W_a (информационных, управляющих и координационных) выходных сообщений); BM={BM_j}, j=1÷J – множество поведений агента, соответствующих ситуациям, которые зависят от состояний других агентов и их взаимодействий; O=⟨Cl, Atr, Val, Lim⟩, где Cl={Cl_k}, k=1÷K – множество классов – типовых множеств реальных сущностей, обладающих общими признаками; сущность определяется совокупностью значений свойств класса, которому она принадлежит; Atr={Atr_u}, u=1÷U – множество атрибутов классов, представляющих качественные характеристики и отличительные особенности

объектов; $Val=\{Val_u\}$, $u=1\div U$ – множество областей допустимых значений атрибутов; $Lim=\{Lim_b\}$, $b=1\div B$ – множество ограничений, описывающих принадлежность атрибутов классам, принадлежность допустимых значений атрибутам, совместимость классов, функциональные ограничения, заданные над значениями атрибутов; $AD=\{AD_n\}$, $n=1\div N$ – множество дополнительных функций, необходимых для выполнения поставленных задач и/или формирования ответных сообщений; SA – множество внутренних структур агента, описывающих его функциональное устройство в зависимости от его основного назначения.

Множество внутренних структур агента включает в себя: $SA=\langle SA^e, SA^t, SA^d\rangle$, где SA^e – множество агентов, взаимодействующих с внешней средой; SA^t – множество агентов-экспертов определенной предметной области; SA^d – множество агентов, занимающихся непосредственными вычислениями.

Модель агента, взаимодействующего с внешней средой: $SA^e=\langle MO, ER\rangle$, где $MO=\{MO_z\}$, $z=1\div Z$ – множество моделей окружения; $ER=\{ER_g\}$, $g=1\div G$ – множество действий, допустимых для агента.

Агенты, взаимодействующие с внешней средой, представлены координатором, помощником, а также агентом доступа к данным. Координатор и помощник являются инициаторами всех взаимодействий в сообществе агентов, поскольку именно они реагируют на все воздействия со стороны клиентской части системы (вход пользователя в систему, ввод запроса и т.д.).

Модель агента – эксперта определенной предметной области: $SA^t=\langle TQ, TR, TP, Plan\rangle$, где $TQ=\{TQ_m\}$, $m=1\div M$ – множество целей агента; $TR=\{TR_g\}$, $g=1\div G$ – множество действий, допустимых для агента; $TP=\{TP_z\}$, $z=1\div Z$ – множество вариантов декомпозиции задач (библиотека частичных планов); $Plan(TQ_m, TP_z)=TR_g$ – функция формирования плана действий агента (формирует упорядоченную последовательность действий агента из множества его допустимых действий TR_g), исходя из его текущей цели TQ_m и варианта декомпозиции задачи TP_z.

Модель агента, занимающегося непосредственными вычислениями: $SA_d=\langle Task, Par, DR\rangle$, где $Task=\{Task_m\}$, $m=1\div M$ – множество решаемых задач; $Par=\{Par_z\}$, $z=1\div Z$ – множество параметров; $DR=\{DR_g\}$, $g=1\div G$ – множество действий для выполнения конкретных задач.

Таким образом, описанная обобщенная структура модуля МАС позволяет использовать разрозненные источники данных, проводить автоматизированные исследования, не требующие специальной квалификации пользователя, и предоставлять единый аналитический аппарат обработки информации.

Разработанный прототип модуля МАС может составить основу при создании систем информационной поддержки управленческих решений для сбора, извлечения и анализа данных в узкоспециализированных областях применения.

Литература

1. Wooldridge M. An Introduction to MultiAgent Systems. Wiley, 2009.
2. Shamma Jeff. Cooperative Control of Distributed Multi-Agent Systems. John Wiley & Sons, 2008.
3. Bordini R.H., Dastani M., Dix J., Seghrouchni A. Multi-Agent Programming: Languages, Tools and Applications. Springer, 2009.

УДК 519.876.5

РЕШЕНИЕ ЗАДАЧ ПЛАНИРОВАНИЯ В КОАЛИЦИОННОЙ МОДЕЛИ

А.С. Зраенко (ФГУП «Уралгеоинформ», г. Екатеринбург, zraenko@yandex.ru);
В.П. Федотов, д.т.н. (Институт машиноведения УрО РАН, г. Екатеринбург, fedotov@imach.uran.ru);
К.А. Аксенов, к.т.н. (Радиотехнический институт Уральского государственного технического университета им. первого Президента России Б.Н. Ельцина, г. Екатеринбург, wiper99@mail.ru)

Рассмотрены метод теории составления расписаний и его применение для решения задачи планирования в коалиционной модели мультиагентного процесса преобразования ресурсов. На примере практической задачи разработан алгоритм составления планов выполнения работ.

Ключевые слова: *агент, мультиагентные системы, теория составления расписаний, процесс преобразования ресурсов, метод случайного поиска.*

В мультиагентных моделях [1] одним из типов актуальных задач является составление планов выполнения работ агентами и коалициями. Под *планом выполнения работ* будем понимать перечень намеченных к выполнению работ или мероприятий с определенными последовательностью,

объемами и сроками выполнения [2]. Поскольку все факторы, влияющие на планы выполнения работ, практически невозможно учесть, а интересы агентов часто различны, задача составления планов является многокритериальной, с нечетким множеством факторов. Решаются такие задачи, как правило, с помощью *теории составления расписаний* (ТСР) в два этапа [3]: получение плана путем использования определенного метода на основе исходных условий выполнения работ и формализуемых ограничений и его последующая доработка с целью максимального учета неформализуемых ограничений. Для разработки алгоритма составления планов выполнения работ необходимо формальное описание планов действий агентов и коалиций в коалиционной модели *мультиагентного процесса преобразования ресурсов* (МППР) [1].

При составлении плана выполнения работ нужно определить условия их выполнения, учитывая формализуемые (четкие), а также неформализуемые (нечеткие) ограничения. Для учета только формализуемых ограничений задачу составления плана выполнения работ можно решать как задачу целочисленного программирования. Однако при реальных объемах поступающих заказов данная задача не может быть решена за реальное время. Вследствие этого для задач такого типа будем считать необходимым использование методов ТСР по поиску приближенных решений.

В качестве основы алгоритма решения задачи составления планов выполнения работ агентами можно использовать следующие основные эвристические методы ТСР [3]: *метод ветвей и границ, алгоритм Литтла, метод случайного поиска.* Вследствие ориентации на последующее использование выбранного метода ТСР для решения задач в *мультиагентной системе поддержки принятия решений* (МСППР) одним из важных критериев метода должна быть его алгоритмическая простота. Среди наиболее простых эвристических методов ТСР, дающих достаточно точные решения, можно выделить *метод случайного поиска.*

Общее описание прикладной задачи планирования

Из прикладных задач составления планов выполнения работ выбран тип, связанный с ликвидацией лесных пожаров. Для ее решения необходимо создание мультиагентной модели для превентивной оценки количества ресурсов (люди, техника) для устранения пожаров с целью установления равновесия между финансовыми затратами и скоростью ликвидации пожаров. Ключевым моментом в данном случае является динамичность системы – лесные пожары возникают в произвольные моменты времени. В связи с этим нужно оставлять часть ресурсов и средств в резерве и при необходимости отзывать какие-либо из них. Опишем алгоритм работы коалиционной модели МППР для этой задачи.

1. *Инициация работ – выявление пожара.* Происходят определение агента A_1 – координатора работ и выделение ресурсов $\{Res_1, \ldots, Res_n\}$ и средств $\{Mech_1, \ldots, Mech_m\}$ из общего резерва ($Pool$) под его координацию. Агент A_1 планирует использование собственных ресурсов и средств по определенному алгоритму. Каждый агент A_i должен выделять ресурсы $\{Res_1, \ldots, Res_n\}$ и средства $\{Mech_1, \ldots, Mech_m\}$ на устранение лесного пожара путем проведения моделирования, исходя из следующих целей:

– наискорейшая ликвидация лесного пожара: $t\,(W_i) \rightarrow min;$

– минимизация финансовых затрат (то есть максимизация прибыли агента): $S \rightarrow max;$

– необходимость резервирования максимального количества ресурсов и средств в связи с ненулевой вероятностью возникновения другого пожара: $N \rightarrow max.$

2. *Выявление нового пожара.* В процессе устранения первого пожара существует вероятность выявления второго в другом месте. При этом в МСППР происходит определение агента A_2, который планирует возможность выполнения работы W_2 в указанный срок t_2 и с имеющимися резервными ресурсами $\{Res^{A1}{}_1, \ldots, Res^{A1}{}_t\}$ и средствами $\{Mech^{A1}{}_1, \ldots, Mech^{A1}{}_v\}$. При отсутствии необходимых ресурсов или средств агенту A_2 необходимо организовать взаимодействие с другими агентами $\{A_n, \ldots, A_m\}$ в МАС, в результате которого возможно создание коалиции K_p для совместного выполнения работы W_2. При этом прибыль S_{Aj} агента A_j определяется соразмерно количеству его ресурсов $\{Res^{Aj}{}_1, \ldots, Res^{Aj}{}_t\}$, участвующих в выполнении работы W_i коалицией K_p. Далее возможно возникновение третьего пожара и т.д.

3. *Возникновение конфликтов.* В случае нехватки ресурсов или средств возникает необходимость переговоров (в приведенном примере – агента A_2 с агентом A_1) на основе выбранных ими стратегий взаимодействий $\{Str_{A1}, Str_{A2}\}$. При этом в зависимости от стратегий может потребоваться проведение аукциона (основанного на величине последствий от пожара, Pos). При малой величине последствий часто целесообразно оставить ресурсы и средства в резерве, при большой возможно привлечение дополнительных ресурсов.

Формализация задачи планирования выполнения работ

Для последующего использования параметров задачи составления планов выполнения работ в качестве исходных данных алгоритма планирования необходима их формализация.

1. Формирование списков: агентов $\{A_1, \ldots, A_m\}$; работ $\{W_1, \ldots, W_f\}$; типов ресурсов $\{1, \ldots, t\}$;

ресурсов $\{Res^1_1, \ldots, Res^t_i\}$; типов средств $\{1, \ldots, v\}$; средств $\{Mech^1_1, \ldots, Mech^v_j\}$.

2. Определение условий выполнения работ $\{W_1, \ldots, W_f\}$. Взаимосвязи списков из п. 1 определяют следующие условия выполнения работ:

– агент A_i может управлять выполнением только одной работы W_i;

– агенты $\{A_1, \ldots, A_m\}$ могут использовать ресурсы любого типа $\{1, \ldots, t\}$ и средства любого типа $\{1, \ldots, v\}$ для выполнения работы W_i;

– агенты $\{A_1, \ldots, A_m\}$ не могут использовать средство $Mech_k$ без ресурса Res_k для выполнения работы W_i;

– ресурсы типа k могут выполнять работу W_i только с использованием средств типа k;

– в зависимости от типа $\{1, \ldots, t\}$ ресурсы могут быть привязаны только к одной работе W_i до ее окончания либо переходить с одной работы W_i на другую W_j;

– средства всех типов $\{1, \ldots, v\}$ могут переходить с одной работы W_i на другую W_j.

Условия выполнения работ, не определяющие взаимосвязи списков из п. 1:

– минимальный шаг выполнения работы W_i – 1 день;

– ресурсы $\{Res_1, \ldots, Res_t\}$ и средства $\{Mech_1, \ldots, Mech_t\}$ не могут быть захвачены агентом A_j во время их использования другим агентом A_i (нет агентов, имеющих абсолютный приоритет по захвату ресурсов и средств);

– для выполнения работы W_i захват средств $\{Mech_1, \ldots, Mech_t\}$ не является необходимым;

– весовой коэффициент V_i необходимости выполнения i-го формализуемого ограничения R_i плана выполнения работ PW_i может принимать следующие значения: $V_i=\{1, 2, \ldots, 100\}$;

– весовой коэффициент Q_i необходимости выполнения i-го неформализуемого ограничения L_i плана выполнения работ PW_i может принимать следующие значения: $Q_i=\{1, 2, \ldots, 10\}$.

Начальное значение суммарного коэффициента несоответствий i-го плана выполнения работ K_i принимается равным сумме всех коэффициентов несоответствия формализуемых ограничений $\{V_1, \ldots, V_n\}$ и неформализуемых ограничений $\{Q_1, \ldots, Q_m\}$:

$$K_i=\sum_{n=1}^{N} V_n + \sum_{m=1}^{M} Q_m, \qquad (1)$$

где N – число формализуемых ограничений, а M – неформализуемых.

3. Определение формализуемых $\{R_1, \ldots, R_n\}$ и неформализуемых $\{L_1, \ldots, L_m\}$ ограничений; их сообразных весовых коэффициентов несоответствия $\{V_1, \ldots, V_n\}$ и $\{Q_1, \ldots, Q_m\}$ (для определения важности).

К формализуемым ограничениям $\{R_1, \ldots, R_n\}$ относятся следующие:

– R_1 (V_i=100): каждая работа должна быть выполнена вовремя (фактическое время выполнения работы T не может быть больше расчетного максимального времени T_O, $T \le T_O$);

– R_2 (V_i=10): не должно быть окон в работе ресурсов (окна в работе средств допустимы); число окон в работе ресурсов определяется как сумма свободных ресурсов по всем рабочим дням и рассчитывается по следующей формуле:

$$R_2=\sum_{i=1}^{D} w_i * Res_i, \qquad (2)$$

где D – количество дней, на которые составляется план выполнения работ PW_i (текущий день, $d \in (1, D)$); w_i – параметр незанятости ресурса, $w_i=\{1, 0\}$.

К неформализуемым (или сложноформализуемым ограничениям) $\{L_1, \ldots, L_m\}$ относятся следующие:

– L_1 (Q_1=4): загрузка ресурсов одного типа должна быть равномерной с учетом коэффициентов скорости работы, зависящих от дня недели, личных событий, настроения и т.д.;

– L_2 (Q_2=1): необходимо техническое обучение ресурсов всех типов, направленное на повышение скорости выполнения работ (с периодичностью и продолжительностью, связанными с типом ресурса, его квалификацией, временем работы, текущей загруженностью и т. д.);

– L_3 (Q_3=2): необходимо техническое обслуживание средств (с периодичностью и продолжительностью, связанными с типом средства, его сроком службы, временем эксплуатации и т.д.).

4. Загрузка исходных данных по работам $\{W_1, \ldots, W_i\}$:

– дата возникновения пожара, D_b;

– крайняя дата завершения работ, D_e;

– требуемый состав действий, $\{D_1, \ldots, D_s\}$.

Пример загрузки исходных данных представлен в таблице 1.

Таблица 1

Исходные данные по работам

Наименование работы	Дата возникновения пожара	Дата завершения работ	Требуемые действия
W_1	13.05.10	26.05.10	$<D_1, D_2, D_3>$
W_2	21.05.10	21.06.10	$<D_1, D_3>$
W_3	18.06.10	10.09.10	$<D_1, D_2, D_3, D_4, D_5>$

5. Формирование параметров работ $\{W_1, \ldots, W_i\}$:

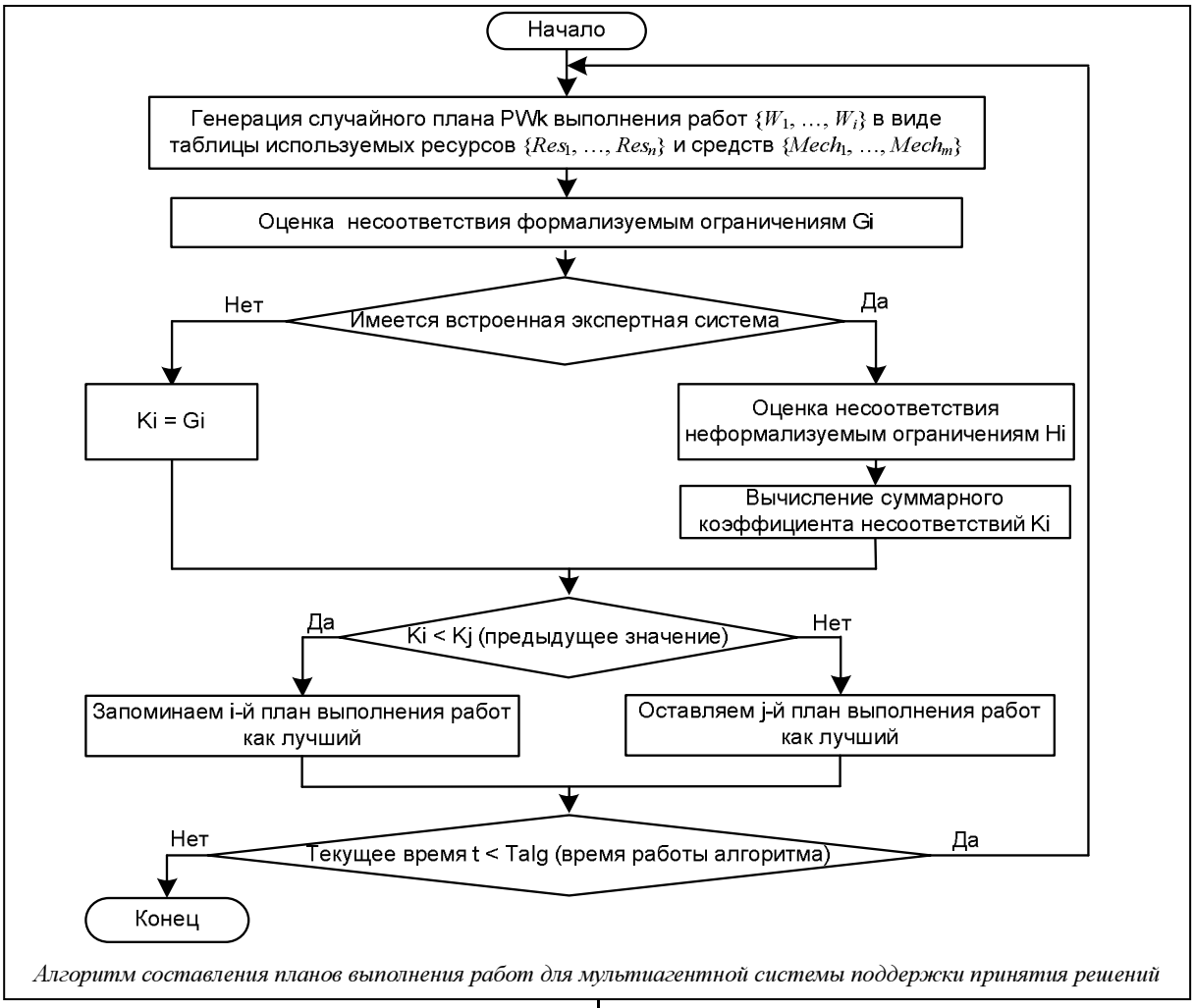
– количество ресурсов типа 1, N_{Res1};

– параметр привязанности ресурсов типа 1 только к одной работе до ее окончания, $P_{Res1}=\{1, 0\}$;

– количество ресурсов типа 2, N_{Res2};

– параметр привязанности ресурсов типа 2 только к одной работе до ее окончания, $P_{Res2}=\{1, 0\}$;

– количество ресурсов типа n, N_{Resn};



Алгоритм составления планов выполнения работ для мультиагентной системы поддержки принятия решений

– параметр привязанности ресурсов типа n только к одной работе до ее окончания, P_{Res3}= ={1, 0};

– количество средств типа 1, N_{Mech1};

– количество средств типа 2, N_{Mech2};

– количество средств типа m, N_{Mechm};

– расчетное максимальное время выполнения работы в днях, T_O.

Так как каждая работа W_j состоит из действий $\{D_1, \ldots, D_i\}$, состав и очередность которых определены в базе знаний мультиагентной системы (GKB): $W_j=D_1+D_2+\ldots+D_n$, то для расчета максимального времени выполнения работы (см. табл. 2) используется формула

$$T_O=t_1+t_2+\ldots+t_n, \quad (3)$$

где t_i – среднее время выполнения действия D_i (из GKB).

Таблица 2

Пример расчета параметров работ

Работа	Параметр								
	N_{Res1}	P_{Res1}	N_{Res2}	P_{Res2}	N_{Res3}	P_{Res3}	N_{Mech1}	N_{Mech2}	T_O
W_1	2	1	1	0	1	0	0	0	3
W_2	4	1	1	0	1	0	1	1	5
W_3	1	1	1	0	1	0	0	1	3
W_4	1	1	2	0	1	0	0	1	4

Алгоритм составления планов выполнения работ

В рамках создания мультиагентной системы поддержки принятия решений необходима разработка алгоритма составления планов выполнения работ. Данный алгоритм основан на методе случайного поиска и представлен на рисунке.

Описание алгоритма.

1. Генерация случайного плана PW_i выполнения работ $\{W_1, \ldots, W_i\}$ (с использованием данных п. 2 и п. 3 данного алгоритма в случае не первого прогона) в виде таблицы используемых ресурсов $\{Res_1, \ldots, Res_t\}$ и средств $\{Mech_1, \ldots, Mech_t\}$, в столбцах которой указываются дни проведения работ, а в строках – сами работы. Необходим ежедневный подсчет свободных ресурсов и средств.

Пример плана выполнения работ PW_i представлен в таблице 3. В примере принято, что агенты-координаторы $\{A_1, A_2, A_3, A_4\}$ управляют работами $\{W_1, W_2, W_3, W_4\}$ соответственно. Все агенты могут координировать действия собственных ресурсов и средств; всего имеется: $4Res_1$ (сотрудники МЧС), $2Res_2$ (пожарные), $1Res_3$ (лесники), $1Mech_1$ (спецтехника МЧС), $1Mech_2$ (пожарные

вертолеты). Каждая работа должна начинаться с Res_1 и заканчиваться Res_3; Res_1 может использовать $Mech_1$, а $Res_2 – Mech_2$.

Таблица 3

Пример плана распределения ресурсов и средств по работам

Работа	Дни				
	1-й	2-й	3-й	4-й	5-й
W_1	$2Res_1$, Res_2	$2Res_1$, Res_2	*«простой»*	Res_2, Res_3	
W_2	Res_1, $Mech_1$	$2Res_1$, $Mech_1$	Res_2, $Mech_2$	$4Res_1$, Res_2	Res_3
W_3	Res_1, Res_2	Res_2, $Mech_2$, Res_3	Res_1, Res_3		
W_4			Res_1, Res_2	Простой	Res_1, $2Res_2$, $Mech_2$
Свободные ресурсы и средства	Res_3, $Mech_2$	–	$2Res_1$, $Mech_1$	$Mech_1$, $Mech_2$	$3Res_1$, $Mech_1$

В данной таблице выполнение работ W_1, W_2, W_3 начинается в 1-й день, работы W_4 – в 3-й день. Крайний срок выполнения работы W_1 – 4-й день, работы W_2 – далее 5-го дня, работ W_3 и W_4 – 5-й день. Из таблицы видно, что работу W_4 агент A_4 не может выполнить с помощью своих ресурсов и средств.

2. Оценка несоответствия неформализуемым ограничениям $\{R_1, \ldots, R_n\}$. Расчет суммарного коэффициента несоответствия формализуемым ограничениям G_x производится по следующей формуле: $G_x = \sum_{i=1}^{n} f_i * V_i$, (4)

где x – текущий вариант плана выполнения работ; n – количество формализуемых ограничений; f_i – число невыполнений i-го формализуемого ограничения; V_i – весовой коэффициент необходимости выполнения формализуемого ограничения R_i плана выполнения работ PW_i.

Пример вычисления суммарного коэффициента несоответствия формализуемым ограничениям G_x (на основе данных из таблицы 3): G_1=1*100+6*10==160.

Оценка несоответствия неформализуемым ограничениям $\{L_1, \ldots, L_m\}$ проводится при наличии *экспертной подсистемы* (ЭС), содержащей информацию по накопленному опыту работы и/или наиболее успешные примеры принятия решений в определенных ситуациях из других регионов РФ. Наличие и полнота данной ЭС не являются обязательными для работы алгоритма, ее цель – получение наилучшего решения путем максимально полного учета неформализуемых ограничений. При отсутствии ЭС – переход к п. 5; определяем $K_x=G_x$.

Экспертная подсистема должна иметь структуру, включающую исходные данные и экспертную информацию.

Пример структуры ЭС.

Тип неформализуемого ограничения: L_1 (загрузка ресурсов одного типа должна быть равномерной с учетом коэффициентов производительности (скорости работы), зависящей от дня недели, личных событий, настроения и т.д).

Исходные данные: 1) стаж работы Res_2^1 – 23 года, Res_2^2 – 1 год, Res_2^3 –7 лет; 2) возраст Res_2^1 – 53 года, Res_2^2 – 19 лет, Res_2^3 – 39 лет; 3) сегодняшняя дата – 21.05.10; 4) завтра день рождения у Res_2^3.

Информация для поддержки принятия решения: 1) наибольший коэффициент производительности у людей со стажем работы более 5 лет и в возрасте до 45 лет; 2) для послепраздничных дней необходимо учитывать снижение средней производительности ресурсов.

Результирующая интерпретация неформализуемого ограничения: на 22.05.10 (второй день в таблице 3) необходимо Res_2^3 оставить в резерве.

С использованием результирующих интерпретаций неформализуемых ограничений рассчитаем суммарный коэффициент несоответствия неформализуемым ограничениям H_x по следующей формуле: $H_x = \sum_{i=1}^{m} d_i * Q_i$, (5)

где x – текущий вариант плана выполнения работ; m – количество неформализуемых ограничений; d_i – число невыполнений i-го неформализуемого ограничения; Q_i – весовой коэффициент необходимости выполнения i-го неформализуемого ограничения L_i плана выполнения работ.

Пример вычисления суммарного коэффициента несоответствия неформализуемым ограничениям H_x (на основе данных из таблицы 3): H_1=1*4=4.

3. Вычисление суммарного коэффициента несоответствий K_x путем сложения суммарного коэффициента несоответствия формализуемым ограничениям G_x и суммарного коэффициента несоответствия неформализуемым ограничениям H_x по следующей формуле: $K_x=G_x+H_x$. (6)

Пример: K_x=160+4=164.

4. Выбор лучшего плана выполнения работ. Если суммарный коэффициент несоответствий x-го плана выполнения работ K_x меньше соответствующего коэффициента несоответствий y-го плана выполнения работ K_y, ранее выбранного как лучший, $K_x<K_y$, то запоминаем x-й план выполнения работ как лучший. Иначе оставляем лучшим y-й план выполнения работ.

Пример: K_1=170, K_2=164, $K_2<K_1$, поэтому запоминаем план выполнения работ PW_2 как лучший.

5. Проверка истечения заранее определенного времени работы алгоритма T_{alg}. Если текущее время $t<T_{alg}$, переходим к п. 1. Иначе – конец алгоритма.

Итак, можно отметить, что для решения рассмотренной прикладной задачи планирования, связанной с ликвидацией лесных пожаров, создана МСППР, использующая алгоритм составления планов выполнения работ.

Алгоритм основан на методе случайного поиска теории составления расписаний. Он позволяет планировать использование ресурсов и средств агентами и коалициями при устранении пожаров с целью установления равновесия между финансовыми затратами и скоростью ликвидации пожаров.

Литература

1. Зраенко А.С., Аксенов К.А., Ван Кай. Коалиционная модель мультиагентного процесса преобразования ресурсов // Науч.-технич. ведомости СПбГУ. СПб: Изд-во политехн. ун-та, 2009. № 5. С. 156–161.

2. Чепасов В.И., Кулов С.К., Раимов Ф.Ф. Алгоритмическая и программная реализация задачи о расписании: учеб. пособие. Оренбург: Оренбург. гос. ун-т, 1999. 192 с.

3. Шахбазян К.В., Тушкина Т.А. Обзор методов составления расписаний для многопроцессорных систем. Л.: Наука, 1975. С. 229–258.

4. Береговых Ю.В., Васильев Б.А., Володин Н.А. Алгоритм составления расписания занятий // Искусств. интел. 2009. № 2. С. 50–56.

УДК 004.4

СОЧЕТАЕМОСТНЫЕ ОГРАНИЧЕНИЯ В СИСТЕМЕ АВТОМАТИЧЕСКОГО СИНТАКСИЧЕСКОГО АНАЛИЗА

М.Г. Мальковский, д.ф.-м.н.; Н.В. Арефьев
(Московский государственный университет им. М.В. Ломоносова, malk@cs.msu.su)

Предложена структура компьютерного словаря сочетаемости, содержащего описания различных типов ограничений на сочетаемость слов. Описан метод, позволяющий использовать словарь сочетаемости для улучшения качества автоматического синтаксического анализа.

Ключевые слова: *автоматический синтаксический анализ, компьютерный словарь, ограничения на сочетаемость слов.*

Задача автоматического анализа текстов на естественном языке возникает в самых различных приложениях: машинный перевод, информационный поиск, извлечение фактов из текстов, автоматическое реферирование и др. Для большинства приложений выполнения поверхностного анализа, основанного, например, на поиске ключевых слов, недостаточно – требуется учитывать различные лингвистические явления, в том числе синтаксические отношения. В данной работе рассматривается проблема учета ограничений на сочетаемость слов в процессе автоматического выделения синтаксических отношений в тексте (синтаксического анализа). Описанный в статье подход к решению этой проблемы реализован в системе автоматического синтаксического анализа Treeton, создаваемой на факультете ВМК МГУ [1].

Алгоритм синтаксического анализа, реализованный в Treeton, базируется на идее эвристического перебора, на каждом шаге которого строятся новые синтаксические связи между словами или словосочетаниями анализируемого предложения. С помощью эвристической функции оцениваются как окончательные структуры, покрывающие анализируемое предложение целиком, так и промежуточные, порождаемые на каждом шаге анализа. Отметим, что эвристическая функция в Treeton также называется штрафной, а ее значение – штрафом синтаксической структуры, поскольку это значение тем больше, чем серьезнее нарушение языковых норм структурой. Использование штрафной функции позволяет отбрасывать заведомо ошибочные гипотезы на ранних этапах перебора, а также упорядочивать результаты работы анализатора. В работе [1] предлагалась штрафная функция, учитывающая только топологические свойства оцениваемых структур (штрафовались пересечение стрелок синтаксических связей, большое количество выходящих из одной вершины стрелок и т.п.), при этом не принималось во внимание конкретное лексическое наполнение структур. Как показала практика, такая штрафная функция часто не отличает правильные структуры от неправильных. Например, предложная группа может быть связана как с глаголом, так и с существительным, поэтому для каждой из фраз *съесть пирог с черникой*, *съесть пирог с удовольствием* анализатор построит как правильную структуру (*пирог→с→черникой, съесть→с→удовольствием*), так и неправильную (*съесть→с→черникой* и *пирог→с→удовольствием*) и, оценивая только топологические свойства, не сможет выбрать нужную.

Авторами предлагается новая штрафная функция, учитывающая сочетаемость слов, описанную в специальном компьютерном словаре сочетаемости. В процессе синтаксического анализа при построении новой связи r от слова w_1 к слову w_2 по-

лученная конструкция $w_1 \rightarrow_r w_2$ проверяется на соответствие словарной информации. Если в словаре нет описания сочетаемости ни для одного из связанных слов, считается, что оба слова свободно сочетаются с любыми словами, следовательно, конструкция $w_1 \rightarrow_r w_2$ допустима и не штрафуется. Иначе конструкция проверяется на соответствие приведенным описаниям и в случае несоответствия штрафуется. Таким образом, словарь задает ограничения на сочетаемость слов. Структуры, не соответствующие этим ограничениям, штрафуются, за счет чего правильные структуры получают преимущество при анализе и на выходе анализатора появляются первыми (как по времени, так и по расположению в списке результатов).

Структура компьютерного словаря сочетаемости

В словаре описываются три типа ограничений сочетаемости слов: морфосинтаксические, лексические и семантические [2]. Ограничения можно представить в виде набора троек $<S_1, r, S_2>$ (далее используется более наглядная форма записи – $S_1 \rightarrow_r S_2$), где r – тип синтаксической связи; S_i – либо лексема, либо сема (название семантического класса), либо знак «*» (любая лексема). Морфосинтаксические ограничения характеризуют возможные типы исходящих синтаксических связей (например, **genet** и **acc** – связи с именной группой соответственно в родительном или винительном падежах), при этом не накладываются ограничения на слова, находящиеся на другом конце связи, поэтому S_2=*. Морфосинтаксические ограничения могут быть указаны как для лексем (***купить*** $\rightarrow_{\mathbf{acc}}$ *), так и для семантических классов (**ЕМКОСТЬ** $\rightarrow_{\mathbf{genet}}$ *). Лексические и семантические ограничения описывают множества слов, которые могут находиться на конце синтаксической связи. В случае лексических ограничений S_1 и S_2 – лексемы (***букет*** $\rightarrow_{\mathbf{genet}}$ ***вино***). В случае семантических ограничений на одном из концов связи (или на обоих) указывается сема (***букет*** $\rightarrow_{\mathbf{genet}}$ **ЦВЕТЫ**). Принадлежность слов к семантическим классам и родовидовые отношения между классами также описываются в словаре. Так, можно указать, что *астры, розы, васильки* – **ЦВЕТЫ**, тогда словосочетания *букет астр / роз / васильков* будут считаться допустимыми. Также можно связать родовидовым отношением классы **ЦВЕТЫ** и **РАСТЕНИЯ**, тогда сочетаемость класса **РАСТЕНИЯ** (например, ***поливать*** $\rightarrow_{\mathbf{acc}}$ **РАСТЕНИЯ**) будет унаследована классом **ЦВЕТЫ** (станут допустимыми словосочетания *поливать астры / розы / васильки*).

Словарь сочетаемости содержит два типа информации: статистическую и онтологическую. Статистическая информация вносится в словарь в результате автоматического анализа корпуса текстов [3]. Она представляет собой количественные оценки сочетаемости и может использоваться без дополнительной обработки в процессе синтаксического анализа. Онтологическая информация вносится в словарь в процессе работы экспертов (в том числе по обобщению имеющейся статистической информации), а также при импорте данных из существующих словарей. Для эффективного формирования онтологической информации эксперты используют специальные инструменты, помогающие выполнять рутинную работу (поиск примеров в корпусе, выявление похожих по смыслу слов и прочее).

Подсистема тестирования синтаксического анализатора позволяет следить за тем, как внесенные в словарь сочетаемости изменения отражаются на качестве работы анализатора. Такой комбинированный подход к описанию сочетаемости позволяет обеспечить изначально широкий охват лексики (который сложно обеспечить вручную), а затем повышать качество синтаксического анализа текстов некоторой предметной области за счет улучшения критичных для качества анализа лингвистических описаний (например, описаний терминов этой предметной области).

Отметим, что в текстах разных предметных областей одно и то же слово может иметь разную сочетаемость (например, можно *свернуть диалоговое окно*, но нельзя его *зашторить*). Поэтому для достижения наилучших результатов в процессе синтаксического анализа наряду со словарем сочетаемости общей лексики желательно использовать специализированный словарь сочетаемости, сформированный на корпусе текстов той предметной области, которой принадлежат анализируемые тексты.

Рассмотрим структуры данных, в которых хранится информация о сочетаемости, – тензор сочетаемости и матрицу семантических классов.

Тензор сочетаемости представляет собой тензор третьего ранга, два измерения которого соответствуют лексемам и семантическим классам, а третье – синтаксическим отношениям. В ячейке тензора $<S_1, r, S_2>$ хранится разнообразная информация о словосочетаниях типа $S_1 \rightarrow_r S_2$: их частотность в корпусе (обозначим ее через $<S_1, r, S_2>_f$), оценка меры их неслучайности ($<S_1, r, S_2>_p$), а также экспертная оценка данного типа словосочетаний ($<S_1, r, S_2>_o$). Таким образом, в тензоре представлена как статистическая, так и онтологическая информация. Для хранения в тензоре морфосинтаксических ограничений сочетаемости вводится специальное значение третьего измерения «*». Ячейка тензора $<S_1, r, *>$ показывает, допускает ли слово или класс S_1 исходящую связь типа r. Не приводя подробных выкладок, отметим, что оценка меры неслучайности вычисляется по следующей формуле: $<S_1, r, S_2>_p = \dfrac{\hat{P}(S_1, r, S_2)}{\hat{P}(S_1)\hat{P}(S_2)}$, где $\hat{P}(S_1, r, S_2)$ – оценка вероятности появления в тек-

сте словосочетания типа $S_1 \rightarrow_r S_2$; $\hat{P}(S_i)$ – оценка вероятности появления слова или одного из слов семантического класса S_i (ср. с мерой MI [4]). Оценки вероятностей вычисляются как относительные частоты соответствующих событий. Если S_1 является семантическим классом, то $<S_1, r, S_2>_p$ обнуляется в том случае, когда менее половины слов класса S_1 сочетаются с S_2 по связи r (аналогично для S_2). Таким образом, семантический класс наследует только общую для входящих в него слов сочетаемость. В ячейках $<S_1, r, *>_p$ вместо меры неслучайности хранится оценка условной вероятности $P(r \mid S_1)$. Для оценки $<S_1, r, S_2>_o$ эксперт использует трехбалльную шкалу: правильно, неправильно, сомнительно. По мнению авторов, трехбалльная шкала оптимальна для подобных задач, поскольку двоичная система (правильно–неправильно) заставляет искусственно сводить сомнительные случаи к одному из двух вариантов, а введение большего числа градаций в отсутствие объективных критериев выбора между ними понижает эффективность работы эксперта, ничего не предлагая взамен.

Используя тензор сочетаемости, любое слово или класс S_1 можно представить в виде вектора с компонентами, соответствующими контекстам (парам r_k, S_i), в которых встречается слово, а значение соответствующей компоненты равно $<S_1, r_k, S_i>_p$ (заметим, что в данный вектор можно также включить компоненты, соответствующие входящим связям $<S_i, r_k, S_1>_p$). Если ввести расстояние между контекстными векторами [4], появляется возможность численно оценивать смысловую схожесть слов и семантических классов. Эта возможность используется, в частности, для автоматизированного выявления семантических классов (кластеризации слов) [3].

Матрица семантических классов содержит информацию о принадлежности слов семантическим классам и о родовидовых отношениях между классами. Как и в случае с тензором сочетаемости, ячейка $<S, C>$ матрицы хранит статистическую и онтологическую информацию: оценку вероятности принадлежности слова (или вложенности класса) S классу C ($<S, C>_p$), вычисленную с использованием контекстных векторов, и экспертную оценку принадлежности (вложенности) по трехбалльной шкале ($<S, C>_o$).

Использование словаря сочетаемости в процессе синтаксического анализа

Ограничения сочетаемости слов используются при оценке структур, порождаемых в процессе перебора. Оценка итоговой синтаксической структуры, являющейся вариантом анализа входного предложения, представляет собой норму штрафного вектора, одной из компонент которого является штраф за нарушение структурой ограничений сочетаемости (другие компоненты связаны с топологическими огрничениями). Данный штраф сводится к оценке условных вероятностей конструкций $w_1 \rightarrow_r w_2$, из которых состоит итоговая структура:

$$pen(Struc) = -\log\hat{P}(Struc) =$$
$$= -\log\prod\hat{P}(r, w_2 \mid w_1) = -\sum\log\hat{P}(r, w_2 \mid w_1),$$

где произведение и сумма берутся по всем входящим в структуру конструкциям $w_1 \rightarrow_r w_2$. В тех случаях, когда оценка вероятности равна нулю, вместо ее логарифма берется заранее фиксированное достаточно большое число (конструкция сильно штрафуется). Заметим, что введенная штрафная функция обладает свойствами аддитивности и монотонности. Как показано в [5], при использовании штрафной функции с такими свойствами у синтаксического анализатора появляется привлекательная особенность: результаты анализа можно использовать, не дожидаясь окончания работы анализатора (или даже остановив его в любое время), при этом алгоритм гарантирует, что выданные результаты лучше (меньше оштрафованы) всех остальных.

Условная вероятность конструкции $w_1 \rightarrow_r w_2$ складывается из вероятностей наличия исходящей из w_1 связи r и нахождения на другом конце связи слова w_2: $P(r, w_2 \mid w_1) = P(r \mid w_1)P(w_2 \mid w_1, r)$ и оценивается по словарю сочетаемости:

$$\hat{P}(r \mid w_1) = \max_{S_1}\hat{P}(r \mid S_1)\hat{P}(w_1 \in S_1),$$

$$\hat{P}(w_2 \mid w_1, r) = \max_{S_1, S_2}\hat{P}(S_2 \mid S_1, r)\hat{P}(w_1 \in S_1)\hat{P}(w_2 \in S_2),$$

где максимумы берутся по множеству, включающему в себя как само слово w_i, так и всевозможные семантические классы, содержащие w_i: $S_i \in \{w_i\} \cup \{C_k \mid P(w_i \in C_k) > 0\}$. Оценки вероятностей в правых частях равенств вычисляются исходя из онтологической информации, если она доступна, либо на основе собранной статистики:

$$\hat{P}(r \mid S) = \begin{cases} 0, <S, r, *>_o = \text{неправильно} \\ 1, <S, r, *>_o = \text{правильно} \\ <S, r, *>_p \text{ иначе} \end{cases}$$

$$\hat{P}(S_2 \mid S_1, r) = \begin{cases} 0, <S_1, r, S_2>_o = \text{неправильно} \\ 1, <S_1, r, S_2>_o = \text{правильно} \\ <S_1, r, S_2>_p \, P(S_2) < S_1, r, *>_p^{-1} \text{ иначе} \end{cases}$$

$$\hat{P}(w \in S) = \begin{cases} 0, <w, S>_o = \text{неправильно} \\ 1, <w, S>_o = \text{правильно или } S = w \\ <w, S>_p \text{ иначе} \end{cases}$$

$$\hat{P}(S) = \sum_{w_i \in S}\hat{P}(w_i \in S)\hat{P}(w_i).$$

Таким образом, даже если конструкция $w_1 \rightarrow_r w_2$ ни разу не встретилась в корпусе текстов, она будет признана допустимой, если найдется класс S_1 (или S_2), содержащий w_1 (w_2) и сочетающийся с w_2

(w_1), либо найдется пара сочетающихся классов, один из которых содержит w_1, а другой w_2.

Подводя итог, можно отметить, что на базе рассмотренного в данной статье подхода в системе Treeton был реализован модуль оценки соответствия синтаксических структур описанным в словаре сочетаемости ограничениям, используемый в процессе синтаксического анализа.

Для поддержки работы со словарем сочетаемости созданы подсистема сопровождения и развития компьютерных словарей сочетаемости, включающая инструменты автоматического извлечения информации о сочетаемости из корпуса текстов и коррекции этого процесса экспертом, а также подсистема тестирования синтаксического анализатора, позволяющая оценивать качество его работы, визуализировать результаты анализа, выявлять ошибки анализа и устранять их причины (например, ошибочные входы словаря сочетаемости).

Литература

1. Мальковский М.Г., Старостин А.С. Система Treeton: Анализ под управлением штрафной функции // Программные продукты и системы. 2009. № 1. С. 33–35.

2. Апресян Ю.Д. Лексическая семантика: 2-е изд.: избран. тр.: Т. 1. М.: Издат. фирма «Восточная лит-ра» РАН, 1995. 472 с. (Языки русской культуры).

3. Арефьев Н.В. Формирование словаря сочетаемости для системы автоматического синтаксического анализа // Научные исследования и их практическое применение. Современное состояние и пути развития '2011: Сб. науч. тр. SWorld. Междунар. науч.-практич. конф. Одесса: Черноморье. 2011. Т. 4. С. 35–39.

4. Julie Weeds, David Weir, Diana McCarthy. Characterising measures of lexical distributional similarity // Proceedings of the 20th international conference on Computational Linguistics, COLING-2004. Geneva, Switzerland, pp. 1015–1021.

5. Старостин А.С., Арефьев Н.В., Мальковский М.Г. Синтаксический анализатор «Treevial». Принцип динамического ранжирования гипотез // Компьютерная лингвистика и интеллектуальные технологии: матер. ежегод. Междунар. конф. «Диалог» (26–30 мая 2010 г., Бекасово). М.: Изд-во РГГУ. 2010. Вып. 9 (16). С. 477–490.

УДК 004.89

ПАРАЛЛЕЛЬНАЯ СИСТЕМА АВТОМАТИЧЕСКОЙ ТЕКСТОВОЙ КЛАССИФИКАЦИИ

Е.В. Котельников, к.т.н.; Т.А. Пескишева
(Вятский государственный гуманитарный университет, Kotelnikov.EV@gmail.com, peskisheva.t@mail.ru)

Описаны структура и алгоритм работы параллельной системы автоматической текстовой классификации на основе метода опорных векторов. Рассмотрены блоки, реализующие основные этапы работы системы, особое внимание уделено процессу обучения классификатора. Приведены результаты экспериментов на коллекции текстов Reuters-21578, подтверждающие эффективность разработанной параллельной системы.

***Ключевые слова**: текстовая классификация, метод опорных векторов, SVM, параллельная система, автоматическая обработка текста.*

Для автоматической обработки значительных объемов текстовой информации применяются различные программные системы, предназначенные для поиска, аннотирования, машинного перевода, извлечения фактов и др. Одним из основных этапов автоматической обработки текстов является тематическая классификация, цель которой – отнесение текстовых документов к одной или нескольким заданным категориям по определенным признакам. Качественная автоматическая классификации делает процесс анализа текстовой информации менее трудоемким и более эффективным по времени выполнения и релевантности полученных результатов.

Как показывают исследования [1], наилучшие результаты в решении задачи автоматической текстовой классификации демонстрирует метод опорных векторов (Support Vector Machines – SVM), предложенный В.Н. Вапником [2], для решения задач распознавания образов двух классов. Классификация с применением SVM сводится к обучению и распознаванию. Наиболее трудоемким является процесс обучения, для которого используется коллекция обучающих документов с известными рубриками. По этим документам формируются обучающие векторы. В задаче тематической текстовой классификации каждый вектор соответствует текстовому документу и представляет собой набор весов ключевых слов (терминов), входящих в данный документ. Вес дает числовую оценку значимости данного термина для определения тематики текста [1]. На сформированных векторах обучается SVM-классификатор. Для оценки качества обучения используется специальная тестовая коллекция (ее можно выделить из обучающей и не использовать для обучения), на которой классификатор тестируется, а полученные ответы сравниваются с требуемыми (рубрики тестовых документов известны). В процессе распознавания система автоматически определяет рубрики новых, ранее не предъявляемых ей документов.

Существующие системы и модули текстовой классификации, в том числе использующие SVM, демонстрируют приемлемую скорость и точность обработки лишь на небольших и средних по объему данных. В случае значительного роста объема обрабатываемой информации, а также увеличения числа рубрик, по которым необходимо классифицировать документы, их производительность существенно снижается. Отдельной проблемой является организация эффективного подбора параметров классификатора.

Решение этих проблем возможно с помощью использования параллельных реализаций методов классификации для многопроцессорных вычислительных систем и комплексов.

В данной работе предлагаются структура и алгоритм функционирования параллельной системы автоматической текстовой классификации на основе метода опорных векторов, позволяющей эффективно осуществлять обучение тематического классификатора и распознавание тематических рубрик текстовых документов.

Структура системы

В структуру предлагаемой параллельной системы автоматической текстовой классификации (рис. 1) входят два вида модулей: модуль главного узла и модули вычислительных узлов. В модуле главного узла содержатся блоки, управляющие процессом параллельной обработки документов в вычислительной системе. Каждый модуль вычислительного узла отвечает за работу со своими подмножествами документов и рубрик.

В блоке предварительной обработки текстовый документ преобразуется из исходного вида в формат, принятый в системе: происходят очистка текста от знаков форматирования, выделение отдельных слов (tokenization, или графематический анализ), приведение всех слов к единому регистру, удаление слов, не несущих смысловую нагрузку (стоп-слов). Соответствующий управляющий блок модуля главного узла назначает документы коллекции вычислительным узлам на основе алгоритма распределения документов.

В блоке морфологического анализа определяются нормальная форма и грамматические характеристики слов. Морфологический анализ в описываемой системе выполняется программой mystem (*http://company.yandex.ru/technology/mystem*). На основе различных слов всех документов формируется локальный словарь узла, который передается на главный узел и участвует в создании глобального словаря.

Блок формирования векторной модели предназначен для представления каждого текста в виде вектора признаков на основе информации, полученной с выхода блока морфологического анализа

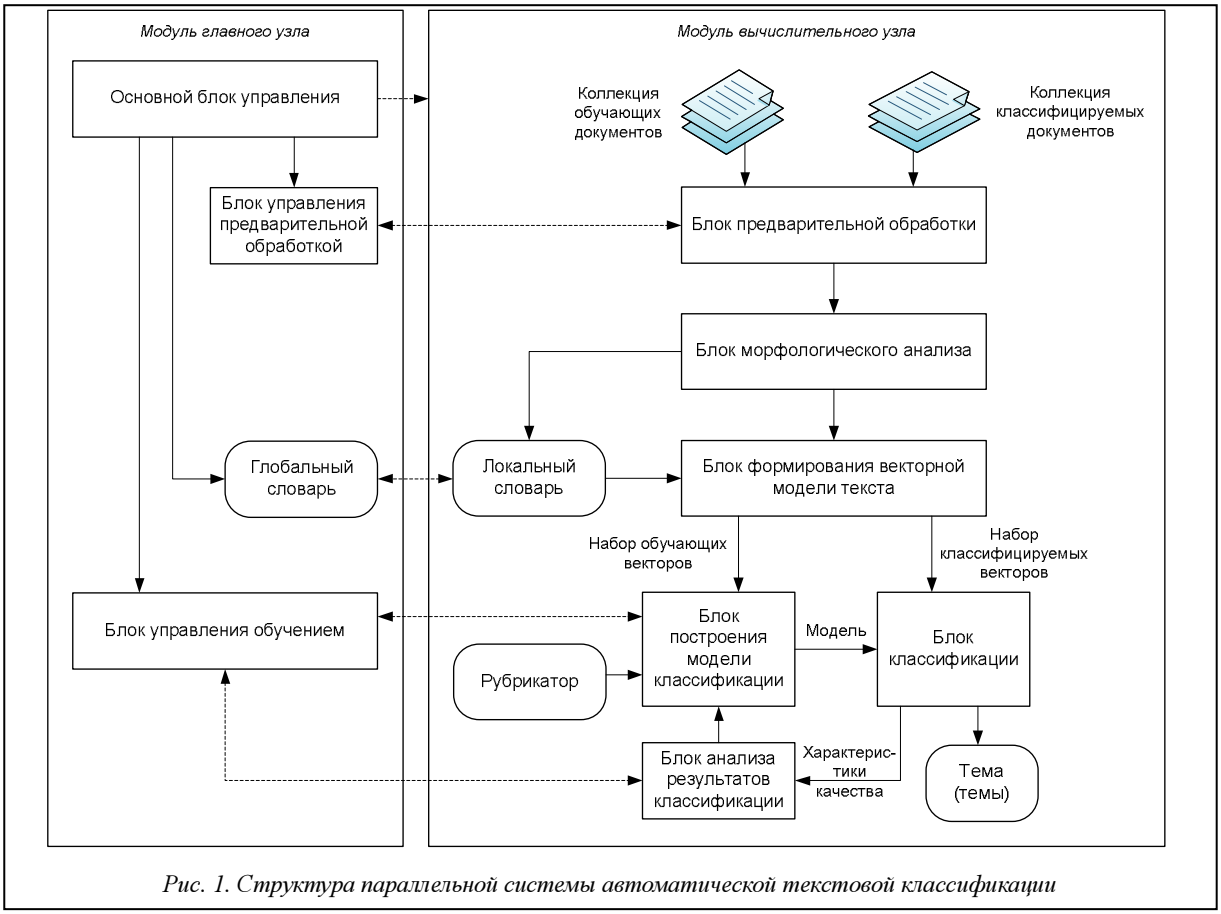


Рис. 1. Структура параллельной системы автоматической текстовой классификации

и из локального словаря. Признак – это вес термина в тексте, размерность вектора признаков равна числу терминов в глобальном словаре. Взвешивание производится в соответствии с подходом TF.IDF [1], где вес термина вычисляется как произведение локального и глобального весов. Локальный вес зависит от частоты употребления термина в данном документе, а глобальный – от распределения термина по документам всей коллекции.

В блоке построения модели классификации происходит обучение SVM на основе обучающих векторов и рубрикатора, содержащего список рубрик. Результатом обучения является модель, представленная набором опорных векторов.

Блок классификации автоматически определяет рубрику документа с использованием модели классификации. Выходом этого блока для каждого документа (вектора) является множество тем, к которым относится данный документ. Для коллекции документов, помимо множества тем, выходом также будут являться характеристики качества классификации – точность, полнота, F1-мера [1].

На основе характеристик качества в блоке анализа качества классификации принимается решение о продолжении процесса обучения с другими параметрами или о его завершении.

Блок управления обучением в модуле главного узла контролирует процесс подбора параметров классификатора и управляет назначением рубрик вычислительным узлам на основе алгоритма распределения рубрик.

В представленную структуру заложены два уровня параллелизма – на уровне модулей и на уровне блоков. Параллелизм на уровне модулей предполагает размещение модулей на разных узлах вычислительной системы и их совместную параллельную работу, например, на основе технологии MPI. Параллелизм на уровне блоков заключается в возможности организации параллельного функционирования каждого из блоков, так, на рисунке 1, например, на основе технологии OpenMP.

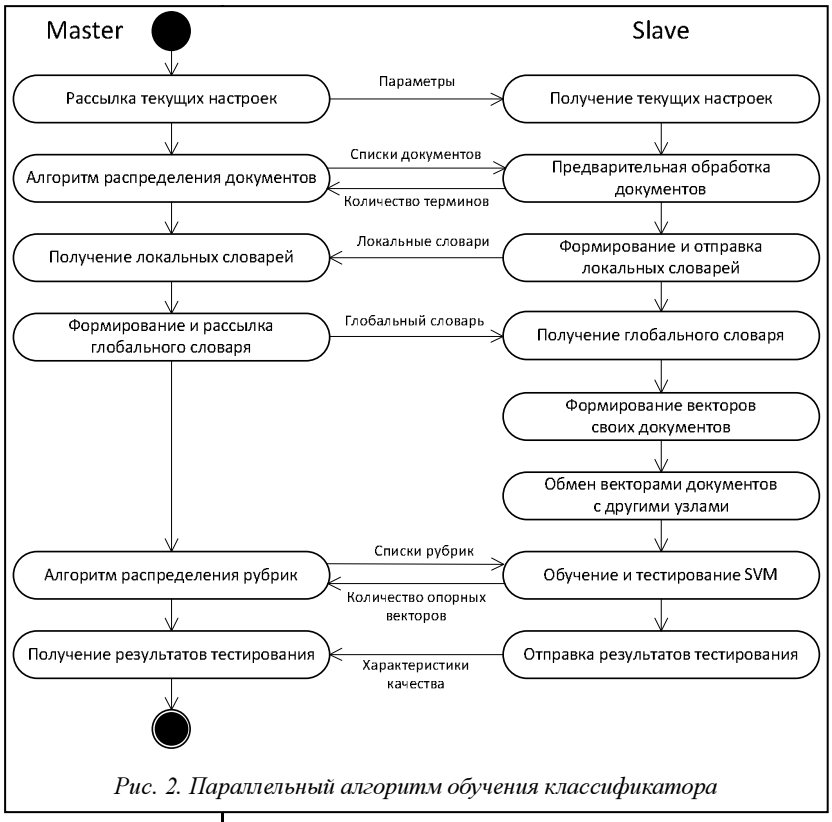


Рис. 2. Параллельный алгоритм обучения классификатора

Алгоритм обучения классификатора

Самым трудоемким этапом работы системы текстовой классификации является обучение классификатора. Это связано с высокой временно́й сложностью как формирования векторных моделей документов, так и обучения SVM. Ситуацию осложняет необходимость подбора оптимальных параметров классификатора.

Параллельная реализация процесса обучения позволит повысить эффективность работы всей системы. На рисунке 2 предложена схема работы параллельного алгоритма обучения для параллельной системы автоматической текстовой классификации.

Алгоритм обучения использует принцип master–slave, суть которого в следующем. Главный узел (master) передает подчиненным вычислительным узлам (slave) задания, которые они должны выполнить. Вычислительные узлы независимо друг от друга выполняют свои задания, затем полученные результаты возвращают главному узлу.

До начала работы алгоритма на всех вычислительных узлах должны находиться файлы, содержащие обучающие и тестовые документы. В начале алгоритма основной блок управления главного узла рассылает на все подчиненные узлы текущие настройки работы, в частности, диапазоны значений для подбора оптимальных параметров классификатора.

Затем блок управления предварительной обработкой на основе алгоритма распределения документов рассылает на вычислительные узлы списки документов. Указанный алгоритм работает по так называемой жадной схеме. В общем случае работа жадного алгоритма (greedy algorithm) заключается в принятии оптимальных решений на каждом этапе, допуская, что конечное решение также окажет-

ся оптимальным. На практике при решении задачи сбалансированного распределения нагрузки жадный алгоритм почти всегда дает решение, в достаточной степени близкое к оптимальному.

Распределение документов может быть основано на информации о размере документов и о количестве терминов в каждом документе. При этом первый вид информации доступен сразу же, а второй позволяет более адекватно определить время обработки документа, то есть возможна ситуация, когда время обработки объемного документа с малым количеством терминов меньше времени обработки небольшого документа с большим количеством терминов.

Первоначально в алгоритме распределения документов известен только размер каждого документа, на основании этой информации и происходит распределение. После первой итерации цикла подбора оптимальных параметров становится известно количество терминов в каждом документе, пересылаемое вычислительными узлами на главный узел, и алгоритм распределения корректируется в соответствии с этой новой информацией.

На следующем этапе вычислительные узлы осуществляют предварительную обработку и морфологический анализ своих документов, формируют и отправляют на главный узел локальные словари, в которых присутствуют термины только из своих документов. Главный узел объединяет локальные словари в глобальный и рассылает его подчиненным узлам.

На основе общего глобального словаря вычислительные узлы формируют векторы своих документов и, поскольку обучение и тестирование классификатора должны происходить на всех векторах, обмениваются друг с другом сформированными векторами.

Блок управления обучением главного узла распределяет список всех рубрик по узлам. Распределение также происходит с применением жадного алгоритма [3], на первой итерации цикла подбора оптимальных параметров используется количество обучающих документов для каждой рубрики, на последующих – информация о количестве опорных векторов, полученная от вычислительных узлов после процесса обучения SVM. Количество опорных векторов является характеристикой, более адекватно определяющей время обучения, чем размер обучающих документов.

Обучение и тестирование SVM происходят на вычислительных узлах в блоках построения модели классификатора и классификации. Характеристики качества классификации передаются главному узлу, а он на их основе принимает решение о продолжении или завершении процесса обучения.

Временна́я сложность алгоритма

Выполним оценку временно́й сложности алгоритма в случае, когда коллекция является сбалансированной, то есть отсутствуют серьезные различия между документами по размеру и между рубриками по количеству обучающих документов. Для этого выделим этапы с наиболее высокой скоростью роста – формирование векторов документов, обмен векторами и обучение SVM.

Введем следующие обозначения: N – общее количество документов в коллекции; N_{train} – количество обучающих документов; M – число вычислительных узлов; D – размер глобального словаря.

Формирование векторов документов имеет сложность $\mathrm{O}\left(\frac{N}{M}\cdot D\right)$, поскольку каждый узел работает со своей порцией данных в среднем из N/M документов, а для построения каждого вектора требуется осуществлять поиск по глобальному словарю.

Временна́я сложность обмена векторами описывается следующим выражением:

$$\mathrm{O}\left(M\cdot(M-1)\cdot\frac{N}{M}\right)=\mathrm{O}\left(M\cdot N\right), \qquad (1)$$

поскольку каждый узел должен передать свою порцию данных всем другим узлам.

Временна́я сложность алгоритма обучения SVM сильно зависит от расположения входных данных в пространстве признаков; на практике можно использовать квадратичную зависимость:

$$\mathrm{O}\left(\left(\frac{N_{train}}{M}\right)^2\right). \qquad (2)$$

Таким образом, временна́я сложность разработанного параллельного алгоритма обучения A описывается следующим выражением:

$$\mathrm{O}\left(A\right)=\mathrm{O}\left(\frac{N}{M}\cdot D\right)+\mathrm{O}\left(M\cdot N\right)+\mathrm{O}\left(\left(\frac{N_{train}}{M}\right)^2\right). \qquad (3)$$

Из выражения (3) видно, что при увеличении количества вычислительных узлов обмен данными (второе слагаемое) будет занимать все большее время, а это повлияет на характеристики ускорения и эффективности.

Чтобы оценить этот объем, нужно определить размер векторов, которыми узлы обмениваются между собой. Суммарный объем векторов обычно равен 1–2 объемам текстовой коллекции, по которой они формируются. Для коллекции Reuters-21578 объем векторов составляет 8 Мб; с учетом того, что каждый узел пересылает свою часть векторов каждому узлу, суммарный трафик при этом в кластере с 30 узлами можно оценить примерно в 30×8=240 Мб. При скорости сети 100 Мбит/с и с учетом служебных данных Ethernet и MPI такой объем информации передается за 25 секунд.

В случае несбалансированной коллекции в выражении (3) возрастает вклад первого и третьего слагаемых из-за неравномерного распределения документов по узлам.

Подбор параметров

В описываемой системе предусмотрен подбор оптимальных параметров классификатора, которые можно разделить на четыре группы: 1) параметры предобработки и морфологического анализа; 2) параметры словаря; 3) параметры векторной модели; 4) параметры SVM. Подбор осуществляется на основе метода скользящего контроля по n блокам (n-fold cross-validation). В этом методе для каждого набора параметров классификатора обучающие данные разбиваются на n одинаковых блоков, обучение осуществляется на n–1 блоках, а на оставшемся блоке тестируется полученный классификатор и рассчитываются характеристики качества обучения. Такое обучение повторяется n раз, каждый блок один раз используется в качестве тестового. Затем характеристики качества обучения усредняются и считаются оценкой качества обучения для данного набора параметров.

Весь процесс скользящего контроля повторяется для каждого набора из исследуемого диапазона параметров. Оптимальным считается набор параметров, при котором достигаются максимальные значения характеристик качества обучения.

Условия проведения экспериментов

Разработанные структура и алгоритм функционирования параллельной системы автоматической текстовой классификации были реализованы в среде Microsoft Visual Studio 2010 на языках C# и C/C++ на основе принципов объектно-ориентированного программирования. Для обучения SVM использовалась библиотека LIBSVM [4]. Целевой платформой приложения выбрана кластерная система, механизм передачи сообщений реализован с использованием библиотеки MPI.NET версии 1.0 [5]. Распараллеливание внутри блоков осуществлялось на основе технологии OpenMP (C/C++) и классов пространства имен System.Threading платформы Microsoft.Net.

Расчеты проводились на вычислительном кластере Вятского государственного гуманитарного университета, состоящем из 30 вычислительных узлов и одного головного. Каждый узел представляет собой персональный компьютер с процессором Intel Core 2 Duo 2 ГГц и 2 Гб оперативной памяти. Узлы связаны сетью Fast Ethernet. Кластер функционирует под управлением Microsoft Windows HPC Server 2008.

Для экспериментального исследования использовалась коллекция финансовых новостей агентства Reuters (Reuters-21578, Distribution 1.0) [6]. Обучающий набор документов был выделен в соответствии с общепринятым подходом, названным «ModApte split», при этом исключались документы, не помеченные ни одной рубрикой. Таким образом, обучающий набор в экспериментах был представлен 7 775 документами, тестовый – 3 019 документами, каждый из которых относится к одной или нескольким из 115 рубрик.

Перед проведением экспериментов на все вычислительные узлы была скопирована текстовая коллекция Reuters-21578 объемом 9 Мб, суммарный трафик составил 9×30 = 270 Мб; процесс копирования занял менее 1 минуты.

Результаты экспериментов

Тестирование проводилось с целью измерения характеристик производительности параллельной системы – ускорения и эффективности. Ускорение вычисляется как отношение времени решения задачи на одном вычислительном узле ко времени решения на p вычислительных узлах. Эффективность вычисляется как отношение ускорения к количеству задействованных узлов и отражает долю времени выполнения алгоритма, в течение которой вычислительные узлы реально задействованы для решения задачи.

Для измерения ускорения и эффективности была выбрана сложная вычислительная задача – подобраны параметры классификатора для коллекции Reuters-21578 методом скользящего контроля и проведен ряд запусков на разном количестве вычислительных узлов. Время выполнения на одном узле составило 37 466 с (примерно 10,5 часа). Зависимость времени выполнения на кластерной системе, ускорения и эффективности от разного количества задействованных узлов показана на рисунках 3–5.

Из этих зависимостей видно, что, во-первых, при увеличении количества узлов время работы существенно снижается и происходит значительное ускорение, во-вторых, максимальная эффективность 80–85 % достигается при 2–8 вычислительных узлах, затем последовательно уменьшается до 50 %. Снижение эффективности объясняется особенностями коллекции Reuters-21578, имеющей крайне неравномерное распределение документов по рубрикам – 54 рубрики из 115 представлены менее чем 10 документами. Данное обстоя-

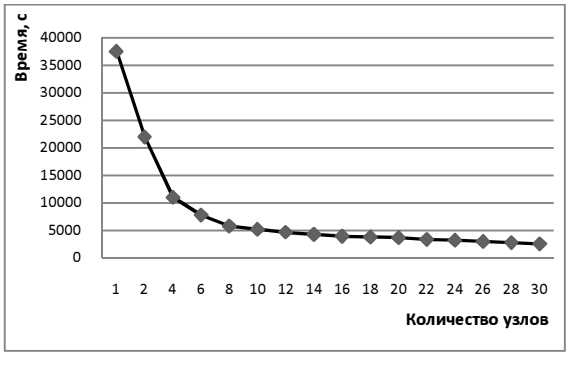


Рис. 3. Зависимость времени работы от количества узлов

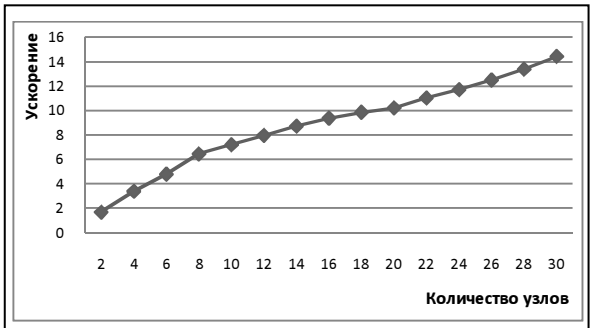


Рис. 4. Зависимость ускорения от количества узлов

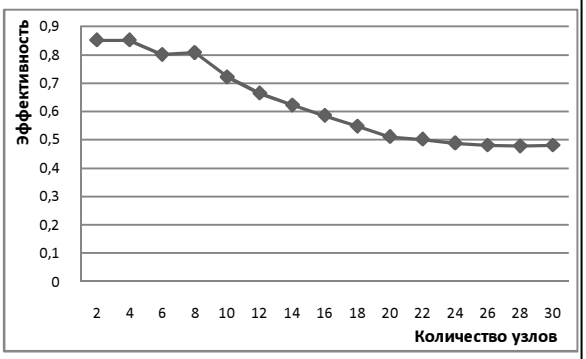


Рис. 5. Зависимость эффективности от количества узлов

тельство приводит к тому, что при увеличении количества узлов параллельной системы возрастает доля обмена сообщениями между узлами по сравнению с вычислениями на узлах. Для других текстовых коллекций, имеющих более равномерное распределение документов по рубрикам, снижение эффективности будет происходить при гораздо большем количестве задействованных узлов.

При исследовании влияния предложенных алгоритмов распределения документов и рубрик на производительность параллельной системы выяснилось, что применение данных алгоритмов увеличивает ускорение в среднем на 80 %, причем 9/10 этого увеличения приходится на алгоритм распределения рубрик, поскольку он используется гораздо чаще во время обучения, чем алгоритм распределения документов.

Таким образом, можно сделать вывод, что предложенная структура и алгоритм обучения параллельной системы автоматической текстовой классификации являются достаточно эффективными и позволяют существенно сократить время обработки больших тестовых коллекций. Так, для коллекций размером порядка 10^5 документов при использовании вычислительного кластера, состоящего из 20–30 обычных персональных компьютеров, время обработки сокращается с десятков часов до десятков минут.

В дальнейшем планируется использовать другие, более крупные коллекции (RCV-1, РОМИП) для более детального исследования производительности системы, а также проанализировать влияние на ускорение параллельных реализаций отдельных блоков системы.

Литература

1. Sebastiani F. Machine Learning in Automated Text Categorization. ACM Computing Surveys. Vol. 34. No. 1. March 2002, pp. 1–47.

2. Vapnik V. Statistical learning theory. Wiley, New York, 1998.

3. Котельников Е.В., Пескишева Т.А., Пестов О.А. Параллельный алгоритм обучения текстового классификатора для многопроцессорной системы с иерархической архитектурой // Вопросы современной науки и практики. Ун-т им. В.И. Вернадского. 2011. № 3 (34). С. 103–110.

4. LIBSVM – A Library for Support Vector Machines. URL: http://www.csie.ntu.edu.tw/~cjlin/libsvm/ (дата обращения: 01.10.2011).

5. Project MPI.NET // The Open Systems Lab, Indiana University. URL: http://www.osl.iu.edu/research/mpi.net (дата обращения: 01.10.2011).

6. Reuters-21578, Distribution 1.0. URL: http://www.daviddlewis.com/resources/testcollections/reuters21578 (дата обращения: 01.10.2011).

УДК 519.6

ПРОГРАММНАЯ РЕАЛИЗАЦИЯ УНИВЕРСАЛЬНОГО МЕТОДА ИССЛЕДОВАНИЯ УСТОЙЧИВОСТИ ДЕФОРМИРУЕМЫХ СИСТЕМ В MAPLE

Е.В. Чусова

(Тверской государственный университет, lioness@pop3.ru)

Представлена программа для исследования устойчивости решения нелинейной системы. Разработанная программа позволяет составить функцию Гамильтона, вычислить якобианы для основной и сопряженной систем, найти точки положений равновесия решения, вывести линейное уравнение второго порядка для сопряженной функции, а также рассчитать выражения кривизны решения.

Ключевые слова: *устойчивость, универсальный метод, программная реализация, блок-схема.*

Теория устойчивости движения и равновесия механических систем устанавливает признаки, позволяющие судить об устойчивости рассматриваемого движения или равновесия.

Актуальной проблемой теории устойчивости является создание строгих и эффективных методов исследования устойчивости движения систем с распределенными параметрами, особенно сплошных сред. С появлением персональных компьютеров и мощных математических программ стало возможным решать сложные задачи за короткое время, в том числе исследовать динамическую устойчивость деформируемых систем.

В настоящей статье представлена программная реализация исследования устойчивости решения нелинейных систем в программе Maple 12, причем для исследования динамической устойчивости деформируемых систем используется новый метод, предложенный в работе [1], без введения функций Ляпунова.

Суть данного метода в следующем. На начальном этапе исходная система дифференциальных уравнений с помощью метода Бубнова–Галеркина приводится к дифференциальному уравнению второго порядка, а затем полученное уравнение сводится к нелинейной системе.

Далее составляется функция Гамильтона, которая представляет собой скалярное произведение вектора сопряженных фазовых координат n-мерного из E^{n^*}-эвклидова сопряженного к E^n пространства на вектор $f(x(t))$ функций правых частей исследуемой системы обыкновенных дифференциальных уравнений и пространства E^n, то есть $H=H(x(t), p(t))=(p(t), f(x(t)))$.

Выписывая сопряженную систему обыкновенных дифференциальных уравнений относительно неизвестных функций $p_i(t)$, i=1, 2, …, n, по формуле $\dot{p}_i(t) = -\dfrac{\partial H}{\partial x_i}$, i=1, 2, …, n, следует учитывать, что данная система всегда линейная однородная с переменными коэффициентами и однозначно сводима к дифференциальному уравнению n-го порядка. Так, для i=1 имеем $a_0(t)p_1^{(n)}(t) + a_1(t)p_1^{(n-1)}(t) + \ldots + a_n(t)p_1(t) = 0$, $a_0(t)\neq 0$, где все коэффициенты непрерывны на области изменения компонент вектора $x(t)$ в n-мерном эвклидовом пространстве E^n.

Далее рассчитываются якобианы основной $\dot{x}_i(t) = \dfrac{\partial H}{\partial p_i}$ и сопряженной $\dot{p}_i(t) = -\dfrac{\partial H}{\partial x_i}$ систем по формулам $J(x(t)) = \begin{vmatrix} \dfrac{\partial f_1(x(t))}{\partial x_1} & \cdots & \dfrac{\partial f_1(x(t))}{\partial x_n} \\ \cdots & \cdots & \cdots \\ \dfrac{\partial f_n(x(t))}{\partial x_1} & \cdots & \dfrac{\partial f_n(x(t))}{\partial x_n} \end{vmatrix}$ и

$$J(p(t)) = \begin{vmatrix} \dfrac{\partial p_1(t)}{\partial p_1} & \cdots & \dfrac{\partial p_1(t)}{\partial p_n} \\ \cdots & \cdots & \cdots \\ \dfrac{\partial p_n(t)}{\partial p_1} & \cdots & \dfrac{\partial p_n(t)}{\partial p_n} \end{vmatrix}.$$

Если якобиан основной системы неотрицательно определен (отсутствуют положения равновесия типа седла), то решение этой системы устойчиво по Ляпунову, при этом решение сопряженной всегда линейной системы в некоторый произвольно взятый момент времени неустойчиво по критерию Гурвица.

В заключение выписываются выражения кривизны решений по формуле Серре–Френе, при этом только для расчета k_x результат вычисления умножается на минус единицу:

$$k_x = -\frac{\begin{vmatrix} \dot{x}_1(t) & \cdots & \dot{x}_n(t) \\ \cdots & \cdots & \cdots \\ x_1^{(n)}(t) & \cdots & x_n^{(n)}(t) \end{vmatrix}}{\left(\sum_{i=1}^{n} \dot{x}_i^2\right)^{3/2}} \text{ и}$$

$$k_p = \frac{\begin{vmatrix} \dot{p}_1(t) & \cdots & \dot{p}_n(t) \\ \cdots & \cdots & \cdots \\ p_1^{(n)}(t) & \cdots & p_n^{(n)}(t) \end{vmatrix}}{\left(\sum_{i=1}^{n} \dot{p}_i^2\right)^{3/2}}.$$

Необходимо учитывать, что значения кривизны решений должны быть противоположны по знаку.

В результате устанавливаются важные фундаментальные факты:

– условиями существования функции Ляпунова являются неотрицательная определенность якобиана, неустойчивость решения линейной сопряженной системы и для нелинейной (а также линейной) основной системы – положительность кривизны интегральной кривой;

– если решение сопряженной линейной системы орбитально устойчиво, такую же устойчивость имеет и решение основной системы; направления движения по орбитам в фазовых пространствах E^n и E^{n^*} противоположны [1].

Данный метод составляет основу для развития специального направления теории и критериев устойчивости нелинейных автономных и неавтономных динамических систем без введения функции Ляпунова. Именно поэтому была составлена программа в Maple 12 на основе универсального метода, которая позволяет выяснить устойчивость решения заданной деформируемой нелинейной системы. Блок-схема программы приведена на рисунке.

Алгоритм программы следующий:

```
with(LinearAlgebra);
px := y;
py := (lambda*(Pi*n)^2-(Pi*n)^4)*x;
Выясним, устойчиво ли решение нелинейной системы;
dx(t)/dt = px;
dy(t)/dt = py;
Составим для этого функцию Гамильтона;
H(p[1], p[2], x, y) = p[1]*px+p[2]*py;
и выпишем сопряженную систему;
pp[1] := -(diff(p[1]*px+p[2]*py, x));
pp[2] := -(diff(p[1]*px+p[2]*py, y));
d*p[1]/dt = pp[1];
d*p[2]/dt = pp[2];
```

```
Якобианы основной и сопряженной систем записываются
соответственно в виде;
J1 := Matrix([[diff(px, x), diff(px, y)], [diff(py,
x), diff(py, y)]]);
Determinant(J1);
J2 := Matrix([[diff(pp[1], p[1]), diff(pp[1],
p[2])], [diff(pp[2], p[1]), diff(pp[2], p[2])]]);
Determinant(J2);
Соответственно для основной и сопряженной системы
они не равны нулю, при;
Determinant(J1) <> 0;
точки положений равновесия решения существуют,
если;
Determinant(J1) > 0;
Преобразуем сопряженную систему в уравнение второго
порядка относительно функции p[1](t);
pp[1] := subs(p[1] = p[1](t), p[2] = p[2](t), -
(diff(p[1]*px+p[2]*py, x)));
pp[2] := subs(p[1] = p[1](t), p[2] = p[2](t), -
(diff(p[1]*px+p[2]*py, y)));
a := d*p[1](t)/dt-pp[1];
d^2*p[1]/dt^2 = subs(diff(p[2](t), t) = pp[2],
diff(p[1](t), t) = pp[1], p[2](t) = solve(a, p[2]),
diff(pp[1], t));
px := subs(x = x(t), y = y(t), px);
py := subs(x = x(t), y = y(t), py);
Выражения кривизны решений выписываются по формуле
Серре-Френе соответственно в виде;
k[x] = -subs(diff(x(t), t) = px, diff(y(t), t) =
py, (px*(diff(py, t))-py*(diff(px, t)))/(px^2+
+py^2)^(3/2));
k[p] = subs(diff(p(t), t) = pp[1], diff(p[2](t), t)
= pp[2], (pp[1]*(diff(pp[2], t))-pp[2](diff(pp[1],
t)))/(pp[1]^2+pp[2]^2)^(3/2))
```

Для наглядности приведем несколько примеров, а также сопоставим результаты с расчетами, полученными без использования данной программы.

Рассмотрим исследование динамической устойчивости центрально сжатого стержня. Дифференциальные уравнения имеют вид $w_{,xxxx}+\lambda w_{,xx}+ +w_{,tt}=0$, $x\in(0;1)$ при граничных условиях $w=w_{,xx}=0$.

В работе [2] данные уравнения были сведены к системе

$$\begin{cases}\dot{x}(t)=y(t),\\ \dot{y}(t)=\ddot{x}=(\lambda\cdot(\pi n)^2-(\pi n)^4)\cdot x.\end{cases}$$

В результате решения поставленной задачи в программе было получено:

– якобианы равны, точки положений равновесия решений существуют при $-(\lambda\cdot(\pi n)^2+(\pi n)^4)>0$;

– линейное уравнение второго порядка для сопряженной функции $p_1(t)$ имеет вид $\ddot{p}_1(t)=p_1(t)(\lambda\cdot(\pi n)^2-(\pi n)^4)$;

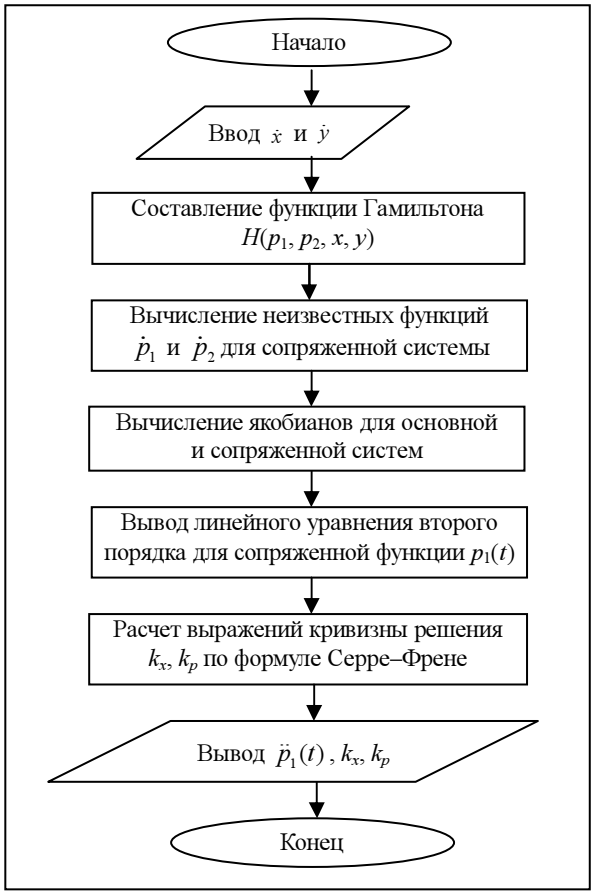
– значения кривизны решения для основной и сопряженной систем противоположны по знаку.

Полученные результаты хорошо согласуются с вычислениями, проведенными в работе [2].

Рассмотрим исследование динамической устойчивости цилиндрической оболочки под действием внешнего давления. Дифференциальные уравнения взяты в следующем виде:

$$D\nabla^4 w-\frac{h}{R}\frac{\partial^2\Phi}{\partial x^2}+Rhp_0t_k\tau-\frac{\partial^2 w}{\partial y^2}+\frac{\rho h}{t_k}\frac{\partial^2 w}{\partial\tau^2}=0,$$

$$\nabla^4\Phi+\frac{E}{R}\frac{\partial^2 w}{\partial x^2}=0,$$



где $\nabla^4=\left(\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2}\right)\cdot\left(\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2}\right)$, при этом внешнее давление изменяется по закону $p=p_0t$, p_0 – скорость нагружения; Ф – функция напряжения; ρ – плотность оболочки; h – толщина стенки оболочки; t_k – критическое время; x – осевая координата; y – координата, отсчитываемая в окружном направлении вдоль срединной поверхности.

В работе [3] данные уравнения были сведены к системе

$$\begin{cases}\dot{\varphi}(\tau)=y(\tau),\\ \dot{y}(\tau)=\ddot{\varphi}(\tau)=\varphi(\tau)\cdot\left(\frac{t_k^3\tau p_0n^2}{\rho hR}-\frac{Dt_k^2}{\rho hR^4}\left(\left(m^2+n^2\right)^2+\frac{Em^4hR^2}{D\left(m^2+n^2\right)^2}\right)\right).\end{cases}$$

Путем исследования поставленной задачи в программе было получено:

– якобианы для основной и сопряженной систем равны, точки положений равновесия решений существуют при

$$\frac{Dt_k^2}{\rho hR^4}\left(\left(m^2+n^2\right)^2+\frac{Em^4hR^2}{D\left(m^2+n^2\right)^2}\right)-\frac{t_k^3\tau p_0n^2}{\rho hR}>0;$$

– линейное уравнение второго порядка для сопряженной функции $p_2(t)$ имеет вид

$$\ddot{p}_2(\tau)-\left(\frac{t_k^3\tau p_0n^2}{\rho hR}-\frac{Dt_k^2}{\rho hR^4}\left(\left(m^2+n^2\right)^2+\frac{Em^4hR^2}{D\left(m^2+n^2\right)^2}\right)\right)\cdot p_2(\tau)=0;$$

– значения кривизны решения для основной и сопряженной систем противоположны по знаку.

Данный вывод полностью согласуется с расчетами, проведенными в работе [3].

Настоящая программа была применена в медицине для исследования процесса взаимодействия клеток опухоли и химиотерапевтического средства, которое задается уравнениями [4]

$$\frac{dx}{dt} = g(x) - \gamma x f(y),\ \gamma - const > 0,\ x(0) = x_0;$$

$$\frac{dy}{dt} = -\alpha y + u,\ \alpha - const > 0,\ y(0) = y_0,$$

где $g(x) = rx - \theta x \ln x\ \left(r, \theta - const > 0\right)$ – функция, описывающая рост числа клеток законом Гомперца.

В результате было получено:

– якобианы равны, точки положений равновесия решений существуют при

$$-\alpha\left(r - \theta \ln x - \theta - \gamma f(y)\right) > 0;$$

– линейное уравнение второго порядка для сопряженной функции $p_2(t)$ имеет вид

$$\ddot{p}_2(t) - \left(r - \theta \ln x - \theta - \gamma f(y) - \alpha\right)\dot{p}_2(t) + \\ + \alpha \cdot \left(-r + \theta \ln x + \theta + \gamma f(y)\right) p_2(t) = 0;$$

– значения кривизны решения для основной и сопряженной систем противоположны по знаку.

Данный вывод полностью согласуется с полученными расчетами, проведенными в работе [4].

Таким образом, предлагаемый метод решения уравнения возмущенного движения может быть применен ко всем задачам динамической устойчивости идеальных пластин и оболочек, уравнения которых сводятся к уравнению второго порядка, а разработанная программа позволяет сократить время на решение подобных задач.

В заключение следует отметить, что разработанная программа позволяет производить расчеты для определения динамической устойчивости деформируемых систем за короткий промежуток времени. В качестве систем дифференциальных уравнений могут использоваться как автономные механические динамические системы, так и неавтономные.

Анализ полученных результатов показывает, что они хорошо согласуются с вычислениями, проведенными ранее вручную.

Литература

1. Кудинов А.Н., Катулев А.Н., Кузнецов А.Ю. Исследование устойчивости автономных нелинейных динамических систем // Динамические и технологические проблемы механики конструкций и сплошных сред: матер. XVI Междунар. симпоз. им. А.Г. Горшкова. М.: МАИ, 2010. С. 110–112.

2. Кудинов А.Н., Чусова Е.В. К исследованию динамической устойчивости деформируемых систем // Синергетика в естественных науках: матер. Междунар. междисциплинар. науч. конф.: Шестые Курдюмовские чтения. Тверь: ТвГУ, 2010.

3. Кудинов А.Н., Чусова Е.В. Исследование динамической устойчивости цилиндрических оболочек // Динамические и технологические проблемы механики конструкций и сплошных сред: матер. XVII Междунар. симпозиума им. А.Г. Горшкова. М.: 2011. Т. 2. С. 121–134.

4. Кудинов А.Н., Чусова Е.В. Исследование устойчивости процесса взаимодействия клеток опухоли и химиотерапевтического средства // Синергетика в естественных науках: матер. Междунар. междисциплинар. науч. конф.: Седьмые Курдюмовские чтения. Тверь: ТвГУ, 2011. С. 227–229.

УДК 004.056

МЕТОДИКА ОЦЕНКИ РЕАЛЬНОГО УРОВНЯ ЗАЩИЩЕННОСТИ АВТОМАТИЗИРОВАННЫХ СИСТЕМ

В.А. Мукминов, к.т.н.; В.М. Хуцишвили, к.т.н.; А.В. Лобузько
(4 ЦНИИ Минобороны России, г. Тверь, mva131@mail.ru, xbm@mail.ru)

Рассмотрена методика оценки реального уровня защищенности АСУ в условиях моделирования компьютерных атак. Приведен перечень инструментальных средств, используемых для проведения оценки, включая средства моделирования компьютерных атак. Предложено комплексное использование новых форм испытаний и тестирования программных средств АСУ на основе натурного и имитационного моделирования. Применение новой методики оценки реального уровня защищенности позволит повысить эффективность работы служб информационной безопасности при эксплуатации АСУ.

Ключевые слова: *АСУ, защищенность, уязвимости, оценка уровня защищенности, средства защиты информации, средства обнаружения компьютерных атак.*

Многие коммерческие и государственные организации, заменив практически всю свою техническую и программную инфраструктуру, вынуждены периодически проводить контроль (технический аудит) созданных и эксплуатируемых АСУ.

С этой целью привлекаются специалисты (эксперты) по информационной безопасности, работа которых заключается в определении уязвимостей программно-технических средств АСУ и разработке рекомендаций по защите информации. По-

сле получения экспертного заключения по защите информации на объекте информатизации зачастую устанавливаются обновления для *операционных систем* (ОС), блокируются незадействованные порты и устройства ввода/вывода (USB, FLOPPY, DVD и т.д.), устанавливаются межсетевые экраны в сегменты вычислительной сети, ужесточается регламент служебного времени (контрольно-пропускной режим) и т.п.

Несмотря на то, что методы и средства защиты информации постоянно совершенствуются, разработка методики оценки реального уровня защищенности и создание инструментальных средств, при помощи которых проводятся подобные проверки, по-прежнему актуальны и востребованы.

Существующие методики оценки, в основе которых заложен вероятностный подход либо подход к проверке соответствия требований по защищенности, заданных на этапе технического задания (уточненных на этапе технического проекта), не учитывают реальное состояние защищенности, а лишь дают приближенную оценку, основанную на данных, полученных экспертным путем.

Программные комплексы (ПК), созданные для автоматизации процесса оценки защищенности, не могут рассчитать реальный уровень защищенности и дать адекватную оценку. Такие ПК, как «АванГард» [1] и аналогичные ему, направлены на оценку рисков нарушения информационной безопасности, но при проведении экспертной оценки не учитывают техническую структуру объекта информатизации и особенности АСУ.

Такое положение в технологиях и методиках оценки защищенности информации сложилось из-за отсутствия системного подхода в методологии анализа и синтеза *средств защиты информации* (СЗИ), сложности объективного подтверждения эффективности СЗИ и неполноты нормативно-методического обеспечения информационной безопасности прежде всего в области систем показателей и критериев.

Инструментальные средства для проведения оценки

Оценку защищенности АСУ предлагается проводить по направлениям локального и сетевого тестирования АСУ [2].

Локальное тестирование заключается в проверке состояния защищенности объекта информатизации от внутреннего нарушителя. С этой целью используются следующие инструментальные средства:

– сканеры портов Nmap, SuperScan и др. (определяют количество и наименования открытых портов);

– средства анализа защищенности XSpider, Сканер-ВС и др. (определяют перечень уязвимых мест и выдают рекомендации по их блокированию).

Сетевое тестирование заключается в моделировании действий внешнего нарушителя и направлено на проверку состояния защищенности объекта информатизации в агрессивных условиях. Для этого используются следующие инструментальные средства:

– снифферы WireShark, Ettercap и др. (проводят пассивное и/или активное прослушивание сетевого трафика);

– средства моделирования компьютерных атак MetaSploit Framework, Nessus и др. (проводят моделирование компьютерных атак на различные ОС и ПО).

Методика оценки защищенности АСУ в условиях компьютерных атак

Исходными данными методики являются: программно-технические средства АСУ (рабочие станции, серверы), объединенные в *локальную вычислительную сеть* (ЛВС); коммуникационное оборудование (маршрутизаторы, коммутаторы ЛВС, концентраторы и т.п.); средства защиты информации (межсетевые экраны, антивирусные средства, средства защиты информации и др.).

К выходным показателям методики относятся количество выявленных IP-адресов и открытых портов на них, а также уязвимых мест и реализованных через них компьютерных атак.

Основные проводимые операции (действия) представлены в таблице.

Алгоритм методики следующий.

1. Выполнение начинается с подключения к вычислительной сети и проведения пассивного прослушивания сетевого трафика, в результате чего определяются IP-адреса компьютеров, находящихся в сети. Большинство современных снифферов, анализируя сетевые пакеты, сразу определяют ОС активных компьютеров. Наименование ОС является основным параметром, который в качестве исходных данных используется при выполнении следующих этапов.

2. Полученные при прослушивании сети IP-адреса компьютеров задаются в виде диапазона, по которому проводится сканирование сети. В результате сканирования формируется уточненный перечень компьютеров и их IP-адресов с указанием количества открытых портов P на них.

3. Следующим действием является анализ уязвимостей. Для этого при помощи соответствующих программных средств задается уточненный перечень IP-адресов компьютеров для сканирования. В результате сканирования формируется список уязвимостей и уязвимых портов Y, на которых функционируют службы и сервисы ОС тестируемых компьютеров.

Таким образом, путем локального тестирования можно определить коэффициент уязвимости компьютеров вычислительной сети. Коэффициент уязвимости L локального тестирования равен от-

Действие	Ожидаемые результаты	Используемые программные средства		Входные данные	Выходные данные
		Windows	Linux		
Прослушивание вычислительной сети средств информатизации	Анализ информационного трафика в сети, выявление диапазона IP-адресов, определение возможной топологии построения сети, ее схемы адресации, определение протоколов, используемых для передачи информации	WireShark	Ettercap	Сетевой трафик	IP-адреса и ОС компьютеров сети
Сканирование сети средств информатизации (конкретного диапазона IP-адресов)	Выявление существующих подключений к сети, открытых портов, версий ОС на сканируемых компьютерах	SuperScan	Nmap	Диапазон IP-адресов выявленных компьютеров	Открытые порты, сетевые службы и сервисы, функционирующие на них, версии ОС
Сканирование на наличие уязвимостей	Сканирование обнаруженных в составе сети компьютеров на наличие уязвимых мест и получение предварительных рекомендаций по их блокированию	XSpider	Сканер-ВС	Конкретные IP-адреса выявленных компьютеров	Уязвимые места, находящиеся на открытых портах
Моделирование компьютерных атак (на конкретные версии ОС)	Моделирование компьютерных атак через выявленные уязвимые места ОС и ПО	MetaSploit Framework	Nessus	IP-адреса компьютеров, версия ОС, ПО	Количество реализованных компьютерных атак

ношению количества открытых портов к количеству портов, на которых обнаружена уязвимость, и находится по формуле

$$L = \sum_{i=1}^{n} \frac{Y_i}{P_i}, \quad (1)$$

где P_i – количество открытых портов на i-м компьютере; Y_i – количество портов i-х компьютеров, на которых были найдены уязвимости.

4. Этап выполнения методики, содержащий активные действия, заключается в моделировании компьютерных атак через уязвимости протоколов передачи данных, ОС, ПО. Программные средства моделирования компьютерных атак включают в свой состав основные (E) и дополнительные (A) сценарии компьютерных атак. Количество возможных компьютерных атак может быть ограничено на основе выявления ОС, установленных на тестируемых компьютерах.

По результатам сетевого тестирования определяется коэффициент уязвимости S вычислительной сети, который равен отношению количества реализованных компьютерных атак к общему количеству возможных (предполагаемых) компьютерных атак и находится по формуле

$$S = \sum_{i=1}^{n} \frac{R_i}{E_O + A_O}, \quad (2)$$

где R_i – количество реализованных компьютерных атак на i-м компьютере; E_O – количество основных сценариев моделирования компьютерных атак (Exploits), ограничивается наименованием ОС (O); A_O – количество дополнительных сценариев моделирования компьютерных атак (Auxiliaries), ограничивается наименованием ОС (O).

Коэффициенты уязвимости, полученные по результатам локального и сетевого тестирования, составляют коэффициент общей уязвимости системы W, который находится по формуле

$$W=L\cdot S. \quad (3)$$

Коэффициент защищенности системы Z_1 можно найти, исходя из того, что уровень защиты системы и уровень общей уязвимости системы дополняют друг друга до единицы:

$$Z_1=1-W. \quad (4)$$

5. На заключительном этапе методики осуществляются сбор данных о количестве обнаруженных компьютерных атак и расчет коэффициента реального уровня защищенности АСУ. Сбор данных может быть реализован двумя основными способами:

– при наличии на объекте информатизации средств обнаружения компьютерных атак данные о количестве компьютерных атак находятся на основе анализа журнала регистрации событий таких средств;

– если на объекте информатизации установлены штатные СЗИ, вывод о количестве компьютерных атак делается путем обхода СЗИ и суммирования фактов обнаружения несанкционированных действий на них.

Коэффициент эффективности СЗИ определяется по формуле

$$Z_2 = \frac{K_{\text{обн}}}{R}, \quad (5)$$

где $K_{обн}$ – количество компьютерных атак, обнаруженных всеми СЗИ; R – общее количество реализованных компьютерных атак.

Оценка реального уровня защищенности АСУ может быть получена на основе расчета обобщенного коэффициента, который находится как произведение коэффициента защищенности системы и коэффициента эффективности СЗИ и выражается формулой

$$Z=Z_1 \cdot Z_2. \quad (6)$$

Обобщая, можно сделать следующие выводы. Авторами разработана методика оценки реального уровня защищенности АСУ в условиях моделирования компьютерных атак. Сложность и территориальная распределенность элементов АСУ, повышение требований к обеспечению их защищенности в условиях компьютерных атак приводят к необходимости поиска новых форм испытаний и тестирования программных средств АСУ, к которым в первую очередь относится натурное и имитационное моделирование компьютерных атак. Приведен перечень инструментальных средств, используемых для проведения оценки, включая средства моделирования компьютерных атак.

Достоверность и обоснованность полученных результатов обеспечиваются полнотой учета исходных данных о сценариях компьютерных атак, уязвимостях АСУ, полученных из практики; полнотой учета факторов, влияющих на защищенность АСУ в условиях компьютерных атак; созданием реальной обстановки (внезапность проведения испытаний, что дает чистоту эксперимента) и наличием штатно функционирующих СЗИ, а также выбором показателей оценки эффективности средств защиты информации, применяемых в разработанных методах и алгоритмах тестирования.

Литература

1. Бурдин О.А., Гордеев Ю.А., Кононов А.А. Оценка рисков нарушения информационной безопасности с помощью комплексной экспертной системы «АванГард» // Системные проблемы качества математического моделирования, информационных, электронных и лазерных технологий: матер. Междунар. конф. и российск. науч. шк. М.: МГИЭМ, 2001. Ч. 5. С. 134–139.

2. Войнов Ю.В., Мукминов В.А. Об архитектуре средств тестирования автоматизированных систем в условиях моделирования информационных воздействий // Изв. Инс-та инженер. физики. 2011. № 1. С. 36–39; Методика проведения технических экспертиз и испытаний автоматизированных систем в условиях моделирования информационных воздействий. С. 8–12.

УДК 004

КОНТРОЛЬ ЦЕЛОСТНОСТИ ВХОДНЫХ ДАННЫХ ПРИ ПРОВЕДЕНИИ АВТОМАТИЗИРОВАННОГО АНАЛИЗА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

(Работа выполнена при финансовой поддержке РФФИ, грант № 11-07-13150)

М.А. Поляничко (Петербургский государственный университет путей сообщения, г. Санкт-Петербург, mark.polyanichko@gmail.com)

Рассматривается возможность введения этапа контроля целостности данных при проведении автоматизированного исследования ПО на наличие уязвимостей и ошибок. Приводится оценка временных затрат при различных объемах входных данных.

Ключевые слова: *автоматизированный анализ, ПО, тестирование, ошибка, уязвимость, фаззинг, верификация.*

Непрерывное развитие информационных технологий и их ПО требует соответствующего развития методов обеспечения качества и безопасности. Исследование ПО на наличие ошибок и уязвимостей – трудоемкий процесс, проводимый на всех этапах жизненного цикла продукта, поэтому важным направлением представляется автоматизация анализа ПО с его последующей верификацией.

Одним из современных решений автоматизации анализа ПО на наличие уязвимостей и ошибок является фаззинг – набирающий популярность подход к исследованию ПО. Он подразумевает передачу в программу заведомо ложных данных или искаженных правильных входных данных [1]. В первом случае данные генерируются случайным образом, во втором тестовые данные получаются путем внесения изменений в известные правильные данные. Второй подход называется интеллектуальным фаззингом и является приоритетным в развитии автоматизированного анализа. Для повышения эффективности процесса анализа программы предпочтительно проводить фаззинг на основе примера правильных входных данных или с учетом спецификации программы. Под правильными входными данными стоит подразумевать данные, при которых программа не перешла в состояние ошибки и успешно завершилась. Поиск набора данных (или последовательности наборов), при выполнении которого программа перейдет в

состояние ошибки, происходит в процессе многократного выполнения программы или ее отдельных частей с последовательным искажением первичных входных данных [2]. Фаззер представляет собой программное средство, которое последовательно вносит изменения в предоставленный ему массив данных по определенному, часто неизвестному пользователю принципу, передает данные в исследуемую программу и запускает ее. В некоторых случаях, если исследуемый программный комплекс имеет необычную реализацию интерфейсов, целесообразна разработка индивидуального средства фаззинга. В реализации фаззера требуется выполнение операций по перемещению блоков данных случайной длины, следовательно, нельзя утверждать, что в самом фаззере нет ошибок.

Для повышения достоверности получаемых результатов следует обеспечить дополнительный контроль передаваемых в программу данных, так как в случае нарушения состава данных, взятых в качестве начальных, результаты исследования нельзя считать достоверными. На рисунке 1 изображена схема анализа ПО с учетом введения дополнительной фазы – контроля целостности используемых данных. Очевидно, используемые данные не будут выполнены, пока не пройдут проверку целостности.

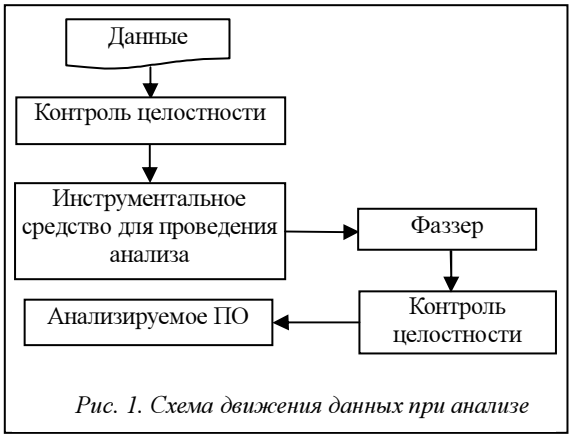


Рис. 1. Схема движения данных при анализе

Следует различать следующие способы передачи входных данных в программу: командная строка, переменные среды окружения, файлы.

Наибольшая вероятность появления ошибок при искажении множества данных во время работы фаззера возможна при передаче в программу файла из-за его большого объема. В случае изменения состава передаваемых данных, а также успешного прохождения исследования полученные результаты окажутся недостоверными. Помимо этого, все последующие итерации анализа будут основываться на ошибочных данных, что может потенциально привести к появлению последующих ложноотрицательных результатов и, как следствие, к ошибочным выводам о надежности, качестве или безопасности программы.

Для реализации этапа контроля целостности требуется разработать алгоритм действий над множеством данных, выполняемый непосредственно перед их передачей в анализируемую программу (рис. 2).

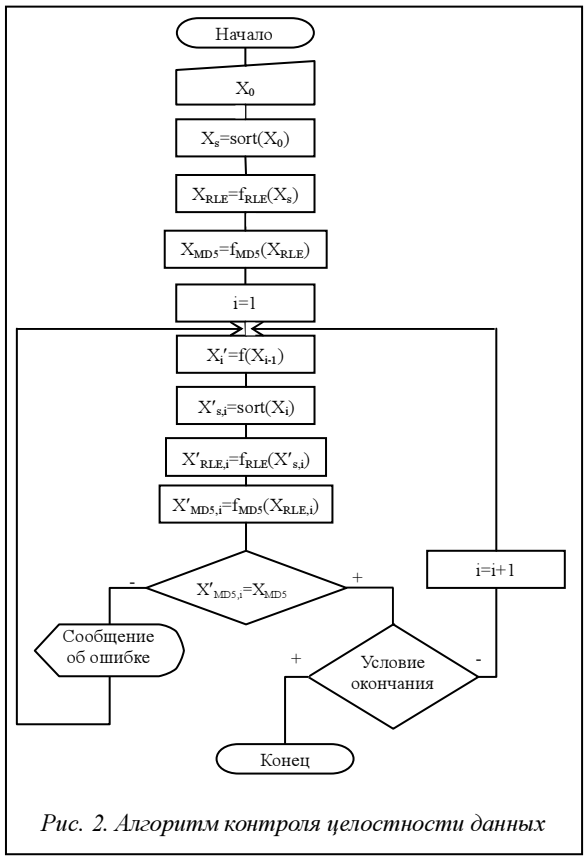


Рис. 2. Алгоритм контроля целостности данных

Пусть $X=\{x_1, x_2, ..., x_n\}$ – начальный массив входных данных, выполнение которого привело к корректному завершению программы. Каждый x_i из множества X принадлежит множеству $\Sigma=\{\sigma_1, \sigma_2, ..., \sigma_m\}$ – множеству символов алфавита начального множества.

Пусть $X'=\{x'_1, x'_2, ..., x'_n\}$ – массив искаженных данных, полученных путем перестановки элементов начального множества X случайным образом. Соответственно все $x'_i \in \Sigma$. $X'=f(X)$, где f() – функция преобразования данных, выполняемая фаззером.

В данном алгоритме для обеспечения целостности предлагается сравнивать не исходные массивы данных, а значения их хэш-функций. Для получения дайджестов, чувствительных к изменению состава множества, необходимо при каждой проверке приводить данные к виду, однозначно отражающему алфавит множества и количество повторений каждого из символов в передаваемом в программу массиве данных.

Следовательно, перед применением хэш-функции необходимо выполнить сортировку массива. Для сортировки больших массивов данных предлагается использовать алгоритм сортировки Хоара

Quicksort [3]. Отсортированный массив входных данных обозначим X_s=sort(X), где sort() – реализованная функция сортировки. Если при искажении массива входных данных изменился только порядок элементов массива, то sort(X)=sort(X').

Для исключения необходимости работы с полным объемом данных к отсортированному массиву X_s применяется алгоритм кодирования длин серий RLE (Run-length encoding). Полученные данные представляют собой последовательность X_{RLE}, содержащую сам повторяющийся символ и количество его повторов. Алгоритм наиболее эффективен при работе с файлами изображений и другими форматами данных, в которых присутствуют большие последовательности одинаковых элементов. В зависимости от особенности обрабатываемых данных возможно использование другого алгоритма.

Если при работе с данными не были допущены ошибки, то последовательность X_{RLE} остается неизменной на каждом шаге анализа программы. Длина последовательности $X_{RLE} \geq 2m$ и равна $2m$ в случае, если каждый символ повторяется только один раз.

В большинстве случаев для наглядности и простоты обнаружения ошибки удобно использовать значение хэш-функции MD5 X_{MD5}, полученной от X_{RLE}, длиной в 32 символа, так как изменение хотя бы одного символа в хэшируемой последовательности приведет к изменению всего дайджеста из-за лавинного эффекта хэш-функции [4].

При неравенстве полученного дайджеста эталонному, полученному от начального массива, можно считать, что при искажении данных произошла ошибка, и необходимо повторить последний шаг проверки с заново обработанными данными.

По причине роста функциональной насыщенности современных программ и, соответственно, объема исполняемого кода одной из основных целей автоматизации процессов анализа ПО является сокращение временных затрат. Однако внедрение такого этапа может противоречить данной цели. Моделирование работы алгоритма показало следующие затраты времени на шаг работы алгоритма в зависимости от длины обрабатываемого массива данных:

Длина массива n (символов)	Затраченное время (сек.)
10	0,373
100	0,377
1 000	0,410
10 000	0,601
100 000	3,211
1 000 000	34,611
10 000 000 (≈10 Mb)	370,432

Экспоненциальный характер роста временных затрат обусловлен особенностью реализации алгоритма сжатия RLE класса сложности P [5], при использовании более эффективных реализаций временные затраты будут заметно снижены. Также необходимо отметить, что использование приведенных алгоритмов хэширования, сжатия и сортировки данных не является обязательным и может варьироваться в зависимости от решаемой задачи. Таким образом, с учетом специфики обрабатываемых данных этап контроля целостности можно оптимизировать для получения приемлемых в рамках поставленной задачи временных затрат.

Введение проверки такого рода в процесс поиска ошибок и уязвимостей может существенно повысить достоверность полученных в ходе исследования результатов. При работе с входными данными большого объема, к которым относятся аудио, видео и другие мультимедийные файлы, контроль целостности может потребовать заметных временных затрат. Если программа работает с файлами размером от 1 Kb до 3 Mb, временные затраты не могут считаться существенными.

Кроме того, контроль целостности можно использовать только при обнаружении ошибки, когда требуется более тщательная проверка, а также при необходимости контролируемого повторения условий возникновения ошибки или для систематизации результатов. При использовании фаззеров сторонних разработчиков в условиях отсутствия достоверных данных о методах искажения данных и корректности их реализации дополнительный этап контроля целостности данных позволяет снизить зависимость от качества используемых инструментальных средств.

Таким образом, дополнительный этап контроля целостности при проведении автоматизированного тестирования является гибким, эффективным и одновременно простым и доступным способом повышения достоверности данных, получаемых при анализе.

Литература

1. Саттон М., Грин А., Амини П. Fuzzing. Исследование уязвимостей методом грубой силы. М.: Символ, 2009.

2. The evolving art of fuzzing, Jared DeMott, 2006.

3. Алгоритмы: построение и анализ / Т. Кормен [и др.]; [под ред. И.В. Красикова]. Изд. 2-е. М.: Издат. дом «Вильямс», 2005. С. 198–219.

4. Réjane Forré. The Strict Avalanche Criterion: Spectral Properties of Boolean Functions and an Extended Definition. Proceedings on Advances in cryptology, Springer-Verlag NY, Inc, 1990.

5. Разборов А.А. О сложности вычислений. М.: МЦНМО, 1999. № 3. С. 127–141.

УДК 004.67

КЛАСТЕРИЗАЦИЯ АТОМАРНЫХ ЗНАНИЙ С ИСПОЛЬЗОВАНИЕМ МЕТОДОВ ЛИНЕЙНОЙ ОПТИМИЗАЦИИ

А.Х. Галеев
(Самарский государственный архитектурно-строительный университет, galeev@netcracker.com)

Рассматриваются методы, позволяющие формировать плотные кластеры фактологических знаний из неструктурированной информации, хранящейся в Интернете. Предложены собственные математические модели для решения задачи с использованием методов целочисленного линейного программирования.

Ключевые слова: *кластеризация, знание, информация, линейное программирование.*

Современная мощная информационная техника (компьютеры и сети), а также специальное ПО вполне могут справиться с огромным количеством информации. Однако большая часть этого потока – неструктурированная текстовая информация, в связи с чем возникает проблема ее структуризации.

Важнейшим источником знаний в настоящее время является Интернет, где большая часть объема информации представляет собой информационный шум, поэтому вопрос о структуризации хранящихся в сети знаний относится к числу наиболее актуальных.

Одной из возможных форм представления знаний является атомарная. Знания в этом случае представляются в виде элементарных элементов – атомов знаний.

Атом знания – элементарная семантическая единица, достоверный факт, его конкретное значение, связанное с неограниченным количеством понятий, являющихся метаинформацией для данного значения. В качестве примера приведем следующие атомы.

«Вода» – «Температура кипения» – «100 градусов». В этом примере «100 градусов» – значение (факт), а «Вода» и «Температура» – понятия (метаинформация).

«Вода» – «Температура замерзания» – «0 градусов». В этом примере «0 градусов» – значение, а «Вода», «Температура» и «замерзания» – связанные со значением понятия.

Набор атомов знаний в простейшем случае можно представить в виде двухмерной таблицы, в строках и столбцах которой располагаются понятия, а в ячейках – связанные с соответствующими понятиями значения.

На основе такой таблицы можно получить матрицу знаний, строки которой содержат объекты, столбцы – их характеристики. Ноль на пересечении строки/столбца показывает отсутствие информации, характеризующей соответствующий объект, единица – ее наличие.

Знания, представленные в данной форме, легко поддаются различным методам обработки, в том числе кластеризации.

Кластер-анализ – это способ группировки многомерных объектов, основанный на представлении результатов отдельных наблюдений точками подходящего геометрического пространства с последующим выделением групп как сгустков этих точек (кластер – от англ. cluster – сгусток, гроздь, скопление).

Введем определения.

Полностью заполненный кластер знаний (ПЗКЗ) – подтаблица исходной матрицы знаний, на пересечении любых строк/столбцов которой находятся только единицы, то есть существует информация, характеризующая любые пары объектов/характеристик, входящих в кластер.

Частично заполненный кластер (ЧЗКЗ) – подтаблица исходной матрицы знаний, на пересечении хотя бы одной строки/столбца которой находится ноль, то есть отсутствует информация, характеризующая зависимость соответствующего объекта/характеристики.

К предложенному виду матрицы знаний применимы следующие варианты кластеризации:

– выделение из заданной таблицы ПЗКЗ максимального размера;

– выделение из заданной таблицы всех ПЗКЗ, размер которых не меньше заданного;

– выделение из таблицы ЧЗКЗ с плотностью, не меньше заданной;

– выделение ЧЗКЗ максимальной плотности при заданном размере кластера.

Применительно к атомам знаний целью кластеризации является выявление сгустков семантически близких атомов знаний на основе общих метаданных, что дает возможность получать набор конкретных фактов в ответ на поисковые запросы пользователя. В то же время выявление частично заполненных кластеров позволит определять фактографические области с пробелами в знаниях, на базе чего может быть рассчитано последующее направление самообучения автоматизированной системы, хранящей такие факты.

Модели кластеризации табличных знаний

Выделение ПЗКЗ наибольшего размера. Для реализации данной модели предложена следующая математическая модель, использующая методы линейного программирования.

Исходные данные/переменные: a_{ij} – клетки исходной матрицы знаний; x_i – признак включения объекта в кластер; y_j – признак включения характеристики в кластер; q_{ij} – признак включения клетки в кластер.

Ограничения: $q_{ij} \geq x_i + y_j - 1$ – клетка должна быть включена в кластер, если включены соответствующие объект и характеристика; $q_{ij} \leq \frac{x_i + y_j}{2}$ – для включенной клетки необходимо, чтобы соответствующие объект и характеристика были включены в кластер; $q_{ij} \leq a_{ij}$ – можно включить только такую клетку, в которой в исходной матрице стоит оптимизируемая функция на максимум числа клеток, включаемых в кластер (размер кластера): $\sum_i \sum_j q_{ij} \rightarrow max$.

Выделение ЧЗКЗ с наибольшим процентом заполненности при заданном размере кластера. Для реализации данной модели предлагается следующая математическая модель, использующая методы линейного программирования.

Исходные данные/переменные: a_{ij} – клетки исходной матрицы знаний; x_i – признак включения объекта в кластер; y_j – признак включения характеристики в кластер; q_{ij} – признак включения клетки в кластер; n – число строк выделяемого кластера; m – число столбцов выделяемого кластера.

Ограничения: $q_{ij} \geq x_i + y_j - 1$ – клетка должна быть включена в кластер, если включены соответствующие объект и характеристика; $q_{ij} \leq \frac{x_i + y_j}{2}$ – для включенной клетки необходимо, чтобы соответствующие объект и характеристика были включены в кластер; $\sum X_i = N$ – число строк выделяемого кластера должно быть равно заданному; $\sum Y_i = M$ – число столбцов выделяемого кластера должно быть равно заданному.

Оптимизируемая функция на максимум числа единиц, включаемых в кластер: $\sum a_{ij} q_{ij} \rightarrow max$.

Реализация моделей

Предложенные модели успешно реализованы на языке C# с использованием библиотеки целочисленного линейного программирования lpsolve55.

Для рассмотрения применимости вышеописанного подхода обработки информации и корректности математических моделей были проведены исследования на имеющей практическую значимость задаче.

Для этого выбрана область «Информационные технологии в образовании». В Интернете найдены таблицы, в той или иной мере относящиеся к данной области (табл. 1–2).

В таблице 1 содержатся сведения о количестве поступивших на обучение в 2000 и 2010 году в учебные учреждения различных ступеней образования, в таблице 2 показано состояние процесса

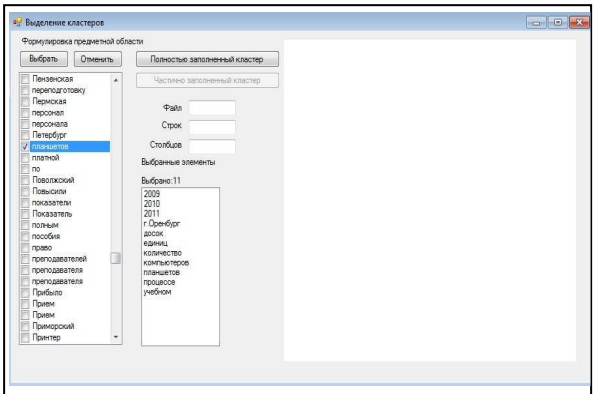


Рис. 1. Первый шаг эксперимента

Полученный кластер

Число строк кластера:3
Число столбцов кластера:3
Время кластеризации:1,8624(с)

	компьютеров	планшетов	досок
2009	2828	211	211
2010	3356	334	334
2011	4110	493	493

Процент заполненности:100

Детализация кластера:

Атомы:

2828
г.Оренбург
Количество компьютеров, единиц
01.01.2009 г.
Проект:Компьютеризация системы образования г.Оренбурга в 2007-2011 гг.

Рис. 2. Отчет по кластеризации на первом шаге

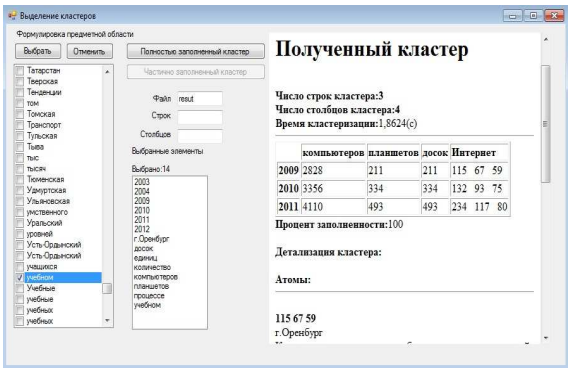


Рис. 3. Третий шаг эксперимента

интернетизации системы образования г. Оренбурга в 2007–2011 гг.

Таблица 1

Оборудование, задействованное в учебном процессе	**Количество по годам**				
	2007	**2008**	**2009**	**2010**	**2011**
Компьютеры	1423	2118	2828	3356	4110
Мультимедиа-проекторы	98	141	478	730	946
Интерактивные доски, приставки, планшеты	53	85	211	334	493

Таблица 2

Интернетизация	Количество по годам				
	2007	2008	2009	2010	2011
Муниципальные образовательные учреждения, подключенные к сети (шт.), в том числе:	86	93	115	124	234
школы	77	81	86	86	86
детские сады	0	2	15	20	130
учреждения дополнительного образования	9	10	14	18	18
Web-сайты образовательных учреждений в сети (ед.)	32	56	67	93	117
Педагоги школ, владеющие информационно-коммуникационными технологиями (%)	23	47	59	75	80

Таблицы были разбиты на атомы знаний, интегрирующие в себе имя источника, метаданные на основе строк/столбцов таблицы и непосредственно сами значения ячеек. Таким образом были подготовлены исходные данные для исследования.

Далее заполненная БД была подключена к модулю выделения сгустков знаний. Проведено исследование получаемых из таких данных кластеров знаний. В исходном тезаурусе содержалось большое количество терминов, поэтому на первом этапе были выбраны термины, точно описывающие некоторые значения исходных таблиц (рис. 1): «г. Оренбург», «количество», «компьютеров», «планшетов», «досок», «единиц», «учебном», «процессе», «2009», «2010», «2011».

Был получен кластер знаний, соответствующий данным исходной таблицы 1 до дезинтеграции (рис. 2).

Далее к исходному набору данных добавлены термины «web-сайтов» и «Интернет».

В итоге был получен кластер, объединивший в себе атомы из исходных таблиц 1 и 2, большего размера, чем на первом шаге.

Далее были добавлены термины «2004» и «2003».

В результате исходный отчет остался прежним (рис. 3), так как в системе отсутствовали атомы, связанные с добавленными терминами, информационный шум успешно отсеялся.

Автор выражает глубокую благодарность профессору Пиявскому С.А. за помощь в работе над материалами статьи.

Литература

1. Ландэ Д.В. Поиск знаний в Internet. М.: Диалектика, 2005. 265 с.

2. Информатизация образования. URL: <http://www.orenschool.ru/index.php?p=content&id=72&area=1> (дата обращения: 10.09.2010).

УДК 004.93'1/852+519.226

СЕМАНТИЧЕСКАЯ СЕГМЕНТАЦИЯ ДАННЫХ ЛАЗЕРНОГО СКАНИРОВАНИЯ

(Работа выполнена при поддержке ФЦП «Научные и научно-педагогические кадры инновационной России на 2009–2013 гг.», контракт № П1189)

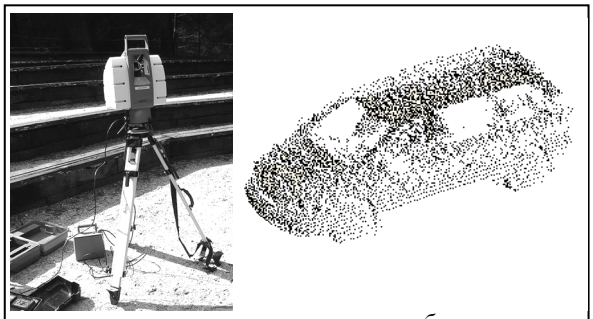
Р.В. Шаповалов; А.Б. Велижев; О.В. Баринова; А.С. Конушин

(Московский государственный университет им. М.В. Ломоносова, (shapovalov, avelizhev, obarinova, ktosh) @graphics.cs.msu.su)

Работа посвящена классификации облаков точек, полученных при лазерной съемке естественных сцен. В настоящее время эта задача успешнее всего решается с помощью ассоциативных марковских сетей. В данной работе используются марковские сети общего вида, что позволяет повысить точность классификации. Показано, как эффективно осуществлять вывод и настраивать параметры модели. Пересегментация используется для сокращения размерности данных, что приводит к ускорению работы алгоритма и упрощению структуры облака.

Ключевые слова: *компьютерное зрение, распознавание, семантическая сегментация, ЛИДАР, лазерное сканирование, марковская сеть.*

Технологии лазерного сканирования занимают все более прочные позиции. Так, с помощью прибора ЛИДАР (LIDAR – light detection and ranging) можно получить трехмерные облака точек, приближающие поверхности реальных сцен (лазерные сканы) (рис. 1а). Данный прибор стал альтернативой или дополнением к фотографическим сенсорам в мобильных системах картографирования и наблюдения, построенных на основе автомобильных транспортных средств или летательных аппаратов, в подвижных автономных роботах и даже в бытовой технике. Лазерный скан представляет собой неупорядоченное множество точек трехмерного пространства без дополнительной



а *б*

Рис. 1. Стационарный ЛИДАР (а) и пример лазерного скана автомобиля (б)

информации о цвете и других характеристиках сканируемой поверхности (рис. 1б). В отличие от фотографий сканы сохраняют масштаб сцены и инвариантны к погодным условиям, что объясняет распространение технологии. Поэтому задача анализа данных лазерного сканирования очень актуальна. Один из методов анализа – семантическая сегментация сканов. В данной постановке каждой точке скана необходимо сопоставить метку класса из заранее определенного набора. Для аэросъемки это могут быть классы зданий, автомобилей, деревьев, земли (рис. 2). Обзор методов семантической сегментации облаков точек можно найти, например, в [1].

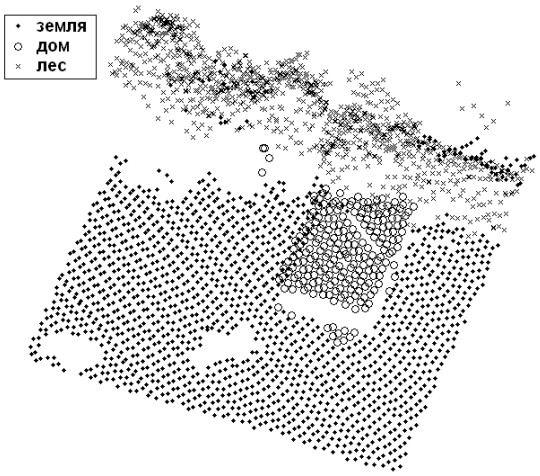


Рис. 2. Результат работы метода на части тестового скана (классы земли, зданий и растительности показаны маркерами разной формы)

Интуитивно анализ трехмерных данных выглядит более простой задачей по сравнению с анализом изображений, так как последний фактически является обратной задачей: при получении фотографии значительная часть информации о пространственной структуре сцены теряется.

Высокий потенциал трехмерного представления данных ранее отмечал один из пионеров компьютерного зрения Дэвид Марр, который считал, что даже распознавание объектов на двухмерных изображениях должно выполняться посредством восстановления поверхностей в трехмерном пространстве, идентифицирующих эти объекты. Однако на практике ситуация с лазерными сканами не столь оптимистична: часто они более похожи на карты глубины, чем на аппроксимацию трехмерной формы объекта. Даже если снимать объект с нескольких ракурсов (разработаны методы эффективной регистрации облаков точек из локальных систем координат в общую), он может оказаться загороженным другими объектами или самим собой в случае его невыпуклости, поэтому скан может содержать неполную поверхность объекта. Данные лазерного сканирования часто зашумлены и разрежены. Они не обладают привычной цветовой информацией. К тому же обработка облаков точек человеком с помощью существующих технических средств затруднена, так как они ориентированы прежде всего на вывод и ввод двухмерных данных.

Рассмотрим сканы естественных сцен (в противоположность сканам одного объекта). В данной работе оценивается качество предложенного метода на данных аэросъемки, полученных посредством мобильного ЛИДАРа, установленного на самолете.

Марковские случайные поля

Для семантической сегментации лазерных сканов и изображений часто используются марковские случайные поля, или марковские сети. В контексте распознавания облаков точек в марковскую сеть объединяются точки облака, а значением случайной величины в узле сети является метка класса, соответствующая точке [2]. Интерес представляет способ задания потенциалов марковской сети, с помощью которых формулируется минимизируемая функция энергии.

Для решения задачи классификации точек вводится следующая марковская сеть. Имеем набор дискретных случайных величин $y=\{y_1, \ldots, y_n\}$, каждая из которых соответствует одной точке скана и принимает значения из $y_i \in \{1, \ldots, k\}$, представляющие метки классов, назначаемые точкам. Зависимость между случайными величинами задается неориентированным графом, в котором вершины соединяются ребром, если соответствующие точки облака находятся поблизости в соответствии с некоторой метрикой (например, евклидово расстояние между точками меньше установленного порога). Таким образом, делаются предположения об условной независимости случайных величин, непосредственно не связанных ребрами при условии известных значений всех остальных величин. Задача поиска меток классов, максимизирующих эту вероятность, называется выводом в марковской сети. Она эквивалентна минимизации по всевозможным назначениям следующей функции энергии:

$$-\sum_{i=1}^{n}\log\phi_i\left(y_i\right)-\sum_{(i,j)\in\varepsilon}\log\phi_{ij}\left(y_i,y_j\right)\to\min_y, \quad (1)$$

где ε – множество ребер графа, а $\phi_i(y_i)$ и $\phi_{ij}(y_i, y_j)$ – соответственно унарные и парные потенциалы, которые могут задавать для каждой вершины и ребра вероятность получения соответствующего назначения либо выбираются эвристически.

Задача минимизации энергии (1) является в общем случае NP-трудной, но были разработаны эффективные приближенные алгоритмы [3]. В данной работе используется алгоритм TRW-S (sequential tree-reweighted message passing) (*http://www.cs.ucl.ac.uk/staff/V.Kolmogorov/papers/TRW-S.html*).

Предлагаемый метод

Опишем подробнее предлагаемые алгоритмы. Реализована следующая схема. Сначала строится пространственный индекс над обучающим/тестовым сканом, который также используется в качестве сегментации. Для сегментов находятся медоиды, и над ними строится граф – основа для марковской сети. (Здесь и далее *медоидом* для множества точек называется одна из точек множества, сумма расстояний до которой от всех точек множества минимальна. В реализации используется приближенный алгоритм поиска медоида.) Затем вычисляются унарные и парные потенциалы вершин и ребер. Для настройки унарных потенциалов используется алгоритм обучения рандомизированных решающих деревьев (Random Forest) [4]. В случае обучения ему на вход подается сокращенная выборка из векторов признаков точек обучающей выборки, при классификации – признаки медоидов сегментов. Для настройки парных потенциалов используется статистика признаков ребер по обучающему скану. В работе описан способ настройки парных потенциалов в рамках порождающей модели.

Когда унарные и парные потенциалы вычислены, запускается алгоритм вывода в марковской сети, который возвращает финальное назначение меток классов медоидам сегментов, распространяемым на сегменты целиком.

Пространственный индекс и пересегментация. Пространственный индекс над облаком точек позволяет быстро выполнять пространственные поисковые запросы (такие как «найти *k* ближайших соседей данной точки»). Для этой цели используется модифицированная структура данных P-дерево (R-Tree), которое представляет собой иерархию вложенных параллелепипедов (*http://graphics.cs.msu.ru/en/science/research/3dpoint/lidark*). Листья дерева содержат близкие точки пространства. Дерево полностью сбалансировано, то есть высота поддеревьев на каждом уровне постоянна. Это дает определенные гарантии по времени поиска.

Компактная форма множества точек в каждом из листьев позволяет использовать структуру дерева как пересегментацию. При классификации признаки вычисляются только для медоидов сегментов, затем марковская сеть строится только на медоидах. Результат классификации распространяется с медоидов на соответствующие сегменты. Это несколько огрубляет результаты классификации, но позволяет бороться с шумом сканирования и разметки. Важно, что пересегментация позволяет на порядок ускорить классификацию, делая возможным эффективную классификацию сканов размером в миллионы точек с помощью одной графической модели.

Построение графа. На сканах как обучающей, так и тестовой выборки необходимо строить графы над медоидами сегментов. В случае обучающей выборки собирается статистика для назначения парных потенциалов, в случае тестовой – назначаются парные потенциалы и производится вывод в марковской сети, основанной на построенном графе. Стандартный подход к построению графа – соединять ребрами все точки с их *k* ближайшими соседями, где *k* типично принимает значения от 3 до 5 [2]. Это позволяет сохранять примерно постоянную степень вершин в графе даже при меняющейся плотности сканирования. В случае ассоциативных марковских сетей задаются усредняющие потенциалы. В настоящей работе предлагается использовать более гибкую форму парных потенциалов, что делает разумным использование больших значений *k*. Для экспериментов используется k=5.

Данные аэросъемки имеют свои особенности. Такой типичный скан представляет собой карту высоты, снятую с отвесной (вертикальной) позиции. Это ведет к тому, что на карте отсутствуют вертикальные поверхности, соответствующие, например, заборам или стенам домов. Отсканированная поверхность оказывается в этом случае разрывной, а таким объектам, как крыша дома или крона дерева, соответствует отдельная компонента связности в графе, построенном по описанной выше методике. Поэтому при классификации данных аэросъемки предлагается добавлять в граф и ребра, соответствующие *k* соседним точкам в проекции на горизонтальную плоскость. Например, это позволяет связать ребрами крыши домов с соседними участками земли, что важно для описываемого метода, поскольку отношения между классами, такие как «крыша лежит выше земли», обрабатываются специальным образом, что не делается в ассоциативной модели.

Признаки и потенциалы. Для назначения унарных потенциалов используется выход классификатора Random Forest [4], представляющего собой набор решающих деревьев, каждое из которых обучено на случайном подмножестве признаков. Используется комитет из 50 деревьев. Предлагается применять вероятностный выход классификатора, считая части деревьев, проголосовавших за отдельные метки классов. Эти числа задают значения унарных потенциалов в марковской сети.

Для обучения классификатора нет необходимости использовать все точки обучающей выборки. Предлагается прореживать обучающую выборку, чтобы ускорить обучение, избежать эффекта переобучения и сбалансировать классы. Выборку следует прореживать так, чтобы в итоге в ней осталось равное число представителей всех классов. Классификатор, обученный на сбалансированной выборке, возвращает несмещенный результат. В противном случае классификатор игнорирует редкие классы, такие как «провод», «столб».

Используются следующие признаки для настройки классификатора [1]:

– спектральные признаки и признаки направления, отражающие, насколько окрестность точки похожа на линию или поверхность, а также их возможную ориентацию, – всего 7 признаков;

– спин-изображения [5] размера 9×18 с пониженной размерностью: точки аккумулировались не по клеткам сетки, а отдельно по строкам и столбцам; эксперименты показали, что такая редукция не приводит к потере различающей способности, – всего 27 признаков;

– угловые спин-изображения [5] размера 9×18, у которых была сокращена размерность таким же образом, – всего 27 признаков;

– распределение точек по высоте в цилиндрической окрестности точки, приближенное гистограммой, высота нижней точки цилиндра и разница между высотами данной точки и нижней точки цилиндра – 7 признаков.

Таким образом, вектор признаков для унарного классификатора состоит из 68 вещественных признаков. Эксперименты показали, что исключение каждой из групп признаков, а также сокращение размерности спин-изображений приводят к ухудшению результата классификации; в этом смысле набор признаков неизбыточен.

Обобщенная модель Поттса, широко используемая для назначения парных потенциалов в ассоциативных марковских сетях, имеет следующий недостаток: парный потенциал всегда равен нулю для различных меток класса ($-\log\phi_{ij}(k, l)=0$, если $k\neq l$; $-\log\phi_{ij}(k, k)<0$ иначе). Таким образом, в этой модели невозможно выразить какие-либо межклассовые взаимодействия (такие как «дерево не может лежать ниже земли»), в то время как они могут быть очень полезны. В данной работе подобные ограничения не вводятся.

Для ребер графа, построенного на медоидах сегментов обучающего или тестового скана, вычисляются следующие признаки:

– разница в высотах медоидов, нормализованная на длину ребра, или синус угла наклона ребра к горизонту;

– косинус угла между аппроксимированными нормалями в медоидах;

– расстояние между медоидами, или длина ребра.

Обозначим значения этих признаков для какого-либо ребра f_1, f_2 и f_3 соответственно. Используя теорему Байеса, легко вычислить вероятность назначения меток l_1 и l_2 паре вершин при условии наблюдаемых признаков:

$$P(l_1, l_2|f_1, f_2, f_3) = \\ = \frac{P(f_1|l_1, l_2)P(f_2|l_1,l_2)P(f_3|l_1, l_2)P(l_1,l_2)}{P(f_1, f_2, f_3)}. \qquad (2)$$

Эксперименты показали, что член $P(l_1, l_2)$ доминирует, то есть, если априорная вероятность для некоторой пары классов велика, вероятность получить любое другое назначение мала, даже если признаки голосуют за это. Таким образом, установлена равномерная априорная вероятность $P(l_1, l_2)=K^{-2}$ для всех l_1, l_2 и апостериорная вероятность оценивается как

$$P(l_1,l_2|f_1, f_2, f_3) \propto \\ \propto \frac{P(f_1|l_1, l_2)P(f_2|l_1, l_2)P(f_3|l_1, l_2)}{P(f_1, f_2, f_3)}. \qquad (3)$$

Для оценки вероятностей в правой части (3) признаковое пространство дискретизируется (распределения приближаются гистограммами), затем собирается соответствующая статистика с размеченного скана обучающей выборки. На стадии классификации парные потенциалы оцениваются в соответствии с признаками ребер:

$$\phi_{ij}(k, l) = P(k,l \mid f_1, f_2, f_3).$$

Вывод в марковской сети. Для вывода максимальной апостериорной оценки назначения в марковской сети часто используются алгоритмы на основе разрезов в графах. Для решения поставленной задачи они не применялись, так как потенциалы не удовлетворяют субмодулярным ограничениям. Поэтому использовались алгоритмы циклического распространения доверия (loopy belief propagation) и передачи сообщений с помощью перевзвешивания деревьев (TRW; tree-reweighted message passing) [3]. Последний показал лучшую производительность и эффективность. Точнее, использовалась модификация TRW-S (последовательная TRW), разработанная Колмогоровым [3]. Алгоритм находит глобальный максимум вогнутой функции – нижней границы функции энергии (1). Для этого на графе задается порядок вершин, затем его ребра покрываются монотонными цепями-подграфами (каждое ребро должно оказаться покрытым хотя бы одной цепью). Для эффективности вывода имеет смысл максимизировать длину таких цепей. Поскольку каждая цепь не содержит циклов, на ней возможен вывод с помощью передачи сообщений. Алгоритм выполняет итеративно две стадии – репараметризации и усреднения. На первой передаются сообщения с учетом порядка вершин и структуры монотонных цепей, на второй результат усредняется по всем деревьям. Колмогоров также доказал сходимость алгоритма и дал критерий останова, однако на практике удобно

просто ограничивать число итераций или следить за поведением неубывающей функции – нижней границы энергии.

Экспериментальные результаты

Для проведения экспериментов была подготовлена программная реализация на языке C++. Производительность метода оценивается на наборе данных аэросъемки, полученном с помощью сканирующей системы ALTM 2050 (Optech Inc.). Выделяются следующие классы объектов: «земля», «здание», «автомобиль», «дерево», «куст». Данные распределены между классами на первом скане в следующих пропорциях: 44,1 % земли, 2,0 % зданий, 0,3 % автомобилей, 53,1 % деревьев и 0,8 % кустов. Скан разбит на пространственно разнесенные обучающую и тестовую выборки размерами 1,1 и 1,0 миллиона точек (*http://graphics.cs. msu. su/ru/science/research/3dpoint/classification*).

Для обучения классификатора, назначающего унарные потенциалы, используются признаки не всех точек обучающего скана. Признаки считаются по исходному скану, но при формировании обучающей выборки векторы признаков прореживаются в соответствии с популярностью классов так, чтобы в выборке оказалось примерно равное число объектов каждого класса. Классификатор, обученный на такой выборке, выдает несмещенный результат. Таким образом, обучающая выборка состоит из 9 тысяч векторов признаков.

Для настройки наивного Байесовского классификатора необходимо приблизить распределения значений признаков гистограммами. Разность высот и косинус угла между нормалями принимают значения из сегмента [–1, 1]. Он делится на 10 равных интервалов, и считаются количества точек, значения признаков которых попадают в подинтервалы, затем полученные значения нормализуются. Признак-расстояние между точками делится на 6 корзин, которые находятся эмпирически, чтобы гистограмма хорошо приближала распределение. Границы корзин (2,0; 4,0; 6,0; 8,0; 12,0).

Было поставлено несколько экспериментов для сравнения предложенного метода с существующими подходами: выходом классификатора, не учитывающего зависимость между назначениями меток классов соседним точкам (без марковской сети), и с результатом марковской сети с постоянными парными потенциалами $\phi_{ij}(Y_i, Y_j)=[Y_i=Y_j]$, в которой ребрами соединены 5 ближайших соседей каждой точки. Информация о результатах работы методов отражена в таблице. Для предлагаемого метода приведены точность (процент верных точек среди найденных) и полнота (процент найденных среди верных), а также F-мера – их среднее гармоническое. Также указаны F-меры для результатов альтернативных подходов. Средние F-оценки по всем классам для независимой классификации, ассоциативной и неассоциативной марковских сетей равны 46,7 %, 46,8 %, 59,3 % соответственно.

Точность, полнота и F-мера по классам для предложенного метода (предл.), ассоциативных марковских сетей (AMN) и RandomForest (RF)

Класс	Точн. предл.	Полн. предл.	F-мера предл.	F-мера AMN	F-мера RF
Земля	0,898	0,962	0,929	0,842	0,870
Здание	0,868	0,585	0,699	0,079	0,258
Авто	0,370	0,161	0,224	0,255	0,175
Дерево	0,923	0,997	0,959	0,944	0,931
Куст	0,716	0,089	0,158	0,218	0,109

Ассоциативная марковская сеть с постоянными потенциалами сглаживает результаты, удаляя как шум, так и мелкие классы. Неассоциативная марковская сеть содержит больше ребер, потенциалы которых зависят от расстояния между точками и других признаков. Это ведет к более разборчивому сглаживанию. Такие потенциалы позволяют выражать отношения типа «крыша находится выше земли», что приводит к более качественной сегментации. На рисунке 2 приведен результат классификации для части тестового скана.

Как видно из рисунка, после применения неассоциативной марковской сети все равно остаются ошибки. Но это ошибки другого рода. Ассоциативная марковская сеть часто неправильно классифицирует целые мелкие объекты, в то время как описываемый алгоритм ошибается в частях одного объекта. Подобные ошибки можно исправить на этапе постобработки с помощью фильтрации либо более аккуратной настройки потенциалов.

Таким образом, в статье предложен новый метод классификации данных лазерного сканирования, основанный на аппарате неассоциативных марковских сетей. Показано, как можно осуществлять приближенный вывод в таких сетях. Для обучения потенциалов применяется генеративный подход на основе наивного байесовского классификатора. Согласно результатам экспериментов, метод превосходит традиционный подход с ассоциативными марковскими сетями по качеству классификации.

Литература

1. Shapovalov R., Velizhev A. and Barinova O. Non-associative Markov networks for 3D point cloud classification, Photogrammetric Computer Vision and Image Analysis, Paris, France: 2010.

2. Anguelov D. [et al]. Discriminative Learning of Markov Random Fields for Segmentation of 3D Scan Data, IEEE Conference on Computer Vision and Pattern Recognition, San Diego, CA: 2005, pp. 169–176.

3. Szeliski R. [et al]. A comparative study of energy minimization methods for Markov random fields, Lecture Notes in Computer Science. 2006. Vol. 3952, p. 16.

4. Breiman L. Random forests. Machine Learning. 2001. Vol. 45, pp. 5–32.

5. Endres F. [et al]. Unsupervised Discovery of Object Classes from Range Data using Latent Dirichlet Allocation, Robotics: Science and Systems, Seattle, WA: 2009.

УДК 004.932.2

ВЫДЕЛЕНИЕ ОБЛАСТЕЙ ИНТЕРЕСА НА ОСНОВЕ КЛАССИФИКАЦИИ ИЗОЛИНИЙ

О.В. Сенюкова; В.Е. Галанин
(МГУ им. М.В. Ломоносова, olsen222@yandex.ru)

Предлагается новый подход к выделению областей интереса на изображениях. Показано применение разработанного алгоритма к задаче выделения очагов поражения на изображениях магнитно-резонансной томографии мозга. Результаты тестирования разработанного алгоритма на реальных данных продемонстрировали его превосходство над аналогами.

Ключевые слова: *компьютерное зрение, машинное обучение, сегментация изображений, семантическая сегментация, обработка изображений, изолинии, выделение очагов поражения.*

Задача семантической сегментации – выделение объектов или областей интереса на изображениях – особенно сложна при нечетких границах регионов, сильной зашумленности, отсутствии цветовой информации. Подобными свойствами обладают, например, изображения, полученные с помощью медицинского оборудования. Поэтому традиционные методы сегментации изображений, основанные на разрастании регионов, выделении границ, кластеризации и т.д., подходят для решения лишь небольшого класса задач, связанных с обработкой таких изображений.

В данной статье предлагается новый метод выделения областей интереса, который применим к изображениям с названными выше свойствами. Работа метода продемонстрирована на примере задачи выделения очагов поражения на изображениях *магнитно-резонансной томографии* (МРТ) мозга. Очаги поражения на изображениях МРТ мозга – это обычно небольшие области повышенной или пониженной интенсивности, встречающиеся в нескольких местах. Иногда их количество достаточно велико. Они могут различаться между собой по форме и резкости границ, имеют вариации интенсивности и, как правило, визуально хорошо различимы по какому-либо признаку или набору признаков.

Предложенный алгоритм направлен на то, чтобы максимально учесть глобальные свойства подобных областей интереса – те свойства, которыми руководствуется человек при визуальном анализе.

Алгоритм основан на построении изолиний функции интенсивности, расчете признаков на основе этих изолиний и построении классификатора, который идентифицирует изолинии, являющиеся контурами областей интереса.

Обзор существующих алгоритмов

В одной из первых работ в области выделения очагов поражения на изображениях МРТ мозга описан алгоритм, основанный на классификации пикселей в зависимости от их интенсивности с помощью искусственной нейронной сети [1].

Фундаментальная работа [2] посвящена выделению областей интереса на основе принципа нечеткой связности. Связным объектом считается множество точек, в котором каждая пара точек удовлетворяет некоторому критерию связности. В данном случае этот критерий строится на основе разности интенсивностей пикселей и их пространственной близости.

Алгоритмы из [1] и [2] являются полуавтоматическими, то есть требуют взаимодействия с пользователем. Для их работы необходимо предварительно отметить на изображениях достаточное количество точек, относящихся к другим областям, фону и/или непосредственно к областям интереса. Данный подход может оказаться неустойчивым, так как принципиально зависит от аккуратности пользовательского ввода. К тому же у разных пользователей результаты могут сильно отличаться.

Большинство существующих методов являются полностью автоматическими.

Достаточное количество работ посвящено алгоритмам, которые используют классификаторы для отнесения пикселя к объекту (области интереса) или фону (за пределами области интереса) подобно [1]. Вместо нейронных сетей применяются более современные и качественные классификаторы – метод опорных векторов, рандомизированный решающий лес и т.д. [3]. Для обучения классификаторов вместо отмеченных пользователем точек, как в [1] и [2], как правило, используются размеченные базы изображений.

В таких алгоритмах решение об отнесении к объекту или фону принимается на уровне отдельных пикселей, имеющих весьма ограниченный набор характеристик: интенсивность, координаты, информация о соседях. Этих сведений не всегда достаточно для принятия правильного решения.

Существуют и другие подходы. Так, метод, описанный в [4], использует кластеризацию пикселей по интенсивности с помощью алгоритма ISODATA (итерационная самоорганизующаяся методика анализа данных). Но при построении кластеров также берется лишь информация об интенсивности отдельных пикселей.

Авторский алгоритм

В данной работе предлагается новый подход – классификация происходит сразу на уровне линий, а не на уровне отдельных пикселей. Это позволяет учесть информацию о форме, резкости границ и другие признаки объекта интереса, что невозможно в случае попиксельной классификации.

Разработанный алгоритм является полуавтоматическим. На каждом изображении МРТ врач-радиолог выделяет прямоугольные регионы, в которых он подозревает наличие очагов поражения, если они имеются. Подобный подход описан в [5], где аналогичные ограничивающие прямоугольники называются *слабыми метками*. Далее внутри каждого из этих регионов очаги поражения выделяются автоматически.

Данный подход позволяет повысить качество разметки по сравнению с полностью автоматическими методами и в то же время значительно сократить объем рутинной работы, необходимой при ручной разметке.

Для построения замкнутых линий, среди которых имеются контуры искомых очагов поражения, применяется алгоритм построения изолиний функции интенсивности. Отбор линий, действительно являющихся контурами очагов поражения, происходит путем классификации методом опорных векторов.

Построение изолиний. Алгоритм построения изолиний в общем случае используется для построения контурных карт – карт изолиний некоторой функции от двух переменных. Изолиния функции f(x, y) – это геометрическое место точек (x, y), таких, что f(x, y)=*const*. Линия уровня h функции f(x, y) – это геометрическое место точек *(x,y)*, таких, что f(x, y)=h. Если h входит в диапазон значений функции f(x, y), то для уровня h могут существовать одна или несколько изолиний.

Задача построения изолиний часто встречается в топографии для построения топографических карт с горизонталями (линиями равных высот), в метеорологии, геологии и в других областях.

В настоящей работе используется алгоритм, наиболее близкий к описанному в онлайн-документации к MATLAB (*http://www.mathworks.com/help/techdoc/creating_plots/f10-2524.html#f10-2614*). В данном случае в роли функции f(x, y) выступает функция интенсивности пикселя I(x, y). По изображению строится карта высот, где высота точки (x, y) задается интенсивностью пикселя с координатами (x, y) на исходном изображении. Чтобы линии получались гладкими, определенным образом применяется интерполяция.

В контексте задачи выделения областей интереса наибольшего внимания заслуживают замкнутые изолинии. Так, существует набор уровней, для которых замкнутые изолинии в точности соответствуют контурам очагов поражения на изображениях МРТ мозга (см. рис. 1).

Классификация методом опорных векторов. Чтобы отделить замкнутые изолинии, соответствующие контурам искомых областей, от других замкнутых изолиний, требуется набор признаков, который будет использоваться классификатором. В задаче выделения очагов поражения на изображениях МРТ мозга врач-радиолог может идентифицировать нужные линии визуально. Необходимо подобрать признаки, наиболее схожие с теми, которыми пользуется человек при визуальном анализе и которые обеспечивают наилучшее разделение между искомыми и остальными замкнутыми изолиниями. Предлагается следующий набор признаков:

- площадь фигуры, ограниченной линией;
- периметр фигуры, ограниченной линией;
- отношение средней интенсивности внутри и снаружи замкнутой линии;
- разность интенсивностей вдоль внешней и внутренней границ замкнутой линии;
- удлиненность фигуры, ограниченной линией.

Обучение классификатора. Для обучения классификатора строится обучающая выборка. Она состоит из n элементов, каждый из которых представляет собой вектор-признак $\mathbf{x}_i \in R^r$ и соответствующую ему метку класса $y_i \in \{+1, -1\}$: $X^n=\{(\mathbf{x}_i, y_i): i=1, 2, \ldots, n\}$.

В эту выборку входят вектор-признаки, состоящие из описанных выше признаков для всевозможных замкнутых изолиний, построенных для некоторого диапазона уровней (интенсивностей), а также метки «очаг» или «не-очаг», соответствующие каждой из этих изолиний. Метка класса y_i принимает два значения: 1 в случае «очаг» и –1 в случае «не-очаг».

С целью наибольшей репрезентативности обучающая выборка формируется на основе изолиний, построенных по разным изображениям и регионам интереса.

Для построения классификатора используется нелинейный метод опорных векторов с ядром RBF. Этот метод продемонстрировал лучшие результаты в отличие от метода с применением по-

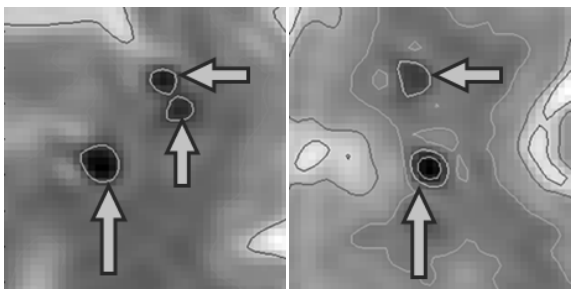


Рис. 1. Построение изолиний нескольких уровней для двух различных регионов интереса. Стрелками показаны замкнутые линии, соответствующие контурам очагов поражения

линомиального ядра и линейного метода опорных векторов.

Автоматическое выделение областей интереса внутри указанного региона. Чтобы найти контуры областей интереса (в данном случае очагов поражения) внутри выделенного прямоугольного региона, необходимо сделать следующее:

– построить изолинии для того же диапазона уровней, который использовался в обучающей выборке;

– отобрать из построенных изолиний замкнутые линии;

– для каждой из замкнутых линий рассчитать вектор-признак;

– применить построенный в процессе обучения классификатор к каждому вектор-признаку;

– оставить только те замкнутые линии, для которых классификатор выдал метку «очаг».

Таким образом, для каждого указанного прямоугольного региона с помощью описанной выше процедуры можно автоматически построить изображение, на котором обозначены контуры искомых областей.

Эксперименты

Для обучения и тестирования использовались данные, полученные из НИИ неотложной детской хирургии и травматологии (г. Москва). База состояла из МРТ-последовательностей мозга для семи пациентов, имеющих очаги диффузно-аксонального повреждения.

Тестирование проводилось методом кросс-валидации. Каждая из семи МРТ-последовательностей поочередно была тестовой, а другие шесть в это время использовались для обучения. Результаты тестирования разработанного алгоритма и ISODATA приведены в таблице 1.

Таблица 1

№ последовательности	Чувствительность		Избирательность		DSC	
	ISODATA	Наш метод	ISODATA	Наш метод	ISODATA	Наш метод
1	0,64	**0,93**	**0,31**	0,24	0,45	**0,83**
2	0,71	**0,98**	**0,28**	0,25	0,51	**0,79**
3	0,63	**0,93**	**0,35**	0,23	0,48	**0,78**
4	0,61	**0,94**	**0,31**	0,11	0,38	**0,73**
5	0,57	**0,93**	**0,38**	0,26	0,49	**0,83**
6	0,61	**0,96**	0,28	**0,30**	0,52	**0,80**
7	0,69	**0,97**	**0,30**	0,25	0,53	**0,83**
Среднее	0,64	**0,95**	**0,32**	0,23	0,48	**0,80**

Чувствительность алгоритма рассчитывается как $\frac{TP}{TP+FN}$, избирательность – как $\frac{TN}{TN+FP}$, DSC (Dice Similarity Coefficient – мера схожести множеств) – как $\frac{2 \cdot TP}{2 \cdot TP + FP + FN}$, где *TP* (True Positive) – количество изолиний (в предложенном алгоритме) или кластеров (в алгоритме ISODATA), правильно классифицированных как объект интереса («очаг»); *FN* (False Negative) – количество изолиний (кластеров), ошибочно классифицированных как фон («не-очаг»); *TN* (True Negative) – количество изолиний (кластеров), правильно классифицированных как фон; *FP* (False Positive) – количество изолиний (кластеров), ошибочно классифицированных как объект интереса.

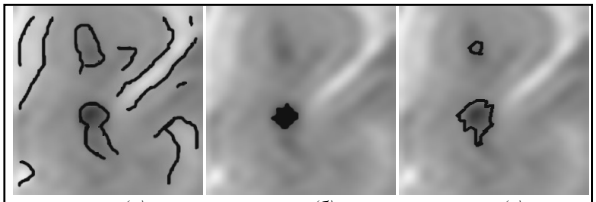
В таблице 2 предложенный алгоритм сравнивается с алгоритмом Xie [5], в котором используются похожие «слабые метки» – прямоугольные регионы интереса, предварительно указанные врачом.

Таблица 2

Алгоритм	Чувствительность	Избирательность	DSC
Xie	0,74	**0,68**	0,71
Our	**0,95**	0,23	**0,80**

Из таблиц 1 и 2 видно, что и чувствительность предложенного алгоритма значительно превосходит аналоги, и коэффициент DSC выше. О показателях избирательности следует сказать, что большинство лишних изолиний отбрасывается еще на этапе выбора диапазона уровней для построения изолиний, потому линий с меткой «не-очаг» в обучающей выборке всегда примерно в 2 раза меньше, чем линий с меткой «очаг». Таким образом, показатели избирательности предложенного алгоритма объективно выше приведенных в таблице значений. Более того, чувствительность алгоритма в медицинских приложениях, как правило, важнее избирательности, так как цена ошибки первого рода (пропустить заболевание) гораздо выше цены ошибки второго рода (ложной тревоги).

В своей работе авторы проводили сравнение предложенного алгоритма с базовыми алгоритмами сегментации – детектором краев Canny и разрастанием регионов. Результаты сравнения приведены на рисунке 2.



(а) *(б)* *(в)*

Рис. 2. Сравнение предложенного алгоритма с базовыми алгоритмами сегментации: а) алгоритм Canny; б) разрастание регионов; в) предложенный алгоритм

Как видно из рисунка 2а, алгоритм Canny дает слишком много лишних линий, к тому же контуры очагов поражения могут получаться незамкнутыми. Разрастание регионов (рис. 2б) в целом демонстрирует лучшие результаты, чем детекторы краев, но этот метод пропускает много пикселей, относящихся к очагам поражения, и/или дает много ложных срабатываний. Предложенный авторами

алгоритм значительно превосходит оба этих алгоритма (рис. 2в).

Изолинии были построены для 20 уровней интенсивности из диапазона $\left[\min_{x,y} I(x,y)+10,\ \min_{x,y} I(x,y)+70\right]$. Эти значения подобраны в ходе экспериментов.

Для реализации предложенного алгоритма было разработано интерактивное приложение на языке C#. Основными функциями приложения являются:

– загрузка и отображение изображений МРТ в формате DICOM;

– предложенный полуавтоматический алгоритм для выделения очагов поражения (отдельные окна для выделения прямоугольных регионов интереса, обучения классификатора, тестирования);

– дополнительные функции для предварительной обработки изображений (эквализация гистограммы, гамма-коррекция и др.).

На основании изложенного можно сделать следующие выводы. Предложенный подход к выделению областей интереса на изображениях основан на классификации изолиний, он может быть особенно полезен при обработке изображений, полученных с помощью медицинского оборудования. Разработанный алгоритм применен к задаче выделения очагов поражения на изображениях МРТ мозга. В работе проведен анализ существующих алгоритмов для решения подобных задач, перечислены их недостатки. Показано значительное превосходство предложенного алгоритма над базовыми алгоритмами сегментации, основанными на выделении краев, разрастании регионов, кластеризации. Кроме того, проведено сравнение алгоритма с одним из новых полуавтоматических алгоритмов выделения очагов поражения [5], которое показало общее превосходство предложенного алгоритма почти на 10 %.

Авторы выражают благодарность А.В. Петряйкину, А.С. Крылову и Т.А. Ахадову за ценные консультации.

Литература

1. Zijdenbos A.P., Dawant B.M., Margolin R.A., Palmer A.C. Morphometric Analysis of White Matter Lesions in MR Images: Method and Validation, IEEE Transactions on Medical Imaging. 1994. Vol. 13, № 4, pp. 716–724.

2. Udupa J.K., Samarasekera L.W.S., Miki Y. [et al.]. Multiple Sclerosis Lesion Quantification Using Fuzzy-Connectedness Principles. IEEE Transactions on Medical Imaging. 1997. Vol. 16, № 5, pp. 598–609.

3. Scully M., Anderson B., Lane T. [et al.]. An Automated Method for Segmenting White Matter Lesions through Multi-level Morphometric Feature Classification with Application to Lupus, Frontiers in Human Neuroscience. 2010. Vol. 4, № 27.

4. Hillary F.G., Biswal B.B. Automated Detection and Quantification of Brain Lesions in Acute Traumatic Brain Injury Using MRI. Brain Imaging and Behavior. 2009. Vol. 3, pp. 111–122.

5. Xie Y., Tao X. White Matter Lesion Segmentation Using Machine Learning and Weakly Labeled MR Images. Proceedings of SPIE. 2011. Vol. 7962, pp. 79622G. URL: http://spiedigitallibrary.org/proceedings/resource/2/psisdg/7962/1/79622G_1 (дата обращения: 14.09.2011).

УДК 004.925.86

ПОСТРОЕНИЕ КРИВОЙ СКОЛЬЖЕНИЯ КОНИЧЕСКОГО ИНСТРУМЕНТА ПРИ МНОГОКООРДИНАТНОЙ ОБРАБОТКЕ

А.И. Будник; Е.И. Кац, к.т.н.
(Уральский федеральный университет им. Б.Н. Ельцина, г. Екатеринбург, budnikalexandr@mail.ru)

Рассматривается задача построения кривой скольжения на поверхности конического инструмента в определенный момент времени. Разработан алгоритм построения кривой скольжения на поверхности конического инструмента. Моделирование такой кривой необходимо, в первую очередь, для моделирования поверхности (объема), заметаемой инструментом в процессе многокоординатной фрезерной обработки.

Ключевые слова: *многокоординатная обработка, геометрическое моделирование, кривая скольжения, граничные точки, конический инструмент.*

Решение задачи построения кривой скольжения на поверхности конического инструмента в определенный момент времени необходимо, в первую очередь, для моделирования поверхности (объема), заметаемой инструментом в процессе многокоординатной фрезерной обработки. Эта задача, в свою очередь, актуальна при разработке CAM-систем и систем моделирования обработки.

Для решения рассматриваемой задачи рядом исследователей предложены различные методы (см. например [1–3]). Настоящая статья посвящена разработке подхода, в наибольшей степени соответствующего требованиям имеющегося пакета геометрического моделирования фрезерной обработки на станках с ЧПУ. Получено решение, обеспечивающее приемлемое быстродействие при требуемой точности построения.

Особенностью предлагаемого подхода является использование линейности изменения станочных координат положения инструмента при обработке.

В любой момент времени вектор мгновенной скорости в пространстве изменяется линейно от одной точки к другой, что позволяет существенно упростить поиск точек на кривой скольжения. Кроме того, удалось построить алгоритм поиска таким образом, чтобы отсекаемая часть аппроксимированной поверхности инструмента оставалась выпуклой.

Постановка задачи: пусть на входе имеются контур конического инструмента, полная производная матрицы скорости по времени, матрица инструмента. Изменение всех координат считается линейным по времени [4]. Необходимо построить кривую скольжения на поверхности инструмента в данный момент времени.

При рассмотрении задачи контур инструмента задан в плоскости xOz в положительной области по оси Ox. Инструмент может состоять из нескольких простых инструментов. Сам контур инструмента изначально не имеет экстремумов вдоль оси Ox. Контур непрерывен. Он начинается и оканчивается на оси Oz.

Для построения кривой скольжения необходимо дать ее определение. Кривая скольжения – это замкнутый контур на поверхности инструмента, в каждой точке которого скалярное произведение вектора внешней нормали к поверхности и вектора мгновенной скорости равно нулю [1]. Такое скалярное произведение в точках на поверхности режущей части инструмента является положительным (назовем эту область положительной). В результате задача построения кривой скольжения переходит в задачу построения кривой, ограничивающей положительную область поверхности инструмента. Для решения задачи необходимо научиться находить точки кривой скольжения.

Как известно, при движении сферической поверхности точки, лежащие на окружности большого радиуса в плоскости, перпендикулярной поступательной составляющей вектора скорости в центре сферы, будут образовывать кривую скольжения [5]. Назовем их граничными точками (grazing points). Поскольку инструмент, как и его контур, имеет вертикальную ориентацию относительно своей оси и существует понятие режущей стороны инструмента, есть смысл говорить о левой и правой сторонах кривой скольжения в направлении движения инструмента. Для построения кривой скольжения возьмем окружность с центром на оси инструмента, перпендикулярную этой оси, то есть набор точек на поверхности инструмента, расположенных на определенной высоте. Так как вектор скорости **d** в граничной точке должен быть перпендикулярным вектору внешней нормали **n**, пара граничных точек слева и справа должна располагаться в плоскости, перпендикулярной вектору **d**, а вектор **n** лежать в этой плоскости. Из рисунка 1 видно, что точки M_0 и M_1 лежат на кривой скольжения, так как удовлетворяют ее определению. При одной точке пересечения плоскости, перпендикулярной вектору **d** и окружности, взятой на некоторой высоте, можно говорить о петле кривой скольжения. Тогда кривая скольжения либо заканчивается в этой точке по причине замкнутости, либо будет состоять из нескольких компонент связности, каждая из которых удовлетворяет определению, данному выше.

Рассмотрим случай с конической поверхностью (рис. 1). Для нахождения граничных точек на некоторой высоте по оси Oz возьмем окружность с центром E на оси конуса, лежащую на его боковой поверхности и одновременно в плоскости, перпендикулярной оси Oz. Точки M_0 и M_1 имеют ту же скорость, что и точка Q. В этих точках внешние нормали перпендикулярны вектору скорости **d**, а значит, являются граничными точками.

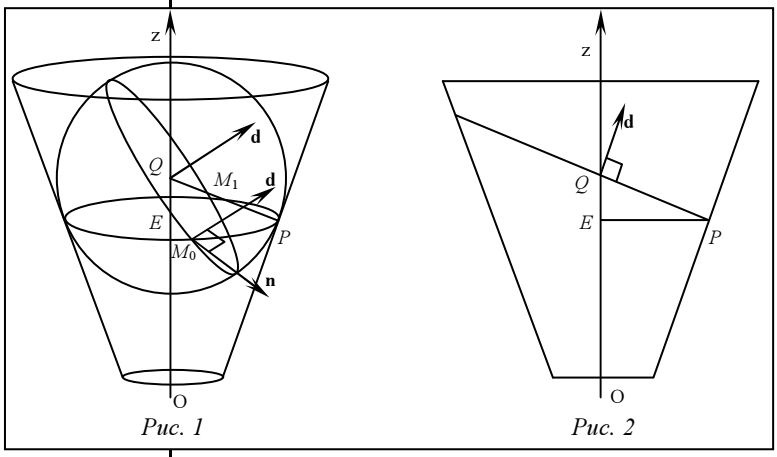


Рис. 1 *Рис. 2*

При варианте с конусом наличие лишь одной граничной точки на определенной высоте по Oz возможно в двух случаях (рис. 2). Для этого необходимо, чтобы вектор скорости **d** в точке Q центра сферы имел тот же угол наклона к оси Oz, что и контур конуса. Тогда получаются две петли на конусе, причем будет справедливо следующее утверждение:

$$\frac{\vec{d}\cdot\vec{k}}{\left|\vec{d}\right|\cdot\left|\vec{k}\right|}=\cos q\,, \qquad (1)$$

где $\vec{d}$ – вектор мгновенной скорости в точке Q; $\vec{k}$ – орт оси Oz; q – угол полураствора контура.

Так как скорость вычисляется по формуле

$$\vec{d}(t)=\dot{P}(t)=P_0\dot{M}(t)\,, \qquad (2)$$

где $\vec{d}(t)$ – вектор мгновенной скорости в момент времени t; $\dot{P}(t)$ – производная закона движения точки по t; P_0 – точка в системе координат инструмента; $\dot{M}(T)$ – полная производная матрицы ско-

рости по времени, то можно утверждать, что вектор скорости линейно зависит от координат точки.

Представим в параметрическом виде вектор скорости $\vec{d}$ через векторы скорости $\vec{\upsilon}_0$ и $\vec{\upsilon}_1$:

$$\frac{\left(\left(\vec{\upsilon}_0+\left(\vec{\upsilon}_1-\vec{\upsilon}_0\right)\cdot t\right)\cdot\vec{k}\right)^2}{\left(\vec{\upsilon}_0+\left(\vec{\upsilon}_1-\vec{\upsilon}_0\right)\cdot t\right)^2}=\cos^2 q\,, \qquad (3)$$

где $\vec{\upsilon}_0$, $\vec{\upsilon}_1$ – векторы скоростей в точках Q_0 и Q_1; t – параметр вдоль оси инструмента (рис. 3).

Корнями уравнения

$$t=\frac{|Q-Q_0|}{|Q_1-Q_0|}, \qquad (4)$$

где Q, Q_0, Q_1 – центры сфер, касательных к поверхности инструмента, служат значения параметров t, используя которые, можно найти точки Q для вершин петель, образованных кривой скольжения.

Используя рисунок 3, получим следующее выражение для вычисления координаты z экстремумов петель:

$$z=\frac{t\cdot\left(1+\mathrm{tg}^2 q\right)\cdot\left(z_1-z_0\right)+\left(z_0+R_0\cdot\mathrm{tg}q\right)-\left(R_0\cdot\mathrm{tg}q-z_0\cdot\mathrm{tg}^2 q\right)}{\left(1+\mathrm{tg}^2 q\right)}, \qquad (5)$$

где z_0, z_1 – координаты z вершин A и B контура конуса; R_0 – радиус нижнего основания по оси Oz.

На элементе инструмента может располагаться одна петля в случае выхода одного из найденных z за пределы этого элемента или при вырождении уравнения (3) в линейное. У цилиндра, очевидно, $\cos^2 q=1$ и, следовательно, совпадут точки Q и E. В итоге возможны два варианта: либо плоскость, перпендикулярная **d**, пересекает выбранную на определенной высоте окружность в двух точках, либо содержит ее целиком. Для построения кривой скольжения необходимо найти граничные точки, лежащие на ней.

Рассмотрим несколько способов нахождения граничных точек на кривой скольжения с последующим ее построением.

Первый способ – нахождение граничных точек для конкретных значений z, взятых с некоторым фиксированным шагом по высоте инструмента. Отклонение аппроксимирующей кривой от истинной кривой скольжения при этом не учитывается. Количество точек определено лишь количеством шагов вдоль оси инструмента и видом кривой.

Второй способ – поиск граничных точек с учетом отклонения аппроксимирующей ломаной от истинной кривой скольжения. Для каждой найденной точки аппроксимирующей кривой в направлении касательного вектора к кривой подбирается такой шаг, чтобы отклонение касательного вектора от истинной кривой скольжения не превышало допустимого значения. Для полученного значения z конца вектора ищется следующая граничная точка. В первом и втором способах используется метод касательной сферы [5].

Третий способ основан на поиске граничных точек с учетом линейности изменения вектора мгновенной скорости при движении по отрезку. Таким образом можно находить граничные точки на прямых, образующих боковую поверхность конического инструмента (рис. 4), используя определение кривой скольжения. На концах образующих вычисляются векторы мгновенных скоростей. Из линейного уравнения находится точка на образующей, в которой скалярное произведение нормали к поверхности и вектора мгновенной скорости равно нулю. При этом учитывается отклонение при построении образующих конуса. Количество точек ограничено габаритами инструмента, значением максимально допустимого отклонения и геометрией самой кривой скольжения. Для большей точности образующие подбираются так, чтобы точно найти экстремум петли граничной кривой.

Такой способ позволяет просто решить еще одну важную задачу. Нахождение кривой скольжения нужно, в первую очередь, для построения заметаемой поверхности, частью которой будет часть поверхности инструмента, ограниченная кривой скольжения. При аппроксимации этой поверхности многогранником (сеткой из треугольников) важно сохранить выпуклость. Если находить граничные точки одним из первых двух способов, а затем просто соединять их с точками на направляющей, то выпуклость, вообще говоря, будет нарушена. Третий же способ лишен этого недостатка, так как, если соединять найденные точки с концами соответствующих образующих, выпуклость гарантирована.

На основании полученных результатов разработаны программы моделирования кривой скольжения

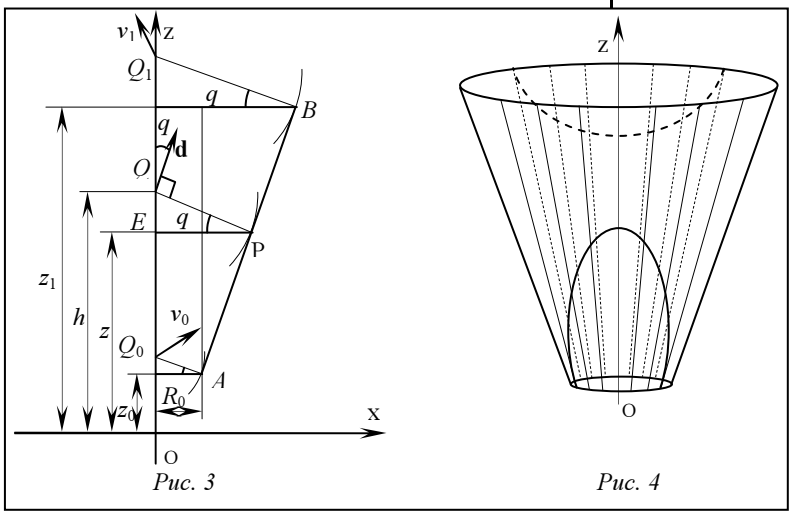


Рис. 3

Рис. 4

на поверхности конического инструмента при многокоординатной обработке. Программный модуль добавлен в состав симулятора обработки на станках с ЧПУ. Полученные результаты служат основой для построения поверхности, заметаемой инструментом при многокоординатной обработке.

Литература

1. Wang W.P., Wang K.K. Geometric modeling for swept volume of moving solids. IEEE Computer Graphics and Applications. 1986. № 6 (12), pp. 8–17.

2. Abdel-Malek K., Seaman W., Yeh H.-J. NC-verification of up to 5 axis machining processes using manifold stratification. ASME Journal of Manufacturing Science and Engineering. 2000. № 122, pp. 1–11.

3. Weinert K., Du S.-J., Damm P., Stautner M. Swept volume generation for the simulation of machining processes. International Journal of Machine Tools and Manufacture. 2004. № 44 (6), pp. 617–28.

4. Будник А.И., Кац Е.И. Построение траектории движения инструмента при многокоординатной обработке // Программные продукты и системы. 2009. № 3.

5. Eyyup Aras. Generating cutter swept envelopes in five-axis milling by two-parameter families of spheres. Computer-Aided Design. 2009. № 41, pp. 95–105.

УДК 004.925.86

КРИВАЯ СКОЛЬЖЕНИЯ НА ИНСТРУМЕНТЕ ПРОИЗВОЛЬНОЙ ФОРМЫ ПРИ МНОГОКООРДИНАТНОЙ ОБРАБОТКЕ

А.И. Будник (Уральский федеральный университет им. Б.Н. Ельцина, г. Екатеринбург, budnikalexandr@mail.ru)

Построение кривой скольжения при многокоординатной фрезерной обработке аналитическим методом позволяет рассчитывать на более точное представление обработанной поверхности. В статье рассматриваются все возможные варианты форм кривой в зависимости от формы инструмента. На основе алгоритма построения создан программный модуль в составе симулятора NCManager.

Ключевые слова: *многокоординатная обработка, геометрическое моделирование, кривая скольжения, граничные точки, заметаемая поверхность.*

Для построения кривой скольжения существуют различные методы, предложенные, например, в работах [1, 2]. Решение задачи в дальнейшем необходимо для построения поверхности, заметенной инструментом при геометрическом моделировании фрезерной обработки на станках с ЧПУ. В данной статье рассматриваются особенности построения кривой скольжения на поверхности произвольного по форме инструмента.

Основная постановка задачи: на входе имеются контур инструмента, полная производная матрицы скорости по времени, матрица инструмента. Необходимо построить кривую скольжения на инструменте при движении в данный момент.

При рассмотрении задачи контур инструмента задан в плоскости xOz в положительной области по оси Ox. Инструмент может состоять из нескольких простых инструментов, контур каждого из которых представляет набор определенных элементов. В качестве элементов контура могут быть дуга окружности и прямая. В результате инструмент как тело вращения может представлять собой совокупность нескольких поверхностей вращения (рис. 1), таких как конус (цилиндр – как частный случай), сфера, тор, плоскость.

Кривой скольжения у сферы всегда будет являться окружность. Из постановки задачи понятно, что сфера может быть представлена максимум четвертью своей поверхности. Значит, кривая скольжения будет выглядеть как дуга, не превышающая четверти окружности.

Рассмотрим применение способа нахождения точек кривой скольжения методом касательной сферы (рис. 2) для таких видов элементов инструмента, как тор, сфера и плоскость.

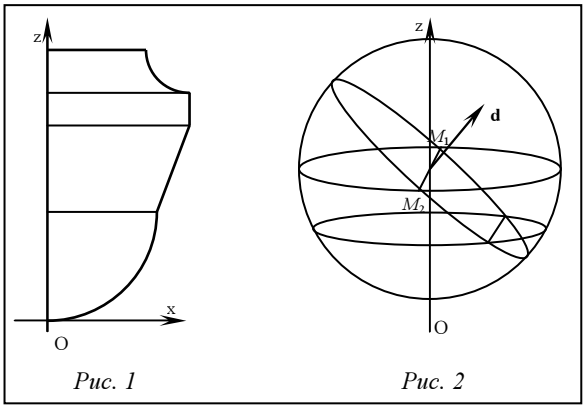


Рис. 1 *Рис. 2*

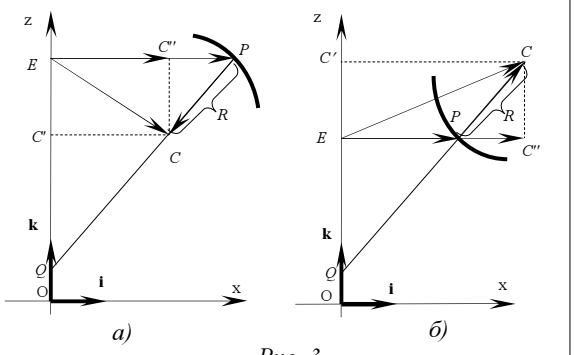
На инструменте фрагмент поверхности тора может быть выпуклым как внутрь, так и наружу. Рассмотрим оба варианта расположения контура элемента инструмента. На рисунке 3а изображен контур выпуклого наружу фрагмента тора.

Для поиска центра Q сферы, касающейся выпуклого тора в точках на высоте точки E, используется следующее выражение, полученное по рисунку 3а:

$$Q = \mathrm{E} + \vec{\mathrm{i}} \cdot \left|E\vec{P}\right| + P\vec{C} \cdot \frac{\left|E\vec{P}\right|}{\sqrt{R^2 - \left|E\vec{C} \cdot \vec{k}\right|^2}}. \qquad (1)$$

Случай тора, выпуклого внутрь инструмента, изображен на рисунке 3б. Выражение для Q будет выглядеть следующим образом:

$$Q = \mathrm{E} + \vec{\mathrm{i}} \cdot \left|E\vec{P}\right| - P\vec{C} \cdot \frac{\left|E\vec{P}\right|}{\sqrt{R^2 - \left|E\vec{C} \cdot \vec{k}\right|^2}}. \qquad (1.1)$$



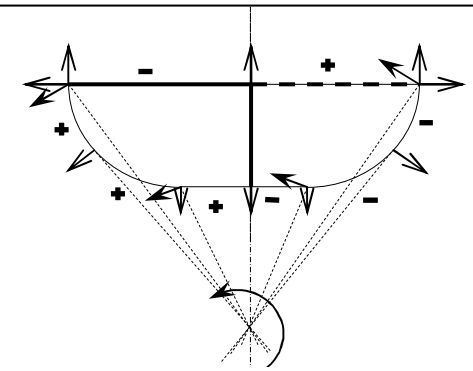
а) *б)*

Рис. 3

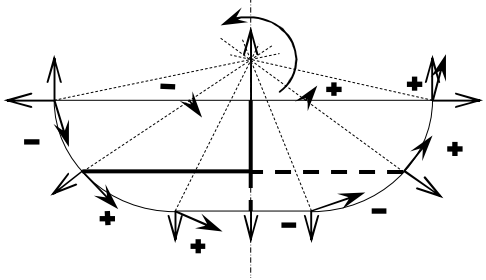
Если нормаль на поверхности тора параллельна его оси и точка Q стремится в бесконечность, радиус сферы $|QP|$ с центром в Q будет бесконечно большим. Необходимо, чтобы граничные точки лежали на расстоянии $\left|E\vec{P}\right|$ от оси инструмента. Это возможно только тогда, когда плоскость, перпендикулярная вектору скорости $\vec{d}$ в точке Q, перпендикулярна окружности в плоскости, перпендикулярной оси Oz с центром в точке E и с радиусом, равным $\left|E\vec{P}\right|$. Граничные точки при этом могут быть лишь тогда, когда вектор скорости в точке Q будет перпендикулярен вектору $\vec{k}$.

Поиск кривой скольжения происходит таким же образом, что и у сферы. Разница лишь в геометрии получившейся кривой скольжения. На рисунках 4а–4г показаны различные ситуации расположения кривой скольжения на поверхности тора. В качестве положительной области выступает режущая часть инструмента, в качестве отрицательной – нережущая.

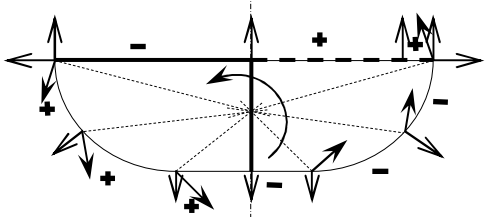
Вращение вокруг оси, пересекающей ось инструмента, дает наличие двух экстремумов того же типа, как и у вырожденного в цилиндр конуса. Экстремумов вдоль оси Oz кривой скольжения на торе может быть как один, так и два. Случай с одной петлей является более общим случаем расположения кривой на поверхности тора (рис. 4г). Пунктирными линиями на рисунках изображены радиусы вращения граничных точек. Как и в предыдущих случаях, угол между вектором нормали к поверхности и вектором скорости определяет знак скалярного произведения (2).



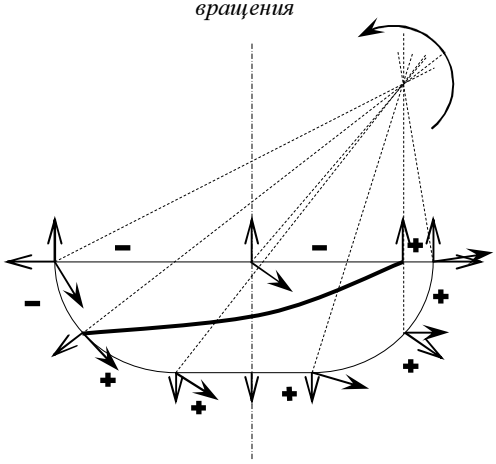
а) кривая скольжения на торе в случае пересечения осей вращения и инструмента



б) совпадение экстремумов кривой скольжения по высоте



в) пересечение оси внутри инструмента осью вращения



г) образование одной петли кривой скольжения на торе

Примечание: – отрицательная область; + – положительная область; → – вектор внешней нормали; ➔ – вектор мгновенной скорости.

Рис. 4

При построении кривой скольжения можно аппроксимировать контур тора ломаной кривой. В результате тор можно представить в виде несколь-

ких конусов разной конусности, соединенных друг с другом. В таком случае решение задачи сводится к построению кривой скольжения на поверхности конического инструмента, а также к соединению сегментов кривых между соседними элементами инструмента.

Рассмотрим случай с нахождением граничных точек на элементе инструмента в виде плоского диска. Из определения кривой скольжения [2] можно вывести следующее выражение:

$$\vec{n} \cdot \vec{d}(x, y) = 0, \quad (2)$$

где $\vec{n}$ – вектор нормали к диску; $\vec{d}(x, y)$ – вектор мгновенной скорости в точке на диске с координатами (x, y);

$$\vec{d}(x, y) = P(x, y) \cdot M, \quad (2.1)$$

где $P(x, y)$ – точка на диске с координатами (x, y); M – матрица полной мгновенной скорости.

Точку на диске можно выразить следующим образом.

Поскольку у диска высота граничных точек одна, то результатом его пересечения с плоскостью, перпендикулярной вектору скорости в точке Q, будет являться прямая в виде отрезка. Для нахождения пары граничных точек на окружностях диска необходимо решить систему уравнений (2) и уравнение окружности.

Кривая скольжения на боковой поверхности диска представляет собой один или два отрезка (рис. 5).

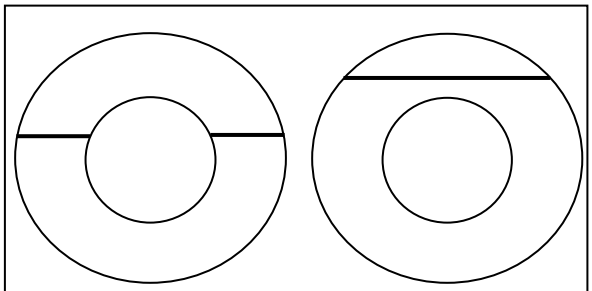


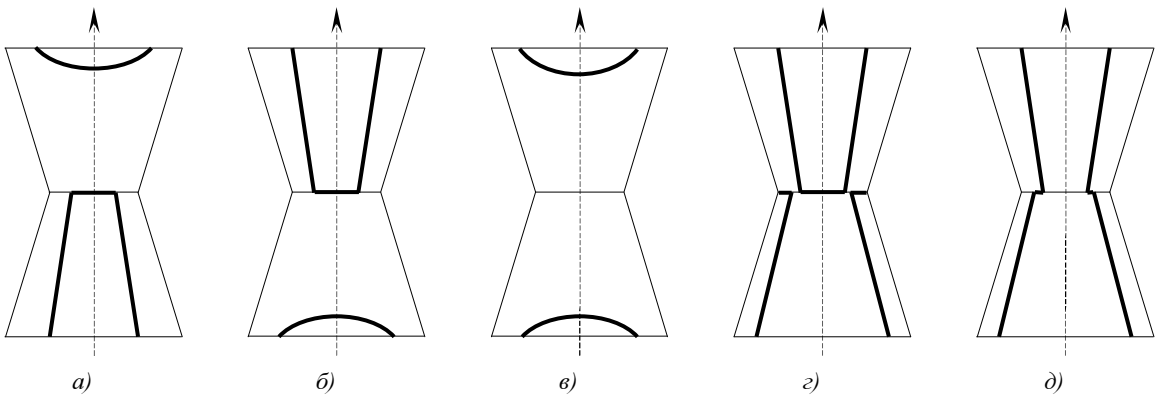
Рис. 5

Так как граничные точки на общем основании для соседних элементов инструмента могут не совпадать (рис. 6), а кривая скольжения обладает свойством непрерывности, необходим алгоритм соединения сегментов кривой скольжения. Из постановки задачи следует, что такое соединение находится исходя из соображения смены режущей поверхности на нережущую.

На основе рассмотренных способов построения кривой скольжения разработаны программы построения кривой скольжения на поверхности различного вида инструмента при многокоординатной обработке. Средой разработки выступила MS Visual Studio 2010, а в качестве языка программирования выбран C++.

Особенность реализации алгоритма заключается в возможности создания методов программы, основанных на операциях с матрицами и векторами, таких как скалярное произведение, вычисление модуля вектора, умножение вектора на матрицу и т.д. Были созданы методы для поиска точки кривой скольжения на прямой образующей конуса, метод поиска сегментов кривой скольжения на основании двух соседних элементов инструмента.

Структура данных программы должна была предполагать все возможные варианты хранения кривой скольжения различного вида. Для этого создан класс GrazingCurve (кривая на всем инструменте), содержащий массив из объектов типа GrazingCurveElemTool, соответствующий части кривой, расположенной на определенном элементе инструмента. Внутри таких объектов содержится до восьми типов сегментов, представленных массивами из опорных точек. Сегменты делятся по типу расположения: на основании элемента инструмента или на его боковой поверхности, на верхней петле или на нижней (рис. 4б, 6в), справа или слева по ходу движения инструмента, если смотреть с конца оси Oz. Такая структура данных дает возможность декомпозиции задачи для каждого элемента инструмента, имеющего определенную



а) *б)* *в)* *г)* *д)*

Вид кривой скольжения на основании: а) петля на верхнем элементе инструмента; б) петля на нижнем элементе инструмента; в) петли на обоих элементах инструмента; г) петли на основании; д) связное поэлементное соединение кривой скольжения на инструменте.

Рис. 6

геометрическую форму, сводя вычисления к более простому виду.

Созданный программный модуль добавлен в состав NCManager – симулятора обработки на станках с ЧПУ. Результаты разработок служат основой для построения поверхности, заметенной инструментом при многокоординатной обработке.

Литература

1. Eyyup Aras. Generating cutter swept envelopes in five-axis milling by two-parameter families of spheres. Computer-Aided Design. 2009. № 41, pp. 95–105.

2. Wang WP, Wang KK. Geometric modeling for swept volume of moving solids. IEEE Computer Graphics and Applications. 1986. № 6 (12), pp. 8–17.

УДК 004.823

БАЗА ЗНАНИЙ ИНФОРМАЦИОННО-УПРАВЛЯЮЩЕЙ СИСТЕМЫ СУШИЛЬНОЙ УСТАНОВКИ

С.В. Артемова, к.т.н.; А.Н. Грибков, к.т.н.
(Тамбовский государственный технический университет, GribkovAlexey@yandex.ru)

Рассмотрены вопросы разработки фреймовой базы знаний информационно-управляющей системы сушильной установки.

Ключевые слова: *информационно-управляющая система, фреймовая база знаний.*

Сушка – это широко распространенный энергоемкий процесс в промышленности, во многих случаях определяющий не только качество продукции, но и технико-экономические показатели производства в целом. Одним из методов снижения энергоресурсопотребления и повышения качества выпускаемой продукции является создание и внедрение *информационно-управляющих систем* (ИУС) для сушильных установок.

Несмотря на многообразие сушильных установок, можно выделить характерные для них особенности, которые необходимо учитывать при решении *задач оптимального управления* (ЗОУ). Как правило, в сушильные установки входит оборудование для подачи тепла, а также для осуществления движения рабочих органов. В соответствии с техническими параметрами сушильной установки и технологическим регламентом можно выделить основные режимы ее работы – пуск и сушка. В режиме пуска происходят разогрев сушильной установки и включение различных исполнительных устройств. Основные цели *оптимального управления* (ОУ) – энергосбережение и экономия топлива. В режиме сушки, самом длительном в процессе, происходит удаление влаги из материала. Целями управления являются качество конечного продукта и производительность сушильной установки.

Наиболее наукоемкий этап разработки ИУС – проектирование ее *базы знаний* (БЗ). При этом предполагается, что математическое и программное обеспечение ИУС, позволяющее решать ЗОУ в режиме реального времени, базируется на следующих подходах:

– рассмотрение моделей объектов управления на множестве состояний функционирования [1];

– анализ ЗОУ режимом пуска с использованием метода синтезирующих переменных [2];

– использование в режиме пуска алгоритмов управления многозонными объектами [3];

– использование в режиме пуска алгоритмов синтеза ОУ с учетом действующих шумов;

– применение в режиме сушки методов искусственного интеллекта, в частности, нейронных сетей для оценки влажности материала [4] и нечеткой логики для управления процессом [5];

– применение в режиме сушки (в случае многозонных сушильных установок) стратегий управления как для одной партии материала, так и для разных партий.

БЗ ИУС имеет стратифицированную иерархическую структуру, представленную множеством взаимосвязанных фреймов (рис. 1), отражающую граф поиска решения ЗОУ в пространстве состояний. Фреймы БЗ ИУС имеют слоты, содержащие не только конкретные данные, но и имена процедур, осуществляющих их обработку по заданному алгоритму. Часть фреймов включает слоты, заполнителями которых являются правила продукций.

Верхний, нулевой, уровень иерархии содержит фрейм, позволяющий проводить идентификацию состояния функционирования объекта управления, первый – идентификацию режима работы объекта управления, а второй – идентификацию цели управления. На третьем уровне располагаются фреймы для структурной и параметрической идентификации модели объекта, пригодной для решения задачи управления, на четвертом – фреймы анализа задачи управления. Пятый уровень составляют фреймы определения стратегии реализации управления, шестой – фреймы синтеза алго-

ритма управления. На седьмом уровне содержатся фреймы с соответствующими имитационными моделями, позволяющими проводить проверку синтезированного алгоритма управления.

Фрейм frAim включает фрейм frFunc, позволяющий рассчитать значения функционалов затрат энергии и топлива. Модуль анализа в зависимости от объекта управления использует фрейм анализа

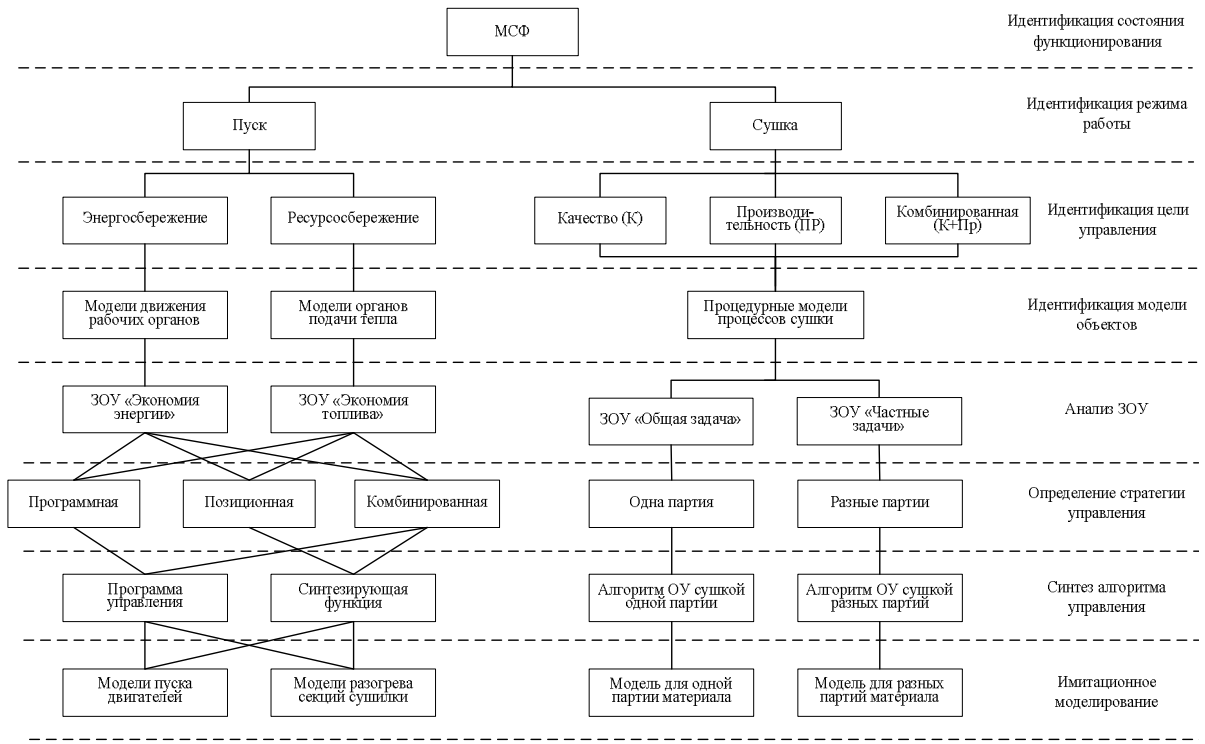


Рис. 1. Структурная схема БЗ ИУС

Иерархическая многоуровневая структура дает возможность организовать процесс приобретения и использования знаний, а также автоматизировать процесс получения оперативного решения ЗОУ. При решении задач используются следующие модули ИУС: идентификации состояния функционирования, режима, цели и модели объекта управления, постановок задач управления, анализа ЗОУ и выбора стратегии, синтеза решения и имитационного моделирования и др.

Программно фреймовая БЗ ИУС представляет собой набор классов, созданных в среде визуального программирования Borland Developer Studio 2006 на языке Object Pascal. Статические модели фрагментов структуры БЗ для режимов пуска и сушки изображены на рисунках 2 и 3 соответственно. Эти модели представлены в нотации UML в виде диаграмм классов.

Фреймы, представленные в виде классов (см. рис. 2), используются для обеспечения функциональности модулей ИУС, работающих в режиме пуска.

Модуль идентификации состояния функционирования использует фрейм frMSF, содержащий фрейм frRegimeWork, агрегирующий фрейм frAim, которым пользуется модуль целей управления. ОУ (frAnalysisOC), агрегирующий фреймы получения условий существования решения ЗОУ (frSolveExists), определения видов функций ОУ (frFunctionOC), границ областей видов функций ОУ (frAreaLimits), расчета параметров функций ОУ (frParamOC).

Модуль имитационного моделирования использует фрейм frImitationModelling, включающий фреймы моделирования так называемых белых (frWhiteNoise) и цветных (frColorNoise) шумов.

Модуль синтеза для решения задач управления сушильной установкой использует фрейм frSynthesisOC, включающий фреймы: математических моделей объектов (frModel); алгоритмов управления (frAlg), от которого наследуются фреймы frAlgNoise и frAlgNoNoise; множества состояний функционирования (frMSF).

Фрейм массива реквизитов (frMasReqv) агрегирует фрейм моделей (frModel) и связан отношением зависимости с фреймом расчета синтезирующих переменных frSynthVarEng.

Фреймы БЗ, представленные на рисунке 3, используются для обеспечения функциональности программных модулей ИУС, работающих в режиме сушки.

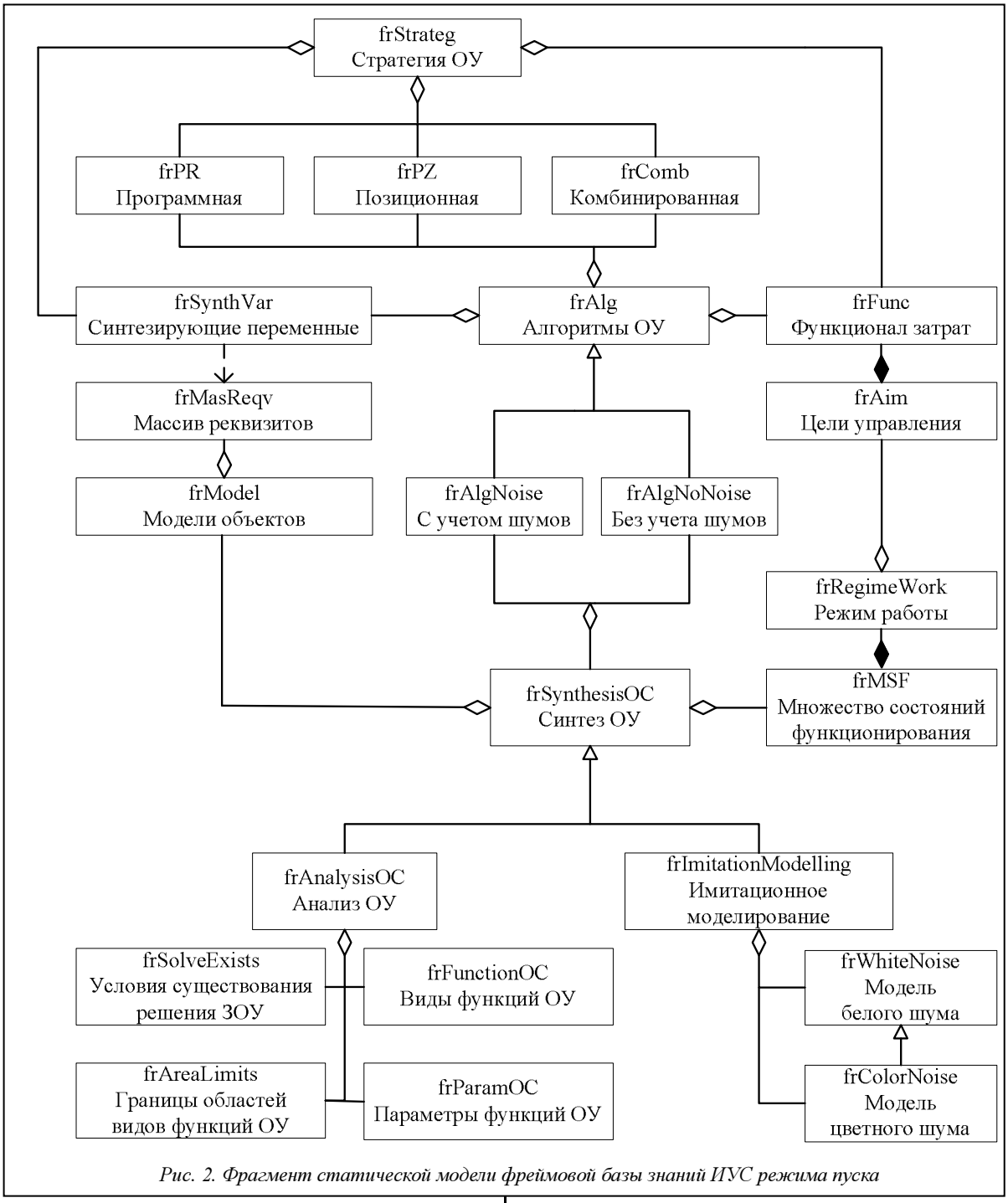


Рис. 2. Фрагмент статической модели фреймовой базы знаний ИУС режима пуска

Модуль идентификации состояния функционирования объекта управления использует фрейм frMSF, но при этом работают слоты, соответствующие режиму сушки. Фрейм frMSF содержит фреймы frRegimeWork и frIdentif, агрегирующий фрейм frAim, которым пользуется модуль целей управления.

Модуль идентификации ситуации использует фрейм frIdentif. Он позволяет в зависимости от состояния функционирования, режима работы выбрать цель управления, согласно которой определяется класс решаемой задачи (frKlassTask). Этот фрейм агрегирует фреймы frStrategy и frFuncPr. Фрейм frStrategy содержит три фрейма стратегий управления – frProdCommonTask, frProdSpecialTask1, frProdSpecialTask2, – соответствующие общей задаче управления и двум частным задачам.

Модуль логического вывода, определяющий управляющее воздействие, использует фрейм процедурных знаний frFuzzyLogic, связанный отношением зависимости с фреймами frProdCommonTask, frProdSpecialTask1 и frProdSpecialTask2, знания в которых представлены в виде продукционных правил. Фрейм frFuzzyLogic также связан

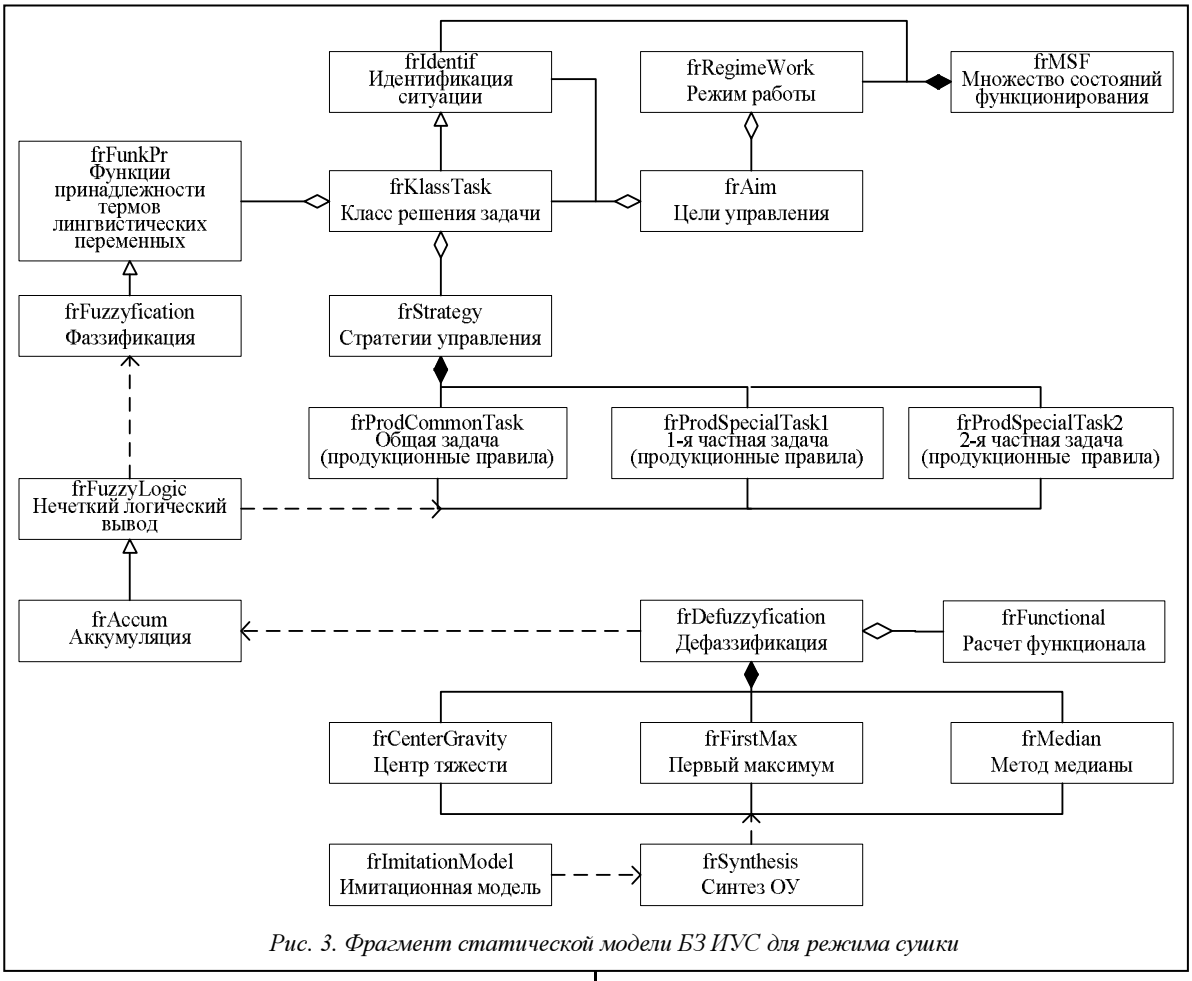


Рис. 3. Фрагмент статической модели БЗ ИУС для режима сушки

отношением зависимости с фреймом frFuzzyfication, который наследует знания фрейма frFuncPr, связанного отношением зависимости с frPrModel. Фрейм frAccumulation наследует знания фрейма frFuzzyLogic, которые использует фрейм frDefuzzification. В состав фрейма frDefuzzyfication входят фреймы, отражающие процедурные знания следующих методов: центр тяжести (frCenterGravity), первый максимум (frFirstMax) и медианы (frMedian). Фрейм frFunctional включен во фрейм frDefuzzyfication.

Модуль синтеза ОУ использует фрейм frSynthesis, который использует знания фрейма frDefuzzyfication.

Модуль имитационного моделирования использует фрейм frImitationModel, связанный отношением зависимости с фреймом frSynthesis.

Созданная структура БЗ обеспечивает оперативную работу модулей алгоритмического обеспечения ИУС, что дает возможность синтезировать ОУ сушильной установкой в реальном времени.

БЗ ИУС позволяет решать следующие задачи: энергосберегающий разогрев сушильной установки, энергосберегающий пуск электродвигателей, достижение требуемого качества высушиваемого материала при максимальной производительности процесса сушки в режиме реального времени. Экономия энергоресурсов при ОУ разогревом сушильной установки составила 6,2 % по сравнению с традиционным. В целом же для всей сушильной установки экономия энергоресурсов в динамических режимах составляет 5–10 %. Применение ИУС позволило увеличить вероятность выхода качественной продукции до 0,98, а также повысить производительность процессов сушки на 5 %.

Литература

1. Артемова С.В., Грибков А.Н. Математическая модель многосекционной сушильной установки на множестве состояний функционирования // Вестн. ТГТУ. 2002. Т. 12. № 4. С. 969–974.

2. Расширенный анализ задач оптимального управления / Артемова С.В. [и др.] // Информационные процессы и управление. 2006. № 1. URL: http://www.tstu.ru/ipu/2006-1/002.pdf (дата обращения: 14.07.11).

3. Грибков А.Н., Артемова С.В. Алгоритм ресурсосберегающего управления динамическими режимами многосекционных сушильных установок // Изв. Томск. политехнич. ун-та. 2008. Т. 313. № 4. Томск: Изд-во ТПУ. С. 48–50.

4. Артемова С.В., Грибков А.Н., Ерышов А.Е. Информационная система мониторинга влажности материалов в процессе сушки // Приборы и системы. Управление, контроль, диагностика. 2009. № 7. С.46–50.

5. Артемова С.В., Грибков А.Н. Система мониторинга процесса сушки с интеллектуальными датчиками влажности // Датчики и системы. 2009. № 3. С. 27–30.

УДК 618.518

ПРОГРАММНЫЙ КОМПЛЕКС УПРАВЛЕНИЯ ИННОВАЦИОННО-ПРОИЗВОДСТВЕННОЙ СИСТЕМОЙ

В.Г. Матвейкин, д.т.н.; Б.С. Дмитриевский, к.т.н.; И.С. Панченко
(Тамбовский государственный технический университет, irina-pnk@mail.ru)

Описаны задача управления и модули системы управления инновационно-производственной системой. Приведены структура программного комплекса и его программная реализация.

Ключевые слова: *программный комплекс, инновационно-производственная система, граф состояний функционирования.*

Высокая доля ошибок оперативного персонала при принятии решений делает актуальной задачу разработки системы автоматизированного управления *инновационно-производственной системой* (ИПС). Для снижения риска опасности, связанного с человеческим фактором, необходимо автоматизировать максимальное количество процессов, в которых он задействован [1].

Под ИПС будем понимать производственную систему, в которой наряду с технологическими и информационными потоками циркулируют интеллектуальные потоки, позволяющие получить конкурентоспособный целевой продукт нового качества.

Представим векторы входных параметров ИПС.

$U_{\text{тех}}$ – характеризует технологические потоки. Вектор состояний *химико-технологической системы* (ХТС) $s=\langle s_1\rangle$, $s\in S$, определяет ее функционирование в каждый момент времени, которое зависит от технологических параметров s_1 – показателя общей эффективности оборудования, фактической производственной мощности и других факторов.

$U_{\text{инф}}$ – характеризует информационные потоки, которые определяются наличием технологических расчетов, регламентов и пр. Вектор состояний производственной системы $s=\langle s_1, s_2\rangle$, $s\in S$, определяет функционирование системы, которое зависит от технологических (s_1) и технико-экономических (s_2) параметров.

$U_{\text{инт}}$ – характеризует интеллектуальные потоки, которые определяются наличием моделей, алгоритмов, программ, совокупностью производственных и общечеловеческих навыков, знаний.

Интеллектуальные ресурсы включают в себя вектор стратегических целей инновационного развития, параметры, характеризующие инновационно-технологический уровень ХТС и продукта, систему сбалансированных показателей.

Цели разработки ИПС: повышение инвестиционной привлекательности, выделение новых технологий, создание новых продуктов, стимулирование развития инноваций.

Инновационно-технологический уровень ХТС и продукта характеризуют следующие параметры: коэффициент рентабельности инноваций, отдача от НИОКР, индекс рентабельности инноваций, общая результативность инноваций, надежность, длительность процесса разработки нового продукта, подготовки производства нового продукта, производственного цикла нового продукта.

Система сбалансированных показателей характеризует оценку эффективности ИПС, зависящей от состояния ИПС.

Вектор состояний ИПС $s=\langle s_1, s_2, s_3\rangle$, $s\in S$, определяет функционирование системы, которое зависит от технологических (s_1), технико-экономических (s_2) и инновационных (s_3) параметров.

В отличие от традиционной производственной системы ИПС должна обеспечивать трансформацию инфраструктуры предприятия для повышения конкурентоспособности и закрепления на новых рынках путем совершенствования имеющихся продуктов или создания принципиально нового продукта.

Общая задача управления ИПС состоит в обеспечении оперативного управления распределением ресурсов для оптимального функционирования ИПС в соответствии со стратегическими целями инновационного развития для получения конкурентоспособной продукции.

Отличительная особенность задачи состоит в согласовании стратегических целей с целями оперативного управления.

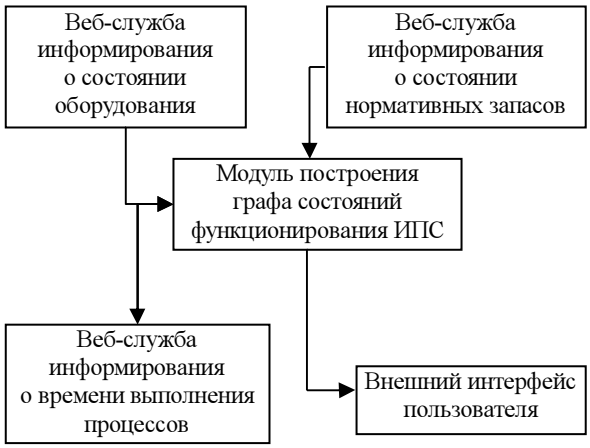
Для ее решения разработан программный комплекс (структуру см. на рисунке), функционирующий следующим образом.

Веб-служба информирования о состоянии оборудования предоставляет текущие сведения о состоянии оборудования: в ремонте, свободно/готово к эксплуатации, поступили новые единицы оборудования.

Веб-служба информирования о времени выполнения процессов предоставляет сведения о структуре производимой продукции, о технологии ее изготовления, а также формирует список процессов для изготовления продукции.

Веб-служба информирования о состоянии нормативных запасов предоставляет информацию о наличии того или иного вида запасов и о процессах, в которых они используются.

Модуль построения графа состояний функционирования ИПС решает задачи построения перво-



Структура программного комплекса

начального (исходного) графа состояний функционирования ИПС, построения графа состояний функционирования ИПС с учетом переналадок и дефекта ресурса, построения графа состояний функционирования ИПС с учетом ограничений на использование непрерывных ресурсов, оперативного контроля состояний функционирования ИПС.

Первая задача.

Для различных исходных данных (количество аппаратов, информационные и интеллектуальные ресурсы) построить граф состояний ИПС и выбрать такой порядок распределения исходных ресурсов в соответствии с построенным графом, который приводил бы к минимальному времени окончания процессов.

Вследствие того, что необходимо учитывать множество ограничений, отражающих реальные производственные условия, поиск точного решения затруднен. Поэтому был предложен эвристический алгоритм, позволяющий строить граф состояний функционирования ИПС, удовлетворяющий ограничениям и позволяющий учитывать приоритетность выполнения конкретных ХТС, при этом в процессе расчета эксперты могут вносить корректировки.

Для работы данного алгоритма требуются следующие входные данные: перечень аппаратов I; множество ХТС J, которое необходимо выполнить; множество процессов L, из которых состоят ХТС; множество информационных ресурсов D, множество интеллектуальных ресурсов M; булева матрица Y, где $y_{ijld} \in \{0, 1\}$ – булева матрица, определяющая выполнение l-го процесса ХТС j на аппарате i при наличии информационного ресурса d и интеллектуального ресурса m.

Последовательно составляем список процессов, которые на данном шаге могут быть задействованы в построении графа состояний функционирования ИПС. Необходимое для этого условие – либо у них не было предшественников, либо их предшественники уже входили в граф. Затем распределяем ХТС, которые следует выполнить, и процессы, составляющие ХТС, по аппаратам. Все процессы делим на уникальные, которые не могут быть выполнены другим аппаратом, и неуникальные. Проверяем наличие соответствующих информационных и интеллектуальных ресурсов.

В дальнейшем уникальность/неуникальность процесса используется для ранжирования приоритетов выполнения процессов.

Обозначим время выполнения на i-м аппарате распределенных на него первых l процессов j-го ХТС f^{l}_{ijdm} и приравняем его к нулю $f^{0}_{ijdm} = 0, i \in I$.

В качестве критерия оптимизации в данном алгоритме используется минимальное время выполнения всех ХТС. Время выполнения операций рассчитывается по рекуррентному соотношению.

В процессе построения графа ведем учет количества процессов, выполненных на каждом аппарате в конкретной ХТС. На каждом этапе выбираем менее загруженный аппарат, то есть, если время выполнения процесса соответствующей ХТС не равно нулю и есть свободный аппарат для его осуществления, выбираем аппарат, который выполнит данный процесс быстрее, а также находим наилучшее решение для распределенных (l-1) процессов предыдущего этапа: $f^{l-1}_{ijdm} \neq 0$ и $y_{ijdml-1} = 1$, то $\max\_f = \max\left(\max\_f, f^{l-1}_{ijdm}\right)$, $\min\_f = \min\left(\min\_f, t_{ijdml}\right)$, $\left(\min\_f, t_{ijdml}\right)$.

Определяем минимальное время выполнения процесса на конкретном аппарате: $f^{l}_{\min\_i,j} = \min\left(\max\_f, f^{l-1}_{\min\_i,jdm} + \min\_f\right)$ и порядок загрузки выбранного аппарата конкретным процессом. После этого рассчитываем время выполнения ХТС и соответствующих им процессов на конкретном аппарате и суммарное время выполнения ХТС, задаваемое как время выполнения самой продолжительной ХТС: f^{l}_{ijdm}, f^{l}_{ijdm}.

В результате работы алгоритма получаем граф состояний функционирования ИПС, оптимальный по данному критерию.

Вторая задача.

Перестроить граф состояний функционирования ИПС с учетом ограничений, накладываемых графиком планово-предупредительных ремонтов и дефектом ресурса, а также аварийных ремонтов. Сбор информации о количестве, периодичности и характере ремонтов аппаратов начинается с начала их эксплуатации; с учетом данных сведений происходит корректировка графика планово-предупредительного ремонта.

Для решения задачи необходимо построить все варианты распределения ресурсов и выбрать тот, у которого время окончания всех процессов минимально ($T_{\min}$).

Все выполненные процессы в графе обозначим x – двоичным N-разрядным числом, номер разряда

двоичного числа x соответствует номеру процесса. Если процесс выполнен, в соответствующем разряде стоит 1, если нет – 0.

Обозначим $R(x)$ граничные процессы (все невыполненные процессы, имеющие связь хотя бы с одним из выполненных процессов); $U(x)$ – допустимые процессы (процессы, принадлежащие $R(x)$, которые еще не выполнены, но вполне выполнимы); $P(x)$ – фронт выполненных процессов (все выполненные процессы, связанные хотя бы с одним из граничных процессов); $\gamma(\delta_i)$ – множество процессов, непосредственно предшествующих данному процессу δ_i.

Между этими множествами можно записать следующие соотношения:

$$P(x) = R(x_p - x),\ R(x) = P(x_p - x),$$

$$U(x) \subset R(x),\ U(x_p - x) \subset R(x_p - x).$$

Одному значению x может соответствовать различное время выполнения этих процессов в зависимости от имеющихся интеллектуальных и информационных ресурсов, его будем обозначать $t_i^B(x)$, (i=1, 2, …, N), и разное время освобождения аппаратов $T_j(x)$, (j=1, 2, …, R). Таким образом, каждому x можно поставить в соответствие векторы $t^B(x)$ и $T(x)$. Следовательно, совокупность $\{x, t^B(x), T(x)\}$ – состояние системы S, которое будет точкой в пространстве размерности N+R+1.

Из первоначального состояния $S_1\{x, t^B(x), T(x)\}$ после выполнения одного процесса $\delta_i \subset U(x)$ j-м аппаратом ИПС переходит в новое состояние $S_2\{y, t^B(y), T(y)\}$, где $y=x+\delta_i$.

При подобном переходе учитываются ограничения.

Неотрицательность действительных переменных: t_{ijld}, z_{ijld}, $p_r \geq 0$, где z_{ijld} – длительность перерыва после окончания процесса i на аппарате j l-м интеллектуальным ресурсом при наличии d-го информационного ресурса; z_{ijld}, b_{ijld} – моменты начала и окончания выполнения процесса i на аппарате j; p_r – длительность простоя j-го аппарата.

При этом $z_{ijld} = p_{r_{ijld-1}} + z_{ijld-1}$, $b_{ijld} = z_{ijld} + t_{ijld}$, $b_{k0} = 0, z_{k1} = z_{k0}$ – логические условия: правильность нумераций процессов, однооператорность, однопроцессорность, одноаппаратность, выполнимость всех процессов $c_{jii+1ld} \leq z_{jild}$, где z_{ijld} – длительность перерыва после окончания процесса i: $b_{ijld} \leq z_{i+1,jld}$.

Для вычисления $t^B(y)$ и $T(y)$ используются следующие соотношения:

– время возможного начала δ_i процесса t_i^H равно максимальному времени выполнения предшествующих процессов:

$$t_i^H(y) = \max_{\delta_i}(t_k^B(x)),\ (\delta_k \subset \gamma(\delta_i));$$

– время выполнения δ_i процесса i-м аппаратом: $t_i^B(y) = \max(T_j(x), t_i^H(y)) + \tau_{ij}$;

– время освобождения i-го аппарата после выполнения j-го процесса: $T_j(y) = t_i^B(y)$.

На основе взаимосвязи компонент векторов можно построить все варианты распределения ресурсов по процессам, которые будем называть графом состояний функционирования ИПС с учетом переналадок.

Для построения такого графа состояний, который удовлетворял бы критерию минимума окончания всех процессов, необходимо определить значения b_{ijld} и z_{ijld}.

Расчет дефекта ресурса выполним по следующей формуле: $Df = \sum_{f=1}^{F}(R - |u_f|) = FR - N$, где u_f – процессы, выбранные к выполнению на f+1-м шаге; $|u_f|$ – число единиц процессов.

Начальному состоянию x_0=(0…0), $T(x_0)$=(0…0) соответствует некоторое множество допустимых процессов $U(x_0)$, и каждый допустимый процесс можно выполнить R различными ресурсами согласно матрице соответствия процесса аппарату.

В результате решения данной задачи получаем граф состояний функционирования с учетом переналадок и дефекта ресурса, оптимальный по данному критерию.

Третья задача.

Перестроить граф состояний функционирования ИПС с учетом ограничений на использование непрерывных ресурсов в каждый момент времени, интегральных ограничений за весь период выполнения плана и решения задачи управления качеством ХТС, которая состоит из следующих задач:

– контроль качества и проверка его соответствия требованиям;

– контроль построения графа состояний функционирования ИПС и проверка его соблюдения или нарушения;

– установление взаимосвязи между графом состояний функционирования ИПС и качеством ХТС;

– определение графа состояний функционирования ИПС, обеспечивающего выполнение ХТС;

– управление графом состояний функционирования в процессе реализации ХТС с целью возможного повышения качества ХТС или снижения затрат на технологию по отношению к их средним значениям.

Качество ХТС конкретизирует набор показателей сбалансированной системы показателей [2]. Возможность объективного упорядочения ХТС по качеству имеет принципиальное значение для ее сравнения, назначения стоимости анализа недостатков и поиска путей совершенствования графа состояний функционирования ИПС.

Четвертая задача.

В режиме реального времени в модуле «Оперативный контроль состояний функционирования ИПС» необходимо видеть, где находится тот или иной вид ресурсов и как он будет размещен в конкретную рабочую смену. В ходе решения данной задачи автоматизируются процессы описания и учета оборудования, планирования ремонтных работ, их выполнения и анализа результатов, определения и обеспечения материально-техническими ресурсами.

Для решения задачи оперативного контроля состояний функционирования ИПС декомпозируем ее на следующие подзадачи.

1. Управление простоями, вынужденными и регламентными ремонтами, сбоями в снабжении запчастями, расходными материалами.
2. Упорядочение оборудования и ведение истории оборудования.
3. Управление расходами на поддержание оборудования в рабочем состоянии без снижения качества его эксплуатации.
4. Оценка по заданным параметрам состояния оборудования, исполняемых с ним и над ним операций, качества их исполнения, понесенных затрат.
5. Управление затратами на техническое обеспечение и ремонт.
6. Поддержка принятия решений при планировании, прогнозировании, подготовке и выполнении технического обслуживания и ремонта, модернизации технологических линий.
7. Обеспечение безопасности работ и соответствие техническим условиям.

Для визуализации хода производственного процесса используются различные варианты графиков и таблиц.

При разработке программного комплекса использовался язык программирования C#, на котором написаны базовые классы и реализован алгоритм построения графа состояний функционирования ИПС.

В качестве host-системы выбрана Microsoft Windows Server 2003 Standard Edition, обеспечивающая повышенный уровень безопасности при обмене данными по сети Интернет, а также высокий уровень надежности и масштабируемости.

Создана отдельная библиотека классов построения графа состояний функционирования ИПС. Каждый из этих классов можно использовать отдельно для решения смежных задач, а именно:

- scheduleEngine – управление процессом формирования графа состояний функционирования ИПС;
- OperationClass – расчет динамических характеристик процессов: наименование процесса, время начала и окончания в зависимости от выбранного оборудования в процессе построения графа состояний функционирования ИПС;
- Device – расчет динамических характеристик оборудования при построении графа состояний функционирования ИПС: время выполнения того или иного процесса на данном оборудовании, аварийного или планово-предупредительного ремонта и т.д.;
- EnqueueOperation – распределение уникальных процессов по оборудованию, которые могут выполняться только на конкретном оборудовании;
- OperationClassList – формирование двух списков процессов: уникальные и неуникальные процессы, которые необходимо выполнить;
- DeviceList – формирование списка готового к эксплуатации оборудования с указанием процессов, которые могут быть на нем выполнены;
- FindSchedule – формирование графа состояний функционирования ИПС в соответствии с данными, предоставляемыми классами OperationClassList и DeviceList.

Интерфейс пользователя реализован при помощи ASP.NET 3.5 SP1, поскольку ASP.NET является частью платформы Microsoft .NET, в которую уже встроено множество технологий для интеграции различных приложений и информационных систем. Такое многообразие решений дает возможность выбрать для каждого случая оптимальную технологию, обеспечивающую наилучшую производительность, безопасность и масштабируемость. Расширяемый набор элементов управления и библиотек классов позволяет быстрее разрабатывать приложения.

ASP.NET-страница создает объекты классов и после инициализации списков оборудования и процессов выводит диаграмму Ганта по рассчитанному графу состояний функционирования ИПС. Внешний интерфейс взаимодействует с модулями программного комплекса для получения текущей информации о ходе производственного процесса. Для оперативного управления корректировкой графа состояний функционирования ИПС используется веб-служба, в которую поступает информация как о новом оборудовании, так и о вышедшем из строя, а также веб-служба, обновляющая информацию о времени выполнения процессов на том или ином оборудовании.

С помощью технологий LINQ2SQL и ASP.NET DynamicData создаются веб-страницы интерфейса к БД (редактирование таблиц).

В качестве СУБД выбрана SqlServer 2008 ExpressEdition; среда разработки – MS Visual Studio 2008.

К программным комплексам для управления сложными системами предъявляются высокие технические требования. Их реализация при сохранении приемлемой производительности конеч-

ного комплекса – сложная задача, решая которую, необходимо учитывать предоставляемую конечной архитектурой возможность легкого добавления в систему управления ИПС новых методов и алгоритмов.

Разработанный комплекс дает возможность добавлять новые алгоритмы построения графа состояний функционирования ИПС к уже существующим благодаря реализации алгоритмов за пределами самой платформы.

При выполнении алгоритма построения графа состояний функционирования ИПС управление полностью передается основному вычислительному потоку приложения, что накладывает жесткие требования безопасности и производительности при реализации.

Программный комплекс предназначен для оперативной диагностики состояний ИПС, для помощи в обнаружении отказов, а также поиска неисправностей в технологическом процессе. После ввода требуемых входных данных ЛПР (обслуживающий персонал) получает построенный граф состояний функционирования ИПС. При обнаружении отказа/неисправности ЛПР вносит корректировки в построенный граф. Кроме того, ему предлагаются альтернативные варианты решения возникшей проблемы (возможные варианты замены оборудования, выполнение другой операции или заказа), предоставление справочной информации о наличных ресурсах для возможного выполнения других заказов.

Для принятия своевременных мер по предупреждению отказа/неисправности система управления ИПС создает граф состояний функционирования с возможностью динамического отображения степени неисправности.

Программный комплекс имеет интуитивно понятный интерфейс, реализованный в виде набора форм для заполнения, на которых приведена полезная информация, а также включает в себя главное меню управления системой.

Литература

1. Управление эксплуатацией основных фондов. Матвейкин В.Г. [и др.] / Деп. в ВИНИТИ № 177-В2009 // Депонир. науч. раб. 2009. № 5.

2. Построение системы показателей для оценки эффективности наукоемкой производственной системы. Матвейкин В.Г. [и др.] // Вестн. ТГТУ. 2009. Том 15. № 2. С. 278–284.

3. Кафаров В.В., Мешалкин В.П. Анализ и синтез химико-технологических систем: учебник для вузов. М.: Химия, 1991. 432 с.

УДК 519.6

АНАЛИЗ НАПРЯЖЕННО-ДЕФОРМИРОВАННОГО СОСТОЯНИЯ В НЕОДНОРОДНЫХ КОНСТРУКЦИЯХ

(Работа выполнена при поддержке грантов РФФИ № 09-01-00523, № 10-01-00121-а и Программ Президиума РАН №№ 15, 17)

И.Н. Кандоба, к.ф.-м.н. (Институт математики и механики УрО РАН, г. Екатеринбург, kandoba@imm.uran.ru);

Л.Ф. Спевак, к.т.н. (Институт машиноведения УрО РАН, г. Екатеринбург, lfs@imach.uran.ru);

О.С. Тарико (Уральская государственная медицинская академия, г. Екатеринбург)

Представлены методы анализа напряженно-деформированного состояния неоднородной, состоящей из конечного числа упругих однородных областей конструкции, подверженной заданной нагрузке. Расчет напряжений и деформаций производится на основе метода граничных элементов с использованием формул аналитического интегрирования. Разработанные алгоритмы реализованы в специализированном пакете программ, предназначенном для решения широкого круга прикладных задач.

***Ключевые слова**: упругость, метод граничных элементов, неоднородная конструкция, контактная граница, оптимальная форма.*

Во многих содержательных прикладных задачах в строительной механике, машиностроении, медицине и в других областях достаточно актуальной является задача анализа *напряженно-деформированного состояния* (НДС) некоторой конструкции, подверженной воздействию внешней статической нагрузки. В большинстве случаев такая конструкция представляет собой объединение конечного числа упругих однородных областей, обладающих различными механическими свойствами. Как правило, одна часть внешней границы жестко закреплена, а другая испытывает заданную статическую нагрузку. В ряде практически важных случаев особый интерес представляет распределение напряжений и деформаций, возникающих на контактных границах, образующих эту конструкцию областей.

Математическая модель задачи

Постановка плоской задачи теории упругости. Для плоской кусочно-однородной области Ω,

на внешней границе которой заданы некоторые нагрузки или перемещения, рассматривается статическая задача теории упругости, заключающаяся в определении в каждой из составляющих область Ω зон $\Omega^{(1)}$, $\Omega^{(2)}$, …, $\Omega^{(n)}$ вектора перемещений u_i, тензора деформаций ε_{ij} и тензора напряжений σ_{ij}, которые в рассматриваемой зоне удовлетворяют системе уравнений

$$\sigma_{ij,i}=0, \quad (1)$$

$$\varepsilon_{ij}=\frac{1}{2}\left(u_{j,i}+u_{i,j}\right), \quad (2)$$

$$\sigma_{ij}=2\mu\varepsilon_{ij}+\frac{2\mu\nu}{1-2\nu}\varepsilon_{kk}\delta_{ij} \quad (3)$$

и заданным на внешней границе $S=S_1\cup S_2$ области Ω граничным условиям

$$S_1:\sigma_{ij}n_j=f_i=f_i^*,\ S_2:u_i=u_i^*. \quad (4)$$

Здесь по повторяющемуся индексу производится суммирование от 1 до 2; $u_{i,j}=\partial u_i/\partial x_j$; μ – модуль упругости при сдвиге; ν – коэффициент Пуассона; δ_{ij} – символ Кронекера; n_i – вектор нормали к границе зоны. Значения модуля упругости и коэффициента Пуассона для разных зон различные. На внутренних границах, являющихся границами между зонами, принимается условие непрерывности перемещений и напряжений.

Метод граничных элементов для неоднородной области. Согласно данному методу [1], внутри каждой зоны $\Omega^{(i)}$ выполняется тождество Сомильяны:

$$u_i(\xi)=\int_{S^{(i)}}\left[f_j(x)u_{ij}^*(\xi,x)-u_j(x)f_{ij}^*(\xi,x)\right]dS(x), \quad (5)$$

где $S^{(i)}$ – граница зоны $\Omega^{(i)}$; ξ – внутренняя точка $\Omega^{(i)}$; функции влияния $u_{ij}^*(\xi,x)$ и $f_{ij}^*(\xi,x)$ для двухмерной задачи теории упругости имеют следующий вид:

$$u_{ij}^*(\xi,x)=c_1\left[c_2\ln(r)\delta_{ij}-r_ir_j\right], \quad (6)$$

$$f_{ij}^*(\xi,x)=c_3\left[\left(c_4\delta_{ij}+2r_ir_j\right)\frac{\partial r}{\partial n}-c_4\left(r_in_j-r_jn_i\right)\right]. \quad (7)$$

Здесь $c_1=-\dfrac{1}{8\pi(1-\nu)\mu}$; $c_2=(3-4\nu)$; $c_3=-\dfrac{1}{4\pi(1-\nu)r}$; $c_4=(1-2\nu)$; $r=r(\xi, x)$ – расстояние между точками x и ξ; $r=\sqrt{r_i\cdot r_i}$; $r_i=x_i-\xi_i$; $r_i=\dfrac{r_i}{r}$.

Выражение (5) с учетом уравнений (2) и (3) дает непрерывное решение задачи теории упругости в области $\Omega^{(i)}$, если найдены компоненты векторов поверхностного напряжения $f_i(x)$ и перемещения $u_i(x)$ на границе $S^{(i)}$. Частично эти функции определяются из граничных условий (4).

Предельный переход $\xi\to x_0$, $x_0\in S^{(i)}$, приводит к граничному интегральному уравнению для каждой зоны:

$$c_{ij}(x_0)u_j(x_0)=\int_{S^{(i)}}\left[f_j(x)u_{ij}^*(x_0,x)-u_j(x)f_{ij}^*(x_0,x)\right]dS(x). \quad (8)$$

Здесь $c_{ij}(x_0)=\delta_{ij}+\alpha_{ij}(x_0)$,

$$\alpha_{11}(x_0)=-1-c_3\left(\omega(c_4+1)+\frac{\sin 2\omega}{2}\right),$$

$$\alpha_{22}(x_0)=-1-c_3\left(\omega(c_4+1)-\frac{\sin 2\omega}{2}\right),$$

$$\alpha_{12}(x_0)=\alpha_{21}(x_0)=-c_3\sin^2\omega, \quad (9)$$

где ω – величина угла внутри деформируемой области, образованного в граничной точке x_0. Когда точка x_0 лежит на гладкой поверхности, $c_{ij}(x_0)=\dfrac{\delta_{ij}}{2}$.

Граничное интегральное уравнение позволяет определить недостающие значения поверхностных перемещений и напряжений. В однородной области (одной зоне) решение уравнений (8) сводится к разбиению границы S на конечное число элементов, построению интерполяционных функций для неизвестных значений $u_i(x)$, $f_i(x)$ и решению получающейся из (8) системы алгебраических уравнений относительно параметров интерполяционных функций. В простейшем случае можно считать компоненты векторов поверхностного напряжения и перемещения постоянными на каждом прямолинейном элементе. Тогда уравнение (8) составляется для единственного узла на каждом элементе, в результате чего получается следующая система уравнений:

$$\frac{1}{2}u_i^{(p)}=\sum_{q=1}^{N}\left(f_j^{(q)}\int_{e^{(q)}}u_{ij}^*\left(x^{(p)},x\right)dS(x)-u_j^{(q)}\int_{e^{(q)}}f_{ij}^*\left(x^{(p)},x\right)dS(x)\right),$$

$i=1, 2$; $p=1, \ldots, N$, (10)

где N – число элементов; $e^{(k)}$ – граничные элементы; $u_i^{(k)}$, $f_i^{(k)}$ – значения компонентов векторов перемещения и поверхностного напряжения на элементе $e^{(k)}$. В неоднородной области, где граничные условия (4) заданы только на внешней границе всей области Ω, систему уравнений, объединяющую системы вида (10), построенные для каждой из зон, необходимо дополнить условиями непрерывности перемещений и напряжений на общих границах зон. Решение полученной системы линейных алгебраических уравнений даст значения поверхностных напряжений и перемещений на всех границах $S^{(i)}$, что позволит определить из соотношений (5), (2), (3) НДС внутри зон $\Omega^{(i)}$.

Для реализации описанного алгоритма необходимо вычислять входящие в уравнения (5) и (10) криволинейные интегралы по граничным элементам от компонент функций влияния. Стандартный подход к решению предполагает численное вычисление этих интегралов в каждой конкретной задаче. Более универсальным подходом является

получение аналитических формул для точного вычисления всех необходимых при решении задачи интегралов. Получение таких формул позволяет включить их в алгоритм, тем самым повысив его универсальность. Кроме того, аналитическое интегрирование по сравнению с численным имеет очевидные преимущества по точности и скорости вычислений. В данной работе использованы аналитические формулы, полученные в [2]. Их применение ускоряет счет в 5–10 раз, а выигрыш в точности соответствует погрешности интерполяции, применяемой при численном интегрировании. Сопоставление полученных решений различных задач деформирования для плоских областей с известными аналитическими решениями, а также с решениями, полученными в ANSYS, показало высокую эффективность разработанных алгоритмов [2].

Применение предложенной методики позволяет во многом избежать традиционных вычислительных трудностей, возникающих при численной реализации операций интегрирования и дифференцирования, необходимых в рассматриваемой задаче для вычисления компонент (касательной и нормальной) напряжений σ_{ij} и перемещений u_i в точках сечения конструкции.

Пакет программ для анализа НДС плоской неоднородной области

На примере задачи анализа НДС восстановленного при помощи цельнокерамического микропротеза депульпированного зуба покажем эффективность разработанных алгоритмов вычисления значений компонент тензоров напряжений и деформаций на контактных границах, образующих неоднородную конструкцию однородных упругих сегментов, приведем результаты математического и численного моделирования с использованием реальных данных.

Для проведения численных экспериментов разработан специализированный пакет прикладных программ, основными модулями которого являются

– модуль обмена данными с внешними файлами;

– графический редактор – построение геометрической модели плоского сечения неоднородной конструкции;

– модуль численного решения плоской задачи упругости – вычисление НДС плоского сечения конструкции;

– модуль анализа НДС неоднородной конструкции – численное решение ряда прикладных задач с использованием данных об НДС конструкции.

Пакет обладает эргономичным и дружественным пользовательским интерфейсом, его функциональные возможности позволяют определять и анализировать НДС достаточно сложно геометрически организованных плоских конструкций (см. рис. 1).

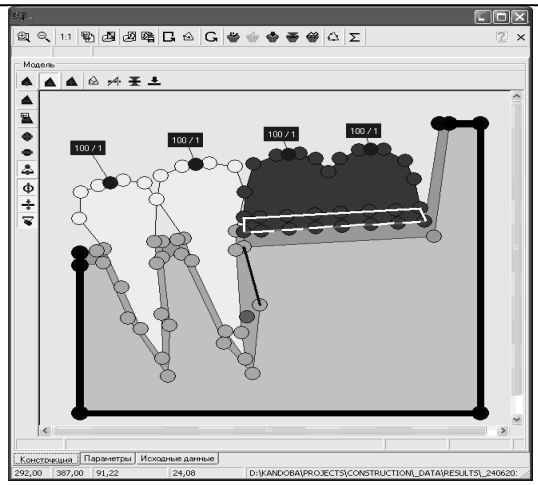


Рис. 1. Фрагмент плоского сечения зубочелюстной системы. Редактирование зоны периодонта – перемещение вершины ломаной

На рисунке 2 представлена общая блок-схема пакета.

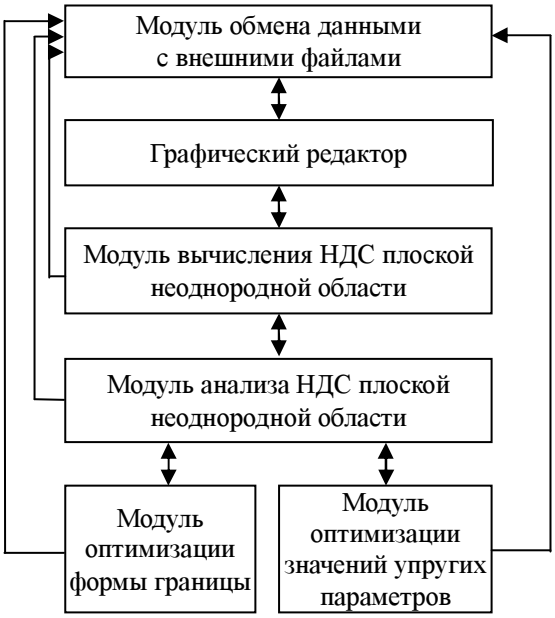


Рис. 2. Структура пакета

Первый модуль предназначен для загрузки во внутренние структуры системы данных о геометрической модели, свойствах материалов и внешней статической нагрузке из внешних файлов, а также сохранения в них результатов работы всех остальных модулей пакета. Все данные во внешних файлах хранятся в текстовом формате. В этом модуле реализованы возможности формирования и редактирования списка материалов, таких как эмаль, дентин, керамика с выбором цвета и пр.

Графический редактор предназначен для построения новой и редактирования существующей геометрической модели плоской неоднородной области, определения внешней статической нагрузки (точек приложения, направлений и значений внешних сил), указания закрепленной части внешней границы области, формирования дополнительных структур данных (вариабельные части контактных границ образующих область зон, анализируемая зона области). Инструментальные средства этого модуля позволяют пользователю в различных режимах строить замкнутые ломаные линии, определяющие границы соответствующих зон (рис. 1). В частности, при построении новой ломаной автоматически активизируются режимы захвата вершин и ребер уже существующих ломаных, позволяющие обеспечить слипание ломаных на контактирующих участках (отсутствие «дырок»). Для редактирования существующей ломаной реализованы операции удаления ее вершин, добавления вершин на ребра ломаной, перемещения вершин (рис. 1).

Закрепленная граница области задается набором ребер ломаных, принадлежащих ее внешней границе (на рис. 1 отображена жирной черной линией).

Внешняя статическая нагрузка задается с помощью набора векторов, для каждого из которых указываются принадлежащее внешней границе области ребро ломаной, серединная точка которого определяет центр площадки приложения соответствующей внешней силы, длина этой площадки, направление и величина силы (рис. 1, 3).

В рамках графического редактора реализована подсистема управления параметрами и режимами визуализации: масштаб визуализации, маркеры (кружки) вершин и их размер, заливка зон и маркеров вершин ломаных, направление обхода по ломаным, размеры рабочего поля, величина отступов на рабочем поле и т.д.

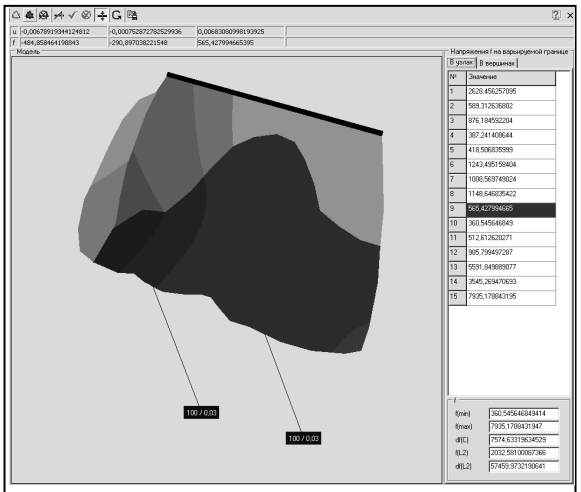


Рис. 3. Плоское сечение неоднородной конструкции

Модуль вычисления НДС плоской неоднородной области предназначен для численного решения плоской задачи упругости (1)−(4) на основе текущей геометрической модели плоского сечения конструкции и исходных данных – заданной внешней нагрузки, закрепленной границы, значений упругих параметров зон (коэффициентов Пуассона и модулей Юнга). В этом модуле реализован описанный выше алгоритм решения задачи (1)−(4) на основе метода граничных элементов. На рисунке 3 показан вывод результатов вычисления НДС плоской неоднородной области.

Модуль анализа НДС плоской неоднородной области предназначен для отображения результатов работы модуля вычисления НДС плоской неоднородной области (рис. 3) и организации доступа к процедурам решения ряда прикладных оптимизационных задач с использованием данных об НДС неоднородной области.

Модуль оптимизации формы границы служит для численного решения одного класса задач оптимизации формы области.

Модуль оптимизации значений упругих параметров предназначен для численного решения следующей оптимизационной задачи. Исследуется одна из зон, образующих неоднородную область. Указывается анализируемая часть ее контактной границы. Необходимо определить значения упругих параметров зоны, доставляющих минимум некоторому критерию. Этот критерий описывается с помощью функционала, характеризующего экстремальные свойства распределения напряжений и деформаций на интересующей части контактной границы выбранной зоны. Реализованы возможности построения ряда вариантов такого функционала. В частности, для проведения численного эксперимента в качестве оптимизируемого критерия было использовано максимальное значение перемещения в точках интересующей части контактной границы выбранной зоны. На рисунке 1 граница выбранной зоны отображена белым цветом, а интересующая часть ее контактной границы – пунктирной линией белого цвета.

Анализ НДС восстановленного при помощи цельнокерамического микропротеза депульпированного зуба

Интересным примером задачи анализа свойств неоднородной упругой конструкции может служить возникающая в стоматологии задача, которая заключается в определении оптимальной формы цельнокерамической реставрации – микропротеза, используемого для восстановления депульпированных зубов [3]. Как показывает клиническая практика, при восстановлении зуба с помощью такого микропротеза геометрические свойства его формы являются одним из важных факторов, определяющих успех протезирования. Оказывается,

за счет оптимального выбора формы микропротеза во многих случаях значительно увеличивается жизненный цикл реставрации. Поскольку возможности варьирования формы жевательной (внешней) поверхности микропротеза очень ограниченные, изменения формы этой части его поверхности могут быть весьма незначительными. В этих условиях увеличение жизненного цикла реставрации может достигаться путем формирования рациональной формы лишь внутренней поверхности микропротеза – контактной границы «микропротез–ткани зуба». Таким образом, возникает целесообразность (а в ряде клинических случаев и необходимость) в решении задачи оптимизации формы упругонапряженного тела – микропротеза в депульпированном зубе, подверженном воздействию заданной внешней нагрузки. При этом на допустимую форму микропротеза накладываются как габаритные ограничения – форма его внешней границы фиксирована, так и изопериметрические – объем материала микропротеза не должен превышать заданной величины, что при протезировании позволяет в максимальной степени сохранить здоровые ткани зуба. В такой задаче вид оптимизируемого показателя, необходимые и достаточные условия оптимальности контактной границы «микропротез–ткани зуба», а также численные методы ее построения [4] основаны на свойствах распределения возникающих на контактной границе напряжений. Перечисленные обстоятельства обусловливают необходимость достаточно точного вычисления значений компонент тензоров напряжений и деформаций на контактных границах, образующих конструкцию областей.

Геометрическая модель конструкции. Рассматривается неоднородная упругая область, подверженная воздействию заданной внешней статической нагрузки. Эта область представляет собой объединение конечного числа упругоконтактирующих друг с другом однородных подобластей (зон), обладающих различными механическими свойствами. Частным примером такой неоднородной упругой конструкции является реставрированный при помощи цельнокерамического микропротеза депульпированный зуб, состоящий из трех материалов – дентина, зубной эмали и керамики (рис. 4). Все материалы жестко связаны друг с другом – при упругой деформации конструкции смещения на контактных границах зон, соответствующих различным материалам, ничтожно малы.

В полном объеме задача определения НДС такой конструкции может быть решена в случае, если построена ее трехмерная геометрическая модель. На практике построение такой модели конструкции представляет собой достаточно сложную алгоритмическую и вычислительную задачу. В стоматологической практике исходные данные для построения модели могут быть получены при помощи специального технического оборудования (например, трехмерного цифрового сканера).

В данной работе рассматривается упрощенная постановка задачи упругости, основанная на общепризнанном экспертами в области сопротивления материалов методе плоских сечений трехмерной конструкции [5]. Для построения двухмерной геометрической модели плоского сечения неоднородной конструкции, например зуба, может использоваться рентгенограмма (рис. 4, 5).

Геометрическая модель плоского сечения конструкции (рис. 5) описывается плоской неоднородной областью, состоящей из конечного числа контактирующих зон. Каждая зона соответствует некоторому материалу (рис. 4) и задается аппроксимирующей ее границу замкнутой ломаной, определяемой последовательностью пар координат ее вершин в декартовой системе координат.

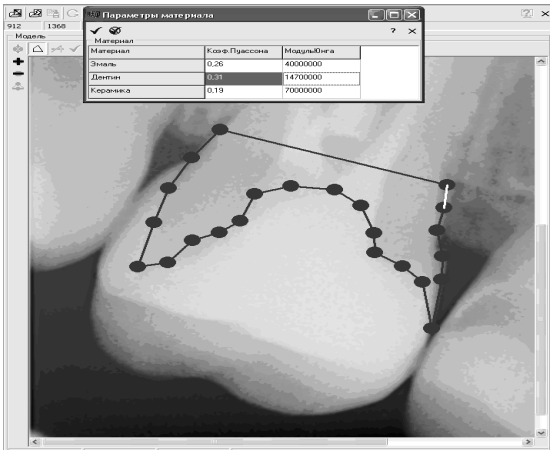


Рис. 4. Рентгенограмма восстановленного с помощью цельнокерамического протеза депульпированного зуба. Построение двухмерной геометрической модели плоского сечения неоднородной конструкции

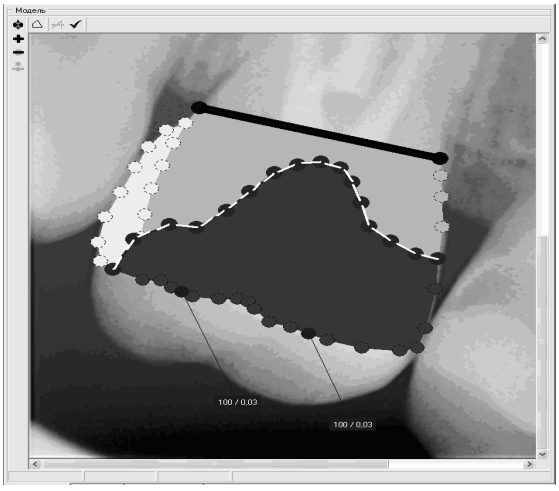


Рис. 5. Геометрическая модель плоского сечения депульпированного зуба, восстановленного при помощи цельнокерамического микропротеза и подверженного воздействию заданной статической нагрузки

Внешняя статическая нагрузка задается набором внешних сил, приложенных к участкам внешней границы области. Как уже отмечалось, каждая из сил характеризуется направлением (вектором), длиной площадки приложения и величиной силы (рис. 5). Центр площадки приложения силы является серединной точкой ребра ломаной, принадлежащего внешней границе области. Часть внешней границы области может быть жестко закреплена – в точках этой части границы перемещения полагаются равными нулю (на рис. 5 отображена жирной черной линией).

Анализ НДС плоского сечения восстановленного зуба. Результаты численного моделирования. Описанный выше алгоритм численного решения плоской задачи упругости используется для анализа НДС плоского сечения восстановленного с помощью цельнокерамического микропротеза зуба. Эта задача решается при следующих предположениях:

1) сечение зуба жестко закреплено на его нижней части границы (смещения равны нулю в точках этой границы) – игнорируется отдача корней зуба (рис. 5);

2) оставшаяся часть границы считается свободной (поверхностные напряжения в точках этой части границы равны нулю, кроме точек приложения внешних сил) – не учитывается взаимодействие исследуемого зуба с соседними зубами.

Указанные допущения адекватно отражают лабораторные условия проведения эксперимента.

Геометрическая модель плоского сечения зуба (рис. 5) описывается плоской неоднородной областью, состоящей в общем случае из трех типов (по типам материалов) контактирующих зон. На рисунке 5 зона дентина отображается светло-серым цветом, зона зубной эмали – белым цветом, зона керамики – темно-серым цветом, закрепленная граница указана жирной черной линией. Замкнутые ломаные, аппроксимирующие границы соответствующих зон, строятся по изображению рентгенограммы (рис. 4, 5). При этом на контактных участках вершины ломаных, соответствующих различным контактирующим зонам, совпадают (рис. 5).

Граничные элементы задаются ребрами ломаных, определяющих границы различных зон сечения зуба. Погрешность вычислений (погрешность метода) зависит от количества аппроксимирующих границы зон граничных элементов, а также от их длин. В численном эксперименте для аппроксимации границ зон сечения зуба использовалось более двухсот граничных элементов. Для вычисления компонент напряжений и перемещений в узлах (средних точках граничных элементов) приходилось решать систему линейных алгебраических уравнений достаточно большой размерности – от четырехсот неизвестных.

При численном моделировании рассматривалось несколько возможных вариантов направлений и точек приложения внешних сил. Эти варианты обусловливаются особенностями механизмов функционирования зубочелюстной системы. На рисунке 5 представлен один из возможных вариантов воздействия внешней нагрузки в виде двух векторов внешних сил, для каждого из которых указаны параметры соответствующей силы (величина силы и длина элементарной площадки ее приложения).

Целью решения задачи теории упругости являлась оценка распределения напряжений на границах зон. Особый интерес представляет контактная граница «микропротез–ткани зуба» (на рис. 5 изображена пунктирной линией белого цвета), так как распределение напряжений на этой границе может использоваться для оценки качества формы плоского сечения микропротеза и формирования прогноза длительности жизненного цикла реставрации в целом.

Свойства распределения напряжений на контактной границе «микропротез–ткани зуба» положены в основу численного метода решения задачи оптимизации формы этой границы. Как уже отмечалось, подобная задача заключается в определении такой ее геометрической формы, при которой реализовывалось бы оптимальное распределение напряжений на этой границе. При этом на допустимую форму сечения микропротеза накладывается изопериметрическое ограничение – площадь сечения должна быть равна заданной величине. Оптимальной формой контактной границы «микропротез–ткани зуба» считается такая, которая при заданной внешней нагрузке на область обеспечивает равномерное распределение напряжений на этой контактной границе.

Содержательно выполнение настоящего условия приводит к тому, что при заданной внешней нагрузке материал восстановленного зуба работает одинаково во всех точках вблизи контактной границы «микропротез–ткани зуба». Это обстоятельство позволяет, в свою очередь, избежать возникновения пиковых нагрузок на контактной границе, приводящих к уменьшению жизненного цикла реставрации. Поэтому в рассматриваемой задаче оптимизируемой характеристикой является величина (величины), отражающая степень равномерности распределения напряжений на варьируемой границе сечения зуба.

Представленный алгоритм применения метода граничных элементов, основанный на аналитическом вычислении интегралов, легко реализуется в программном виде. Предложенный способ вычисления НДС конструкции плоскими сечениями дает возможность достаточно точно оценить распределение напряжений в сечении конструкции. Это особенно актуально в задачах оптимизации форм упругих тел, в которых результаты оптимизации

часто соизмеримы с погрешностями численных методов расчета НДС тела.

Пакет прикладных программ прошел опытную апробацию на кафедре ортопедической стоматологии Уральской государственной медицинской академии (г. Екатеринбург). Реализованные в пакете функциональные возможности использовались для прогнозирования результатов реставрации депульпированных зубов реальных пациентов цельнокерамическими микропротезами. С помощью этого пакета проводился численный анализ НДС плоских сечений восстанавливаемых зубов с целью выработки рекомендаций по формированию рациональной формы контактных границ «микропротез–ткани зуба». В ряде случаев полученные результаты такого численного анализа (рис. 3) учитывались в клинической практике.

На основании проведенных вычислительных экспериментов следует сделать вывод о том, что разработанный пакет программ может использоваться для эффективного решения широкого круга прикладных задач анализа НДС конструкций, обладающих достаточно сложными геометрическими свойствами.

Литература

1. Бенерджи П., Баттерфилд Р. Методы граничных элементов в прикладных науках. М.: Мир, 1984. 494 с.

2. Федотов В.П., Спевак Л.Ф. Модифицированный метод граничных элементов в задачах механики, теплопроводности и диффузии. Екатеринбург: УрО РАН, 2009. 164 с.

3. Кандоба И.Н., Коледа П.А., Жолудев С.Е. Оптимизация формы накладки (pinlay). Моделирование упругонапряженного состояния зуба // Проблемы стоматологии. 2007. № 5. С. 19–23.

4. Кандоба И.Н. Метод решения одного класса задач оптимизации формы неоднородной области. Забабахинские научные чтения: сб. тез. докл. X Междунар. конф. (15–19 марта 2010 г.). Снежинск: Изд-во РФЯЦ-ВНИИТФ, 2010. С. 282–283.

5. Беляев Н.М. Сопротивление материалов. М.: Физматлит, 1962. 856 с.

УДК 62-533.65/69

ИНТЕЛЛЕКТУАЛЬНАЯ СИСТЕМА УПРАВЛЕНИЯ И МОНИТОРИНГА ГАЗОВОЙ КОТЕЛЬНОЙ

О.А. Белоусов, к.т.н.; С.В. Иванов

(Тамбовский государственный технический университет, jiour@mail333.com, udarnikk@mail.ru)

Рассматривается задача интеллектуального управления и мониторинга тепловыми аппаратами, разработан алгоритм управления и мониторинга газовой котельной в реальном масштабе времени на базе нейронной сети и нечеткой логики.

Ключевые слова: *нечеткая логика, нейронные сети, автоматическое регулирование, информационные технологии, управление котлом, газовая котельная, тепловой процесс, система управления.*

Газовые котельные нашли широкое применение в промышленных производствах в силу их превосходных эксплуатационных характеристик. Установка модульных котельных имеет множество преимуществ, так как они не привязаны к устаревшим коммуникациям и обеспечивают производство недорогой тепловой энергии. Сегодня газовые котельные считаются самыми эффективными по КПД и теплоснабжению, способны давать необходимое количество тепла при небольших финансовых и трудовых затратах, просты в монтаже и эксплуатации. Автоматизация котельной позволяет сократить затраты на обслуживание: оборудование, установленное в блочно-модульной котельной, работает автономно, от персонала требуются только наблюдение и контроль.

Любая котельная установка включает один или несколько котлов, нагревающих теплоноситель до нужной температуры. Для создания комфортных условий в отапливаемом помещении на выходе котлов необходимо иметь горячую воду постоянной температуры. На температуру нагреваемой воды влияет множество факторов, таких как суточные и сезонные колебания температуры окружающей среды, переменное давление подводимого газа, неравномерный расход воды и др. Получение постоянной температуры нагреваемой воды в данной ситуации невозможно без регулировки режима работы отопительного котла.

Задача состоит в том, чтобы автоматизировать регулирование температуры воды, для этого в большинстве случаев необходимо управлять мощностью котлов. Плавное управление мощностью котла возможно с использованием сервопривода на подачу газа (рис. 1).

Процесс нагрева или остывания воды несколько инерционен: если до включения большого горения котла температура воды в системе отопления падала, то после включения она может продолжать падать пусть небольшой, но конечный промежуток времени. Аналогичная картина наблюдается и при выключении котла [1].

Условимся, что увеличение подачи газа в котел, а следовательно, и увеличение пламени в горелке

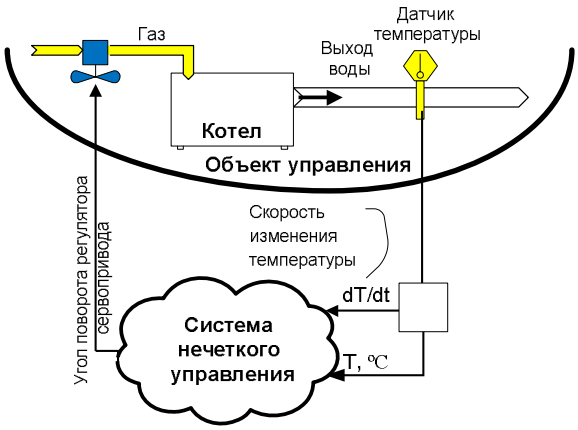


Рис. 1. Структурная схема системы нечеткого управления газовым котлом

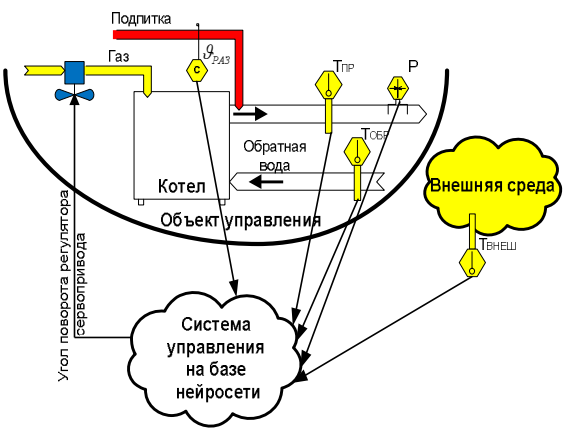


Рис. 2. Структурная схема системы нейросетевого управления газовым котлом

(прибавление мощности) осуществляются поворотом регулятора сервопривода вправо (то есть на положительный угол), а уменьшение подачи газа – поворотом регулятора сервопривода влево (на отрицательный угол).

Как альтернативу традиционным алгоритмам управления газовым котлом можно предложить алгоритм управления на базе нечеткой логики с целью качественного регулирования температуры в системах отопления и горячего водоснабжения. Чтобы учесть особенности процесса нагрева воды и большинство дестабилизирующих факторов, а также исключить затраты, связанные с частым включением и выключением котла, в качестве выходного параметра необходимо рассматривать не только температуру воды в системе, но и скорость ее изменения. Эта информация будет использоваться при построении базы правил системы нечеткого вывода, которая позволяет реализовать данную модель нечеткого управления.

Одним из достоинств нечеткого регулятора является его работоспособность в условиях частичной неопределенности (возможность безотказной работы при выходе из строя практически всех датчиков температуры и давления, за исключением датчика температуры прямой воды), так как требуется всего один датчик температуры. Нечеткий регулятор мощности отопительного котла позволяет достаточно быстро вывести систему отопления на заданный режим при запуске и так же быстро остановить систему при критических температурах, полностью перекрыв подачу газа. К недостаткам нечеткого регулятора можно отнести большое время реакции на падение температуры и, как следствие, отклонение от заданной температуры. Такой регулятор удобно использовать при пуске котельной и в аварийных или критических ситуациях.

В качестве основного регулятора мощности газового котла можно предложить регулятор на основе *нейронных сетей* (НС) (рис. 2).

Как уже упоминалось, на процесс нагрева теплоносителя (воды) влияет множество факторов, которые следует учитывать при разработке нейросетевого регулятора. Представление этих факторов в виде входов НС позволяет достигнуть высокого качества регулирования температуры. Эффективно обученная НС будет обладать минимальным временем реакции на падение температуры в системе и высокой точностью регулирования. Недостатком данного регулятора является полная утрата работоспособности при выходе из строя хотя бы одного входного датчика.

Оценивая достоинства и недостатки предложенных регуляторов, можно построить систему управления котельной, включающую оба эти регулятора. В качестве основного будет использоваться нейросетевой регулятор, а при критических температурах или в случае выхода из строя оборудования и в аварийных ситуациях управление передается нечеткому регулятору. В результате работа не остановится даже при серьезных поломках котельного оборудования.

Построение описанной системы управления следует начать с проектирования нечеткого регулятора мощности газового котла. Для создаваемой системы нечеткого вывода в качестве входных переменных рассмотрим две нечеткие лингвистические переменные: $\overline{T} = \{\overline{t}_{\mathrm{NB}}, \overline{t}_{\mathrm{NS}}, \overline{t}_{\mathrm{Z}}, \overline{t}_{\mathrm{PS}}, \overline{t}_{\mathrm{PB}}\}$ – температура нагреваемой воды с термами «очень низкая температура», «низкая температура», «нормальная температура», «высокая температура», «очень высокая температура» и $\overline{V} = \{\overline{v}_{NS}, \overline{v}_{Z}, \overline{v}_{PS}\}$ – скорость изменения температуры нагреваемой воды с термами «отрицательная скорость», «равна нулю», «положительная скорость». Функции принадлежности нечетких множеств $\overline{t}_{\mathrm{NB}}, \overline{t}_{\mathrm{NS}}, \overline{t}_{\mathrm{Z}}, \overline{t}_{\mathrm{PS}}, \overline{t}_{\mathrm{PB}}$ приведены на рисунке 3.

Скорость изменения температуры воды характеризуется разностью температур нагреваемой воды за единицу времени. Следовательно, скорость

будет иметь положительный знак при нагреве воды и отрицательный при ее остывании. При постоянной температуре воды скорость равна нулю. Функции принадлежности нечетких множеств $\overline{v}_{NS}, \overline{v}_{Z}, \overline{v}_{PS}$ приведены на рисунке 4.

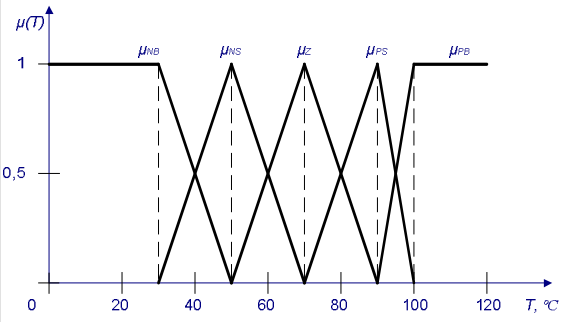


Рис. 3. Функция принадлежности для термов входной лингвистической переменной «температура нагреваемой воды»

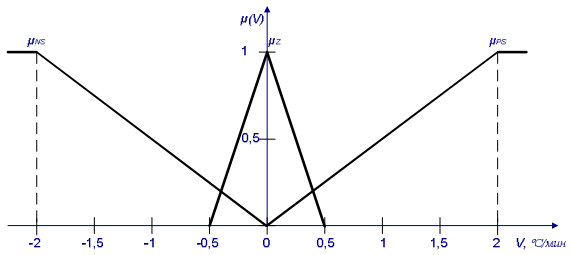


Рис. 4. Функция принадлежности для термов входной лингвистической переменной «скорость изменения температуры нагреваемой воды»

Выходная лингвистическая переменная, которой является угол поворота регулятора сервопривода $\overline{A} = \{\overline{\alpha}_{NB}, \overline{\alpha}_{NM}, \overline{\alpha}_{NS}, \overline{\alpha}_{Z}, \overline{\alpha}_{PS}, \overline{\alpha}_{PM}, \overline{\alpha}_{PB}\}$, имеет следующие термы: «очень большой угол влево», «большой угол влево», «небольшой угол влево», «не поворачивать», «небольшой угол вправо», «большой угол вправо», «очень большой угол вправо».

Функции принадлежности нечетких множеств $\overline{\alpha}_{NB}, \overline{\alpha}_{NM}, \overline{\alpha}_{NS}, \overline{\alpha}_{Z}, \overline{\alpha}_{PS}, \overline{\alpha}_{PM}, \overline{\alpha}_{PB}$ приведены на рисунке 5 [2].

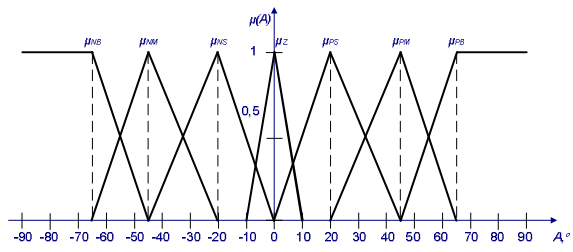


Рис. 5. Функция принадлежности для термов выходной лингвистической переменной «угол поворота регулятора сервопривода»

На основании входных и выходных лингвистических переменных формируется база продукционных правил системы нечеткого вывода:

П1: «Если $\overline{T}$ есть $\overline{t}_{PB}$ и $\overline{V}$ есть $\overline{v}_{PS}$, то $\overline{A}$ есть $\overline{\alpha}_{NB}$»

П2: «Если $\overline{T}$ есть $\overline{t}_{PB}$ и $\overline{V}$ есть $\overline{v}_{NS}$, то $\overline{A}$ есть $\overline{\alpha}_{NS}$»

…

П14: «Если $\overline{T}$ есть $\overline{t}_{Z}$ и $\overline{V}$ есть $\overline{v}_{NS}$, то $\overline{A}$ есть $\overline{\alpha}_{PS}$»

П15: «Если $\overline{T}$ есть $\overline{t}_{Z}$ и $\overline{V}$ есть $\overline{v}_{Z}$, то $\overline{A}$ есть $\overline{\alpha}_{Z}$»

Таким образом, в результате выполнения данного этапа были определены функции принадлежности для всех входных и выходных переменных, а также созданы правила нечеткой продукции.

Следующим этапом проектирования системы управления газовой котельной является разработка нейросетевого регулятора. Построение НС происходит в три этапа: выбор типа сети (архитектуры), подбор весовых коэффициентов (обучение) и проверка сети.

На первом этапе разработчик НС сталкивается со следующей проблемой: с одной стороны, существует большое количество типов НС (сети Хопфилда, Хэмминга, Кохонена, перцептроны и т.д.), а с другой – отсутствует строгая теория по выбору конкретного типа, поэтому основой этого этапа является оценка необходимого числа синаптических весов и нейронов в скрытых слоях, которые соответственно рассчитываются по следующим формулам [3]:

$$\frac{mN}{1+\log_2 N} \le L_w \le m\left(\frac{N}{m}+1\right)(n+m+1)+m, \quad (1)$$

где n – размерность входного сигнала; m – размерность выходного сигнала; N – число элементов обучающей выборки;

$$L = \frac{L_w}{n+m}, \quad (2)$$

где L – число нейронов в скрытом слое.

Следующим этапом разработки НС является ее обучение. Процесс обучения может рассматриваться как настройка архитектуры сети и весов связей для эффективного выполнения специальной задачи. Обычно НС должна настроить веса связей по имеющейся обучающей выборке. Функционирование сети улучшается по мере итеративной настройки весовых коэффициентов. Процесс обучения НС показан на рисунке 6.

На завершающем этапе созданную НС проверяют на примерах, не вошедших в обучающую выборку, после чего делается заключение о пригодности данной сети к применению.

Рассмотрим процесс создания НС для регулирования мощности газового котла.

Мощность отопительного котла будем регулировать изменением угла поворота (α) сервопривода, ограничивающего подачу газа в котел. Критерием оптимальности положим стремление темпе-

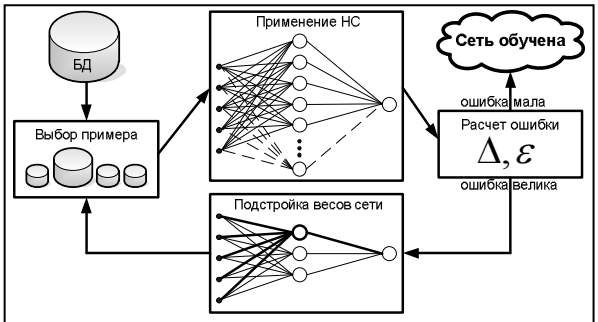


Рис. 6. Иллюстрация процесса обучения НС

ратуры прямой воды, $T_{ПР}$, к заданной температуре воды, $T_{ЗАД}$ ($T^*_{ПР}=T_{ЗАД}$), которая является некоторой функцией от температуры окружающей среды, $T_{ВНЕШ}$ ($T_{ЗАД}=f(T_{ВНЕШ})$).

В качестве параметров, влияющих на процесс нагрева воды, следует отметить также давление (Р) в системе и скорость разбора воды $\vartheta_{РАЗ}$ из системы.

Таким образом, единственным выходом НС будет величина угла поворота (α) сервопривода. На входы НС подаются следующие величины:

- температура прямой воды ($T_{ПР}$);
- температура обратной воды ($T_{ОБР}$);
- заданная температура воды ($T_{ЗАД}$);
- скорость разбора воды из системы $Y_{РАЗ}$;
- давление воды в системе (Р).

Для формул (1) и (2) имеем следующие параметры: n=5, m=1, N=1024, по которым проведем примерную оценку необходимого числа синаптических весов и нейронов в скрытых слоях:

$$\frac{5\cdot 1024}{1+\log_2 1024} \le L_w \le 5\cdot\left(\frac{1024}{5}+1\right)(5+1+1)+5;$$

$$465,45 \le L_w \le 7208;\ L=\frac{1200}{6}=200.$$

Архитектура сети (см. рис. 7) представляет собой трехслойный персептрон. НС обучалась по алгоритму обратного распространения ошибки с использованием программного пакета Matlab.

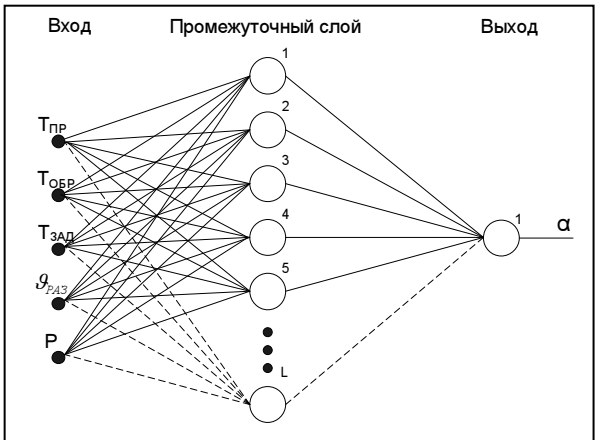


Рис. 7. Архитектура сети

Важным направлением экономии затрат на систему управления является разработка одного управляющего устройства для управления группой газовых котлов.

Общая задача интеллектуального управления группой из n котлов формулируется следующим образом. Задаются:

– оптимальная температура воды ($T_{ЗАД}$) (вычисляется как функция от температуры окружающей среды или задается оператором);

– общие входные параметры, одинаковые для всех котлов в котельной установке (температура обратной воды ($T_{ОБР}$), температура окружающей среды ($T_{ВНЕШ}$), скорость разбора воды из системы ($Y_{РАЗ}$), давление воды в системе (Р)), которые непрерывно поступают на вход контроллера от различных датчиков, расположенных внутри котельной;

– температура воды в котле (температура прямой воды после каждого котла) {$T_{ПР1}$, $T_{ПР2}$, ..., $T_{ПРn}$}.

Требуется определить множество управляющих воздействий {α1, α2, ..., αn} в виде углов поворота сервоприводов на подачу газа для каждого котла, таких, чтобы температура прямой воды после каждого котла стремилась к заданной температуре. Задачи интеллектуального управления котлами решаются промышленным многоканальным контроллером или удаленной рабочей станцией без участия человека.

Управление данными газовых котлов и котельного оборудования осуществляется системой интеллектуального управления и мониторинга (см. рис. 8). В состав данной системы входят датчики входных параметров котельной, виртуальный датчик, база данных, экспертная система с БЗ и блок выбора регулятора управления.

Виртуальный датчик (ВД) является обобщением понятия физического датчика и вычислительного модуля. В любом ВД наряду с сенсорными выполняются вычислительные операции. Непосредственно сенсорные операции применяемый ВД не выполняет, а использует информацию, посылаемую другими датчиками. Получая информацию от физических датчиков, ВД определяет цель управления (нагрев воды до заданной температуры – цель 1, поддержание заданной температуры – цель 2, регулирование в аварийных условиях – цель 3, регулирование при критических температурах – цель 4) [4]. Поддержание заданной температуры является основным режимом функционирования котельной и осуществляется с помощью нейросетевого регулятора. В остальных режимах используется нечеткий регулятор, способный стабильно работать в аварийных ситуациях и при критических температурах. Таким образом, котлы, входящие в одну группу, могут управляться с использованием различных алгоритмов. В БЗ экспертной системы содержатся сведения, получен-

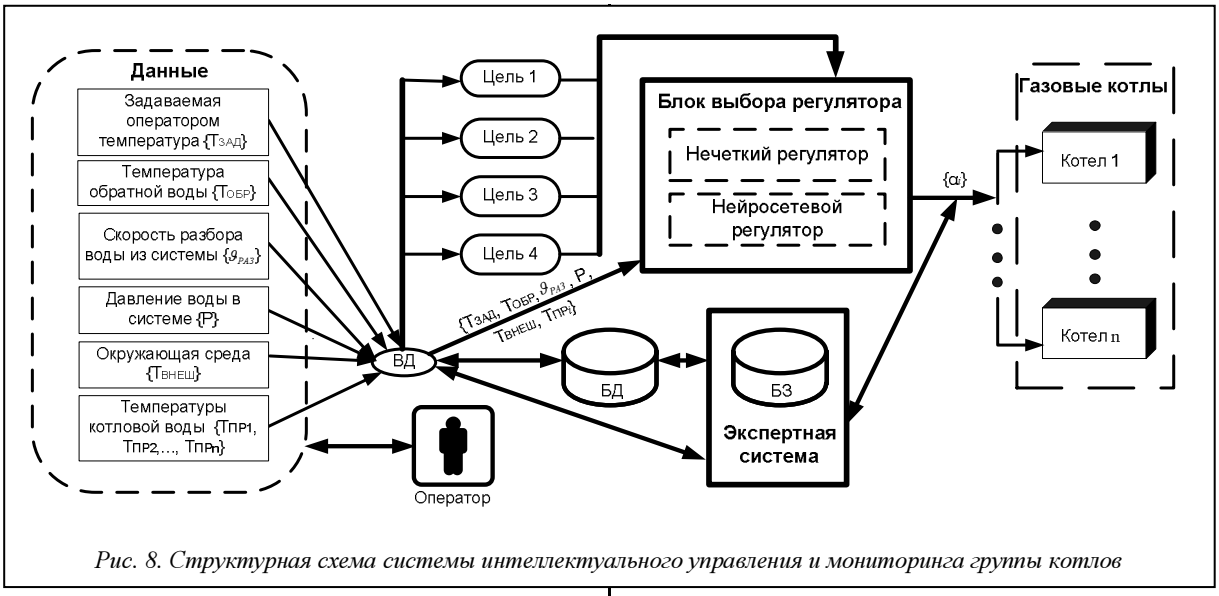


Рис. 8. Структурная схема системы интеллектуального управления и мониторинга группы котлов

ные от экспертов, значения функций принадлежностей входных и выходных лингвистических переменных, данные для обучения НС и др. В БД сохраняются значения всех входных параметров и управляющих воздействий.

Вариант технической реализации системы приведен на рисунке 9. В системе предусмотрены регистрация всех параметров и температур, синтез в реальном времени управляющих воздействий, а также удаленная диспетчеризация. Структура системы интеллектуального управления включает промышленный контроллер, панель оператора, промышленный GSM-модем и рабочую станцию с БД. В БД содержится информация о результатах ранее решенных задач управления [5].

Для управления котлами используются алгоритм нечеткой логики и нейросетевой алгоритм. Предусмотрен режим адаптации с автоматической коррекцией параметров модели и функций принадлежности нечетких множеств, которые по окончании режима адаптации записываются в память контроллера для последующего использования.

Для централизованного управления тепловыми режимами в котлах предусмотрена возможность выхода в Интернет. Это позволяет передавать данные о работе группы котлов или группы котельных установок в единую БД с возможностью не только постоянного визуального контроля и накопления данных, но и изменения или корректировки процесса отопления в режиме удаленного доступа. Также возможно управление работой котельной установки с мобильного телефона, имеющего доступ в Интернет, а сообщения об авариях или ошибках приходят в виде SMS-сообщений на мобильный телефон оператора.

Область применения полученных результатов по интеллектуальному управлению достаточно широка и не ограничивается только газовыми котлами. Предложенные алгоритмы значительно уп-

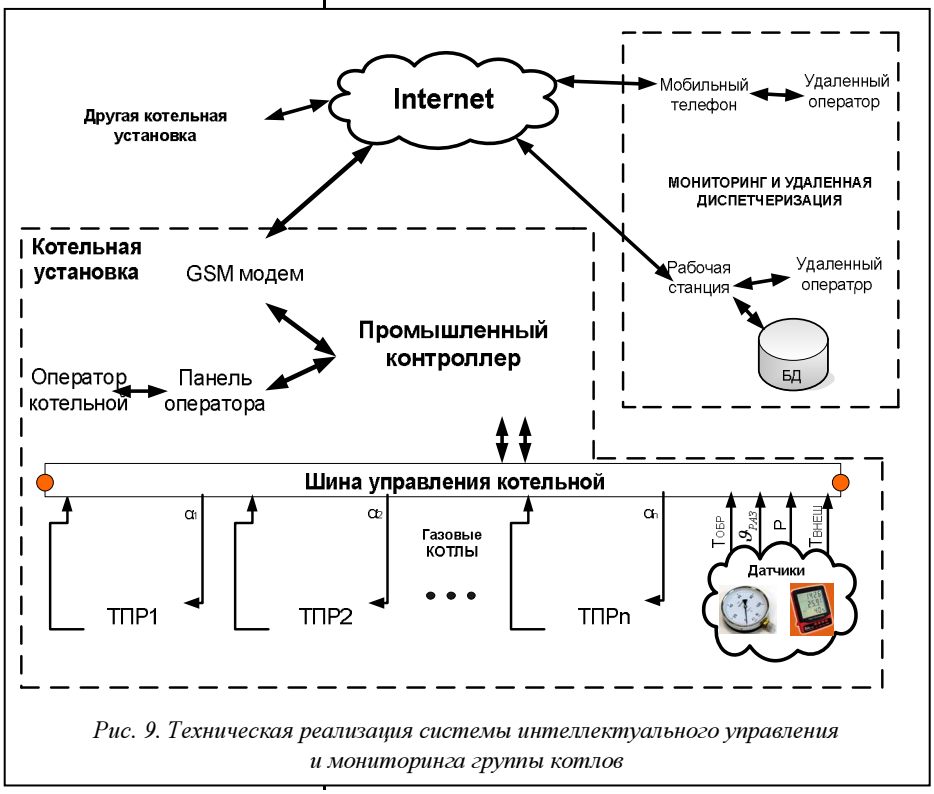


Рис. 9. Техническая реализация системы интеллектуального управления и мониторинга группы котлов

рощают обработку данных, в результате снижаются требования к быстродействию технических средств, что положительно сказывается на их стоимости.

Большие возможности появляются при использовании режимов удаленного доступа и управления пространственно распределенными объектами. Удаленный доступ значительно повышает возможности масштабного контроля процессов. Это особенно актуально для компаний, которые имеют множество территориально разнесенных представительств и могут из центрального офиса осуществлять мониторинг всех технологических процессов филиалов и управление ими. Сочетание интеллектуального управления и удаленного доступа существенно упрощает внедрение и отладку алгоритмов управления и, как следствие, повышает экономическую эффективность эксплуатации оборудования.

Опыт создания систем энергосберегающего управления тепловыми аппаратами на кафедре «Конструирование радиоэлектронных и микропроцессорных систем» Тамбовского государственного технического университета показал, что эффект от их внедрения достаточно высок и может достигать 25 % и более.

Литература

1. Леоненков А.В. Нечеткое моделирование в среде MATLAB и fuzzy Tech. СПб: БХВ-Петербург, 2005. 736 с.

2. Кофман А. Введение в теорию нечетких множеств. М.: Радио и связь, 1982. 432 с.

3. Хайкин С. Нейронные сети: полный курс; 2-е изд.; [пер. с англ.]. М.: Издат. дом «Вильямс», 2006. 1104 с.

4. Белоусов О.А. Виртуальный датчик в системе управления электрокамерными печами // Приборы и системы. Управление, контроль, диагностика. 2005. № 4. С. 31–33.

5. Белоусов О.А. Автоматизированная система энергосберегающего управления электрокамерными печами // Автоматизация в промышленности. 2005. № 5. С. 8–9.

УДК 621.311:658.26

АВТОМАТИЗИРОВАННАЯ СИСТЕМА УПРАВЛЕНИЯ ТЕХНОЛОГИЧЕСКИМИ ПРОЦЕССАМИ В МНОГОНОМЕНКЛАТУРНЫХ ПРОИЗВОДСТВАХ

Г.Б. Бурдо, д.т.н.; Н.А. Семенов, д.т.н.; А.А. Исаев
(Тверской государственный технический университет, gbtms@yandex.ru).

В статье изложены результаты исследований функций, структуры и моделей интеллектуальной автоматизированной системы управления технологическими процессами в единичном и мелкосерийном производствах.

Ключевые слова: *управление технологическими процессами, искусственный интеллект, системный анализ.*

Внедрение *АСУ технологическими процессами* (АСУ ТП) является одним из эффективных средств, позволяющих в единичном и мелкосерийном производствах обеспечить работу по принципу «точно вовремя» [1]. Актуальность постановки такой задачи объясняется недостаточно развитой нормативной базой, низкой структуризацией процессов управления, отсутствием учета динамики ситуации, складывающейся в технологических подразделениях данных типов производств.

Авторами выявлены функции АСУ ТП в условиях единичного и мелкосерийного производств и исходные данные для ее работы [2]. Структура АСУ ТП разработана в соответствии с принципами единства и взаимосвязи, одновременности, непрерывности, гибкости, точности, оперативности, эвристичности, относительной оптимальности, участия [2].

Для исследования работы АСУ ТП был применен системный подход и рассмотрена теоретико-множественная модель АСУ ТП как части интегрированной системы САПР ТП–АСУ ТП.

Технологические подразделения (ТхП) являются управляемыми АСУ ТП подсистемами, состоящими из различных подсистем – участков.

Управляющая система S представляет собой совокупность операторов, реализующих систему управления технологическими подразделениями. Сами технологические подразделения осуществляют преобразование предметов производства в соответствии с алгоритмами, задаваемыми командами $\{W_{ij}\}$ операторам $\{R_j\}$. Операторы $\{R_j\}$ предназначены для контроля за технологической дисциплиной, управления работой непосредственно на рабочих местах и получения информации о ходе работ ($\{R_j\}=\{R_1, R_2, R_j, R_m,\}$). Они реализуются линейными мастерами, отвечающими за работу подчиненных им элементов технологических подразделений ($\{\{K_i\}_j\}$), причем обязательно условие

$$\bigcup_{i=1}^{M}\{K_i\}_j = \{K_{11}, K_{n1}, K, ..., K_{n2}, ..., K_{ij}, ..., K_{mm}\},$$

$\bigcap_{j=1}^{m}\{K_i\}_j = \varnothing$, то есть управляемыми должны быть все подразделения, но одновременного подчинения нескольким операторам из множества $\{R_j\}$ не допускается. Каждым оператором R_j управляются подразделения K_{1j}, K_{2j}, …, K_{ij}, …, K_{nj}, образующие множества $\{K_i\}_j$. Оператором R_j выполняется несколько функций.

Первая из них, R_{1ij}, выполняет функцию контроля за соблюдением технологической дисциплины в каждом из подчиненных оператору R_j i-х: R_{1ij}: $M_{1j}\times V_{ij}\to W_{ij}$, где M_{1j} – множество параметров, описывающих операционные ТП для $\{K_i\}_j$ участков технологических подразделений и содержащихся в технологической документации (передаются от САПР ТП); V_{ij} – множество параметров, описывающих фактическое выполнение технологий на i-м участке; W_{ij} – множество управляющих воздействий, вводящих ТП на участке в рамки, оговоренные технологической документацией.

Тогда совокупная функция $\{R_{1i}\}_j$ по всем подчиненным оператору R_j i-м участкам R_{1j} будет $\{R_{1i}\}_j : M_{1j}\times\{V_i\}_j \to \{W_i\}_j$; $\{R_{1i}\}_j = \bigcup_i R_{1ij} = R_{1j}$, где $\{V_i\}_j$ – множество параметров, описывающих фактическое выполнение технологий на всех ij-х участках (j=1, 2, …, i, …, n); $\{W_i\}_j$ – множество управляющих воздействий, вводящих технологии на ij-х участках, управляемых R_j, в состояние, оговоренное технологической документацией.

Вторая частная функция операторов, $\{R_j\}$, управляет распределением работ по рабочим местам участков технологических подразделений. По каждому i-му участку вторая функция оператора $\{R_j\}$ следующая: $R_{2ij} : Z_{1ij}\times M_{1j}\times Y_{ij}\times L_{3j}\to X_{ij}$, где L_{3j} – управляющее воздействие от управляющей подсистемы с целью приведения *календарного плана-графика* (КПГ) к расчетному данному участку; Z_{1ij} – множество параметров, описывающих КПГ выпуска деталей, узлов и изделий исходя из планового числа рабочих мест на i-м участке, подчиненном j-му оператору R_j; $\{Z_{1i}\}_j = \bigcup_i Z_{1ij} = Z_{1j}$, Z_{1j} – множество параметров КПГ выпуска деталей, узлов и изделий на всех i-х участках, управляемых оператором R_{2j}; M_{1j} – множество параметров, характеризующих маршрутную технологию изготовления деталей, узлов и изделий на i-м участке, получаемых от САПР ТП; Y_{ij} – множество параметров, характеризующих загрузку и число фактических рабочих мест в пределах групп оборудования на i-м участке; X_{ij} – множество параметров, характеризующих управляющие воздействия, приводящие фактическое выполнение КПГ в соответствие расчетному на i-м участке с учетом загрузки рабочих мест в пределах групп оборудования.

Функция оператора R_{2j} по всем подчиненным ему подразделениям, $\{R_{2i}\}_j$, такова: $\{R_{2i}\}_j=\{Z_{1i}\}_j\times\{M_{1i}\}_j\times\{Y_i\}_j\times\{L_{3i}\}_j\to\{X_{1i}\}_j$, где $\{L_{3i}\}_j = \bigcup_i L_{3ij} = L_{3j}$ – множество управляющих воздействий от управляющей подсистемы с целью приведения в соответствие КПГ работы всех участков, подчиненных R_j; $\{M_{1j}\}_i = \bigcup_i M_{1ij} = M_{1j}$ – множество параметров, характеризующих ТП изготовления изделий на всех i-х участках, подчиненных оператору R_j; $\{Y_i\}_j = \bigcup_i Y_{ij} = Y_j$ – множество параметров, характеризующих загрузку и число фактических рабочих мест в пределах групп оборудования на i-х участках, управляемых R_j; $\{X_i\}_j = \bigcup_i X_{ij} = X_j$ – множество параметров, характеризующих управляющие воздействия оператора R_j по всем участкам технологических подразделений, приводящие фактическое выполнение КПГ в соответствие расчетному с учетом загрузки рабочих мест по всем группам станков на всех i-х участках; $\{R_{2i}\}_i = \bigcup_i R_{2ij} = R_{2j}$, $R_{2j}=M_{1j}\times Y_j\times Z_{1j}\times L_{3j}\to X_{1j}$.

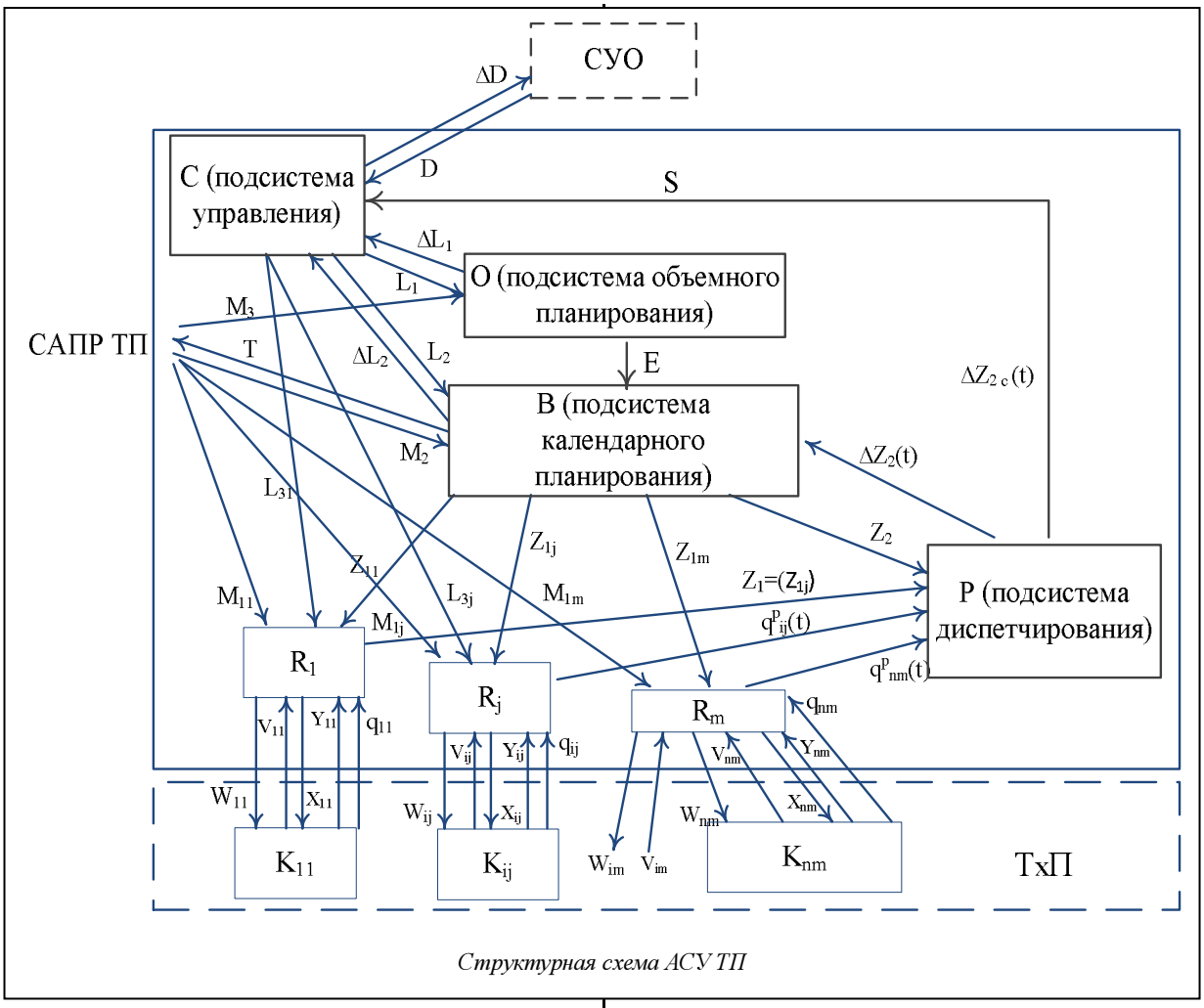
Третья функция операторов, $\{R_j\}$, связана с предоставлением необходимой информации о ходе выполнения КПГ в систему диспетчирования. Следует оговориться, что эта функция может выполняться как линейными мастерами, так и диспетчерами подсистемы P (в зависимости от организации системы диспетчирования).

Функция R_{3ij} в рамках каждого участка технологических подразделений следующая: $R_{3ij} : Z_{1ij}\times q_{ij}(t)\to q_{ij}(t)$, где $q_{ij}(t)$ – множество параметров, описывающих фактическое число рабочих мест и выполнения КПГ на i-м участке по выпуску изделий, управляемом оператором R_j; $q_{ij}^p(t)$ – то же самое, но в принятой в подсистеме (операторе) P форме, индекс t в скобках обеих величин означает, что данные вводятся в строго оговоренные временные интервалы.

Комплекс третьих функций оператора R_j по всем подчиненным ему участкам (элементам) технологических подразделений будет выглядеть как $\{R_{3i}\}_j = \bigcup_i R_{3ij} = R_{3j} : Z_{1i}\times\{q_i\}_j(t)\to\{q_i\}_j^p(t)$, где $\{q_i\}_j(t) = \bigcup_i q_{ij}(t) = q_j(t)$ – множество параметров, описывающих фактическое число рабочих мест и выполнение КПГ на i-х участках, управляемых оператором R_j; $\{q_i\}_j^p(t) = \bigcup_i q_{ij}^p(t) = q_j^p(t)$ – то же самое, но в принятой в подсистеме диспетчирования форме.

Рассмотрим функции подсистем АСУ ТП, показанные на рисунке.

Функции подсистемы диспетчирования P следующие. Первая из них, P_1, связана с оценкой вы-



Структурная схема АСУ ТП

полнения КПГ по всем участкам технологических подразделений в целом: P_1: $Z_2 \times \{\{q_i\}^p\}_j(t) \to \Delta Z_2(t)$, где Z_2 – множество параметров, характеризующих плановый КПГ и плановое число рабочих мест по технологическим подразделениям в целом в форме, необходимой для подсистем P, $\{\{q_i\}^p\}_j(t) = Q^p(t)$; ΔZ_2 – множество параметров, характеризующих отклонения КПГ и числа рабочих мест от планового по всем участкам технологических подразделений.

Вторая функция предназначена для передачи аналогичной информации в подсистему (с начальным производством) в удобном для пользователя виде: $P_2 : Z_2 \times Q^p(t) \to \Delta Z_2^c(t)$, где $\Delta Z_2^c(t)$ – множество параметров, характеризующих отклонения от КПГ и числа рабочих мест от планового по всем участкам технологических подразделений.

Функции подсистемы B следующие. Первые две из них связаны с разработкой множеств параметров, характеризующих КПГ для операторов $\{R_j\}$ и P: B_1:$E \times M_2 \times L_2 \to Z_1$; B_2:$E \times M_2 \times L_2 \to Z_2$, где E – множество параметров, характеризующих номенклатуру изготовляемых изделий с указанием календарных сроков их изготовления (объемные планы); M_2 – множество параметров, характеризующих маршруты изготовления изделий с указанием времени их выполнения (штучно-калькуляционное время, заключительное время и пр., сведения по используемому в операции инструменту), передаваемые от САПР ТП; L_2 – множество параметров, определяющих уточнения и корректировки для оператора B, указания о необходимости или отказе от пересчета календарных планов, их утверждение, предельные календарные сроки в КПГ по изделиям, плановая численность рабочих мест.

Функция B_3 определяет отличие расчетного КПГ от задаваемого множеством параметров L_2: B_3:$E \times M_2 \times L_2 \times \Delta Z_2(t) \to \Delta L_2$, где ΔZ_2 – множество параметров, характеризующих отклонения по срокам выполнения расчетного КПГ от задаваемых в множестве L_2.

Функция B_4 определяет фактическую загрузку оборудования по типам и группам оборудования участков: B_4:$E \times M_2 \times L_2 \times \Delta Z_2(t) \to T$, где T – множество параметров, определяющих фактическую загрузку оборудования по группам на момент t (то есть с помощью учета фактической численности рабочих, прохождения деталей по станкам и т.д.).

Функции оператора O (подсистема объемного планирования) следующие. Функция O_1 определяет исходные параметры для расчета КПГ: O_1: $L_1 \times M_3 . \rightarrow E$, где L_1 – множество параметров, определяющих перечень изделий, изготавливаемых за определенный календарный период (задаваемый объемный план), плановую численность рабочих мест; M_3 – множество параметров, определяющих трудоемкость изделий по видам работ.

Функция O_2 определяет расчетные отклонения от задаваемого объемного плана: O_2: $L_1 \times (L_1 \times M_2) \rightarrow \Delta L_1$, где ΔL_1 – множество параметров, характеризующих отличие расчетного объемного плана от задаваемого.

Подсистема управления (C) реализует следующие функции. Первая из них определяет объем задания для подсистемы O (объемный план) и может быть реализована в следующих вариантах: C_1^1: $D \rightarrow L_1$, где D – множество параметров, определяющих перечень и календарные сроки выпуска изделий. Данная подфункция реализуется при первоначальном формировании плана объемного выпуска изделий (по номенклатуре).

Вторая подфункция, C_1^2: $D \times \Delta L_1 \rightarrow L_1$, используется при итерационных процедурах уточнения рассчитанного плана объемного выпуска изделий.

Третья подфункция, C_1^3: $D \times \Delta L_2 \rightarrow L_1$, используется при уточнении плана объемного планирования путем итерации на основе анализа сформированного КПГ.

Четвертая подфункция реализуется при уточнении объемного плана на основе оценки выполнения КПГ на рабочих местах технологических подразделений: $C_1^4 : D \times \Delta Z_2^c(t) \rightarrow L_1$.

Вторая функция определяет управляющие воздействия на подсистему B и имеет несколько подфункций.

Первая из них, C_2^1: $D \rightarrow L_2$, определяет конечные календарные сроки L_1 выпуска определенных изделий и другие ограничения по разрабатываемому КПГ.

Вторая подсистема реализуется при уточнении сформированного КПГ на основе анализа его отличий ΔL_2 от указания L_2 путем итерационных процедур: C_2^2: $D \times \Delta L_2 \rightarrow L_2$.

Третья подфункция определяет управляющее воздействие L_2 на основе анализа реализации КПГ в технологических подразделениях: $C_2^3 : D \times \times \Delta Z_2^c(t) \rightarrow L_2$.

Третья функция осуществляет обратную связь с системой управления организацией более высокого уровня.

Первая подфункция имеет вид C_3^1: $\Delta L_1 \rightarrow \Delta D$, где ΔD – множество параметров, описывающих отклонения от планового задания D по срокам и номенклатуре изделий на основе анализа сформулированного объемного плана.

Вторая подфункция определяет возможные отклонения ΔD на основе анализа сформированного КПГ: $C_3^2 : D \times \Delta L_2 \rightarrow \Delta D$.

Третьей подсистемой определяются возможные отклонения ΔD на основе анализа результатов работы технологических подразделений: C_3^3: $D \times \times \Delta Z^c(t) \rightarrow \Delta D$.

Четвертая подфункция служит для прямого управления работой операторов $\{R_j\}$ с целью приведения КПГ в соответствие расчетному и охватывает организационную сторону управления: $C_4^1 : D \times Z_2^c(t) \times \Delta L_2 \times D \rightarrow L_3$; $L_3 = \{L_{3j}\}$.

Таким образом, в АСУ ТП реализованы три вида обратных связей: по объемному плану, КПГ и результатам диспетчирования ТП. Кроме того, в составе интегрированной САПР ТП–АСУ ТП последняя подсистема имеет обратную связь с САПР ТП и осуществляет общую обратную связь (ΔD) с системой управления организацией. Наличие таких связей является достаточным для управления всеми подсистемами АСУ ТП и интегрированной системой в целом, учитывая степень формализации проектных процедур и процедур оценки решений. По мере накопления практического опыта обратные связи внутри АСУ ТП могут быть перераспределены: часть связей ΔL_2 (от подсистемы КПГ к подсистеме C) может быть замкнута на подсистему O, а некоторые параметры связи $\Delta Z_2^c(t)$ могут передаваться в подсистемы B и O.

Результаты исследования позволили разграничить функции подсистем, оптимизировать информационные потоки и разработать формальные алгоритмы процедур на основе нечеткой логики. Предложенная методика и разработанные программные средства, зарегистрированные в Роспатенте РФ в реестре программ для ЭВМ (авторы: Бурдо Г.Б., Палюх Б.В., Баженов А.Н.), реализуются на одном из предприятий г. Твери, изготавливающем геофизические приборы малыми партиями [3, 4].

Литература

1. Вумек Джеймс П., Джонс Даниел Т. Бережливое производство. Как избавиться от потерь и добиться процветания вашей компании. М.: Альпина Бизнес Букс, 2008. 716 с.

2. Бурдо Г.Б. Принципы построения автоматизированной системы управления технологическими процессами в многономенклатурных производствах // Вестн. Саратов. гос. тех. ун-та. 2010. № 3 (48). С. 113–118.

3. Бурдо Г.Б. Интеллектуальная система управления технологическими процессами в многономенклатурном машиностроительном производстве // Проблемы информатики. 2011. № 1 (9). С. 51–55.

4. Бурдо Г.Б. Оперативное планирование на основе приоритетных схем в производственных системах геофизического приборостроения // Каротажник. 2011. № 6 (204). С. 92–100.

УДК 004.7

ПОСТРОЕНИЕ КОРПОРАТИВНОЙ СЕТИ МЕТАЛЛУРГИЧЕСКОГО ПРЕДПРИЯТИЯ

Ю.М. Лисецкий, к.т.н.
(«ЭС ЭНД ТИ УКРАИНА», г. Киев, Украина, Iurii.Lisetskyi@snt.ua)

Описано построение корпоративной магистральной мультисервисной сети металлургического предприятия. Сформулирован набор требований к современным корпоративным магистральным сетям, соответствующей им инфраструктуре и функциональности систем. Приведена последовательность задач, решаемых в ходе их интеграции. Описан опыт реализации проекта.

Ключевые слова: *управление ресурсами, мультисервисная сеть, территориально распределенные системы, гетерогенная структура, универсализация и стандартизация.*

Для управления внутренними и внешними ресурсами крупных промышленных предприятий все чаще в качестве интегрированных систем используются ERP-системы (Enterprise Resource Planning System). Как правило, они строятся на централизованной БД и современной *информационно-технологической* (ИТ) инфраструктуре, поскольку без этого предприятия такого масштаба, особенно территориально распределенные, не могут полноценно функционировать. Именно поэтому в последние годы увеличилось количество проектов по построению современных ИТ-инфраструктурных решений в промышленности.

Рассмотрим проблему на примере ОАО «АрселорМиттал Кривой Рог» (АМКР), крупнейшего предприятия горно-металлургического комплекса Украины с полным металлургическим циклом. Предприятие состоит из 5 крупных производственных площадок, занимающих общую территорию около 25 кв. км, число его сотрудников превышает 45 тысяч.

В соответствии с корпоративными политиками в качестве современной ERP-системы выбрано решение от SAP AG.

Однако перед ее внедрением вместо существующей гетерогенной ИТ-среды потребовалось создать совершенно новую, унифицированную ИТ-среду, позволяющую предоставлять сотрудникам предприятия необходимые ИТ-сервисы с обеспечением их непрерывности; расширять их перечень за счет принципиально новых средств доступа к информации предприятия и платформы коллективной работы; стандартизировать и консолидировать ИТ-службы; понизить затраты на управление ИТ-ресурсами. Для реализации данной концепции необходимо было построить современный *центр обработки данных* (ЦОД), консолидирующий вычислительные мощности предприятия и единой корпоративной сети передачи данных (рис. 1).

Основными задачами внедрения современной ИТ-инфраструктуры стали выполнение требований корпоративных политик компании; построение оптической магистрали для консолидации ресурсов и сетей; внедрение отказоустойчивого ядра сети с пропускной способностью 10 Gbps; внедрение отказоустойчивого ЦОД; обеспечение контроля безопасности доступа к ЦОД; консолидация БД-приложений на платформе IBM; обеспечение полной управляемости сетевой инфраструктуры; возможность обнаружения и нейтрализации атак в корпоративной сети; интеллектуальное управление сетью на всех уровнях.

В качестве технологической основы корпоративной сети была выбрана стратегия самозащищающейся сети компании Cisco Systems (Cisco Self-Defending Network strategy), максимально точно отражающая ИТ-потребности предприятия, определенные в корпоративных политиках.

Специалисты «ЭС ЭНД ТИ УКРАИНА», имеющие опыт реализации подобных проектов (см. [1, 2]), выполнили работы по проектированию и построению площадок инсталляции сетевого и вычислительного оборудования, оптических линий, спроектировали и внедрили собственно сеть, а также системы сетевой безопасности и управления. Проект был реализован на основе сетевого оборудования Cisco. Ядро сети построено на базе Catalyst серии 65xx, характеризующейся высокой масштабируемостью и производительностью, поддержкой карт 10 Gbps, широким спектром

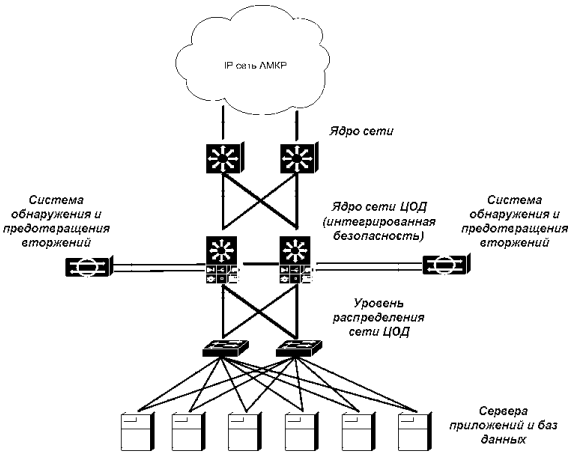


Рис. 1. Архитектура ЦОД

ethernet-карт и сервисных карт, конвергенцией услуг (данные, голос, видео), высокой надежностью и отказоустойчивостью (возможность резервирования элементов управления и питания).

Уровень распределения построен на базе Cisco Catalyst 45xx. Это неблокируемая коммутация 2–4-го уровней, поддержка интерфейсов 10 Gbps, модульность и широкий спектр ethernet-карт, конвергенция услуг, высокая надежность и отказоустойчивость.

Платформа Catalyst серии 65xx, используемая, в частности, в ЦОД для обеспечения отказоустойчивости на всех уровнях, стала основой для ядра сети и собственно ЦОД совместно с Catalyst 4948; платформы Catalyst 3750 и 3560 обеспечили уровень доступа к сети (рис. 2).

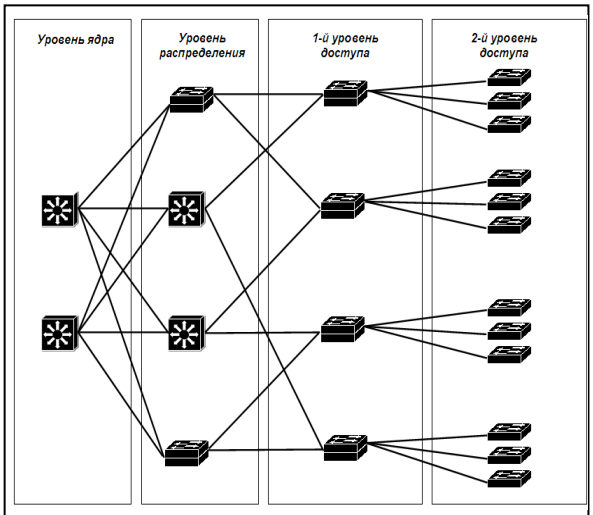


Рис. 2. Архитектура уровня доступа

Были внедрены также система управления локальными сетями CiscoWorks LMS, система мониторинга сетевой безопасности CiscoMARS, системы Cisco Security Manager и Intrusion Prevention System. В комплексе это позволило корпоративной сети выполнить требования концепции Cisco Self-Defending Network: обеспечены централизация сети и ее полная управляемость из единого центра, интеллектуальный контроль доступа, возможность обнаружения и нейтрализации атак в корпоративной сети предприятия; обеспечены высокая надежность и отказоустойчивость на всех участках сети, непрерывный мониторинг ее состояния.

В результате реализации проекта за 6 месяцев построена магистральная корпоративная мультисервисная сеть «АрселорМиталл Кривой Рог» (Украина) с пропускной способностью ядра в 10 Gbps и протяженностью более 120 км волоконно-оптического кабеля (рис. 3). Всего в рамках проекта консолидировано более 30 серверов в ЦОД; активное и пассивное оборудование установлено на более чем 120 сайтах предприятия; реализовано более 5 000 портов на активном сетевом оборудовании и более 1 200 портов СКС, а резервированная мощность электропитания превысила 120 кВА.

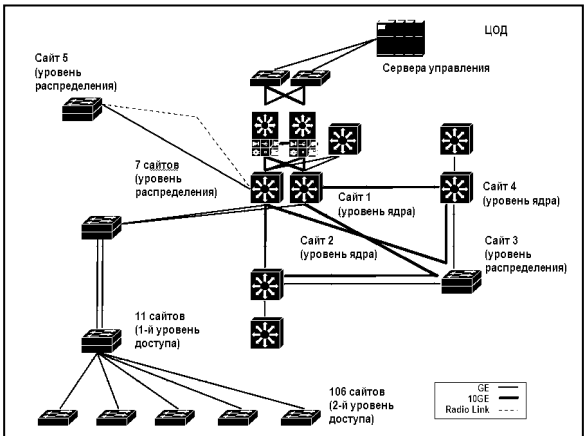


Рис. 3. Архитектура корпоративной сети

Таким образом, разработанная современная мультисервисная корпоративная сеть обеспечивает унификацию и централизацию сетевой и вычислительной инфраструктур предприятия, а также высокие управляемость, отказоустойчивость и защищенность сетевой и вычислительной инфраструктур. Созданнная ИТ-инфраструктура позволяет внедрять современные системы управления ресурсами предприятия на базе программных решений SAP AG.

В реализации проекта использован и дополнен накопленный опыт по формированию и обоснованию набора требований и соответствующей им функциональной структуры систем, последовательности задач, решаемых в ходе их интеграции, практического применения разработанного математического аппарата при выборе компонент систем из представленного на рынке набора программных и аппаратных средств [3–5]. Технологии и подходы к решению задач, примененные в этом проекте, в значительной степени универсализированы, что дает возможность применять их при построении подобных интегрированных территориально распределенных систем с гетерогенной структурой.

Литература

1. Лисецкий Ю.М. Опыт построения корпоративной интегрированнной информационной системы // Программные продукты и системы. 2007. № 2. С. 26–29.

2. Лисецкий Ю.М. Построение оптической транспортной магистрали оператора связи // Программные продукты и системы. 2010. № 4. С. 142–145.

3. Лисецкий Ю.М. Реализация методики комплексной количественной оценки качества сложных систем в программном комплексе «Вердикт». М.: ЦВСИТ ИМВС РАН, 2005. 26 с.

4. Лисецкий Ю.М. Метод комплексной экспертной оценки для проектирования сложных технических систем // Математические машины и системы. 2006. № 2. С. 141–147.

5. Лисецкий Ю.М. Выбор сложных систем по критерию минимума среднего риска // УСиМ. 2007. № 3. С. 22–26.

УДК 622:681.3

ИННОВАЦИОННЫЙ ПОДХОД К ОПРЕДЕЛЕНИЮ СТРУКТУРЫ ПРОГРАММНЫХ РЕШЕНИЙ ДЛЯ БИЗНЕС-ПРОЦЕССОВ ПРОМЫШЛЕННОГО ПРЕДПРИЯТИЯ

Н.И. Федунец, д.т.н.; А.Н. Гончаренко
(Московский государственный горный университет, gan@ngs.ru)

Представлен инновационный подход к созданию единой структуры программных решений для повышения эффективности функционирования промышленного предприятия. Интеграция разработанной модели определения оптимальной структуры программно-инструментальных средств в бизнес-процессы промышленного предприятия позволила улучшить технико-экономические, организационные и финансовые показатели.

***Ключевые слова**: информационные технологии, эффективность бизнес-процессов, промышленное предприятие, инновационный подход.*

Кризисные явления в мировой экономике привели к ужесточению конкуренции на рынке и обострили проблему сохранения конкурентоспособности отечественных предприятий и объединений. Чтобы удержать и усилить свои позиции, предприятиям необходим резерв в производственно-экономической сфере и безопасности промышленного производства.

Значительную роль в сохранении конкурентоспособности промышленных предприятий играет внедрение *информационных технологий* (ИТ). ИТ-подразделения стали полноценным структурным элементом системы управления, влияющим на конкурентные преимущества и прибыль предприятия.

Однако специфика промышленного производства заключается в том, что при подготовке информационного сопровождения производственных бизнес-процессов должны использоваться весьма разнородные модели, методы и алгоритмы обработки информации. Для этого необходимо иметь очень большое количество узкоспециализированных программных продуктов и связующих программных компонентов либо сложную ИТ-систему, охватывающую все информационные аспекты производственной жизни предприятия. В чистом виде эти варианты практически не встречаются. Причиной являются изначально функционально ограниченные программы, расширяющие сферы своего действия на смежные участки производственного предприятия, или крайняя сложность проектирования, настройки и обслуживания такой системы.

Существующее состояние ИТ-систем и программных продуктов не может обеспечить эффективное развитие промышленного предприятия. Это обстоятельство предопределило необходимость разработки инновационного подхода к созданию оптимальной структуры информационных технологий, учитывающих специфические особенности бизнес-процессов промышленного предприятия.

В данной работе предложена методика создания оптимальной структуры информационных технологий, интегрируемых в производственные бизнес-процессы предприятия, апробация которой проводилась в угольной компании ОАО «Гуковуголь» (Ростовская обл.). Методика состоит из пяти основных этапов.

1. Формирование множеств проектов программы развития горного предприятия, бизнес-процессов горного предприятия и информационных технологий.

2. Факторный анализ взаимосвязей показателей эффективности бизнес-процессов и параметров интеграции информационных технологий.

3. Формирование множества показателей эффективности и оптимизационных критериев функционирования бизнес-процессов предприятия.

4. Обоснование выбора информационных технологий для внедрения в бизнес-процессы горнопромышленного предприятия.

5. Определение оптимальной структуры информационных технологий, интегрируемых в бизнес-процессы промышленного предприятия.

На первом этапе сформирован информационный базис, позволяющий установить соответствие проектов программы развития определенным бизнес-процессам горного предприятия на основе свойств информационных технологий и возможности внедрения каждой информационной технологии в конкретный бизнес-процесс.

Информационный базис проектов программы развития промышленного предприятия: $\{PR_i\}=\{PR_1\}\cup\{PR_2\}\cup\{PR_3\}$, i=1, 2, 3 (где $\{PR_1\}$ – множество проектов, обеспечивающих выпуск товарной продукции; $\{PR_2\}$ – множество проектов, обеспечивающих повышение эффективности предприятия; $\{PR_3\}$ – множество проектов для удовлетворения требований государственных органов управления); бизнес-процессов промышленного предприятия: $\{Bp_j\}=\{Bp_1\}\cup\{Bp_2\}$, i=1, 2 (где $\{Bp_1\}$ – множество основных бизнес-процессов; $\{Bp_2\}$ – множество вспомогательных бизнес-процессов) и информационных технологий: $\{It_k\}=\{It_1\}\cup\{It_2\}\cup\{It_3\}$, i=1, 2, 3 (где $\{It_1\}$ – множество модулей TECHBASE Professional Mining;

$\{It_2\}$ – множество модулей систем диспетчеризации; $\{It_3\}$ – множество модулей Data Mining).

На втором этапе с помощью методов системного факторного и корреляционного анализа были определены бизнес-процессы горного предприятия, наиболее чувствительные к внедрению информационных технологий [1].

Третий этап методики направлен на формирование множеств оценочных критериев и показателей эффективности бизнес-процессов. На основе методов корреляционного анализа выявлены зависимости показателей эффективности бизнес-процессов и показателей эффективности функционирования промышленного предприятия.

Пусть $X = \{x_i^k, x_j^k\}$, $\forall$ i=1, …, 13, j=1, …, 10, k=1, 2 – показатели эффективности бизнес-процессов, тогда множество $X = \left\{Рэф1_i^{ИБ}\right\} \cup \cup \left\{Рэф2_j^{ИБ}\right\}$, i=1, 2, …, 13, j=1, …, 10, позволяет определить следующие показатели эффективности функционирования промышленного предприятия: длительность полного производственного цикла $f_{T_{БП}}(X) \to \to \min\limits_{x_i^k, x_j^k \in X}$; производительность готовой продукции промышленного предприятия $f_Q(X) \to \max\limits_{x_i^k, x_j^k \in X}$; себестоимость готовой продукции $f_C(X) \to \min\limits_{x_i^k, x_j^k \in X}$.

Явный вид этих функций находится методом группового учета аргументов. Качество получаемых зависимостей целевых функций от показателей эффективности бизнес-процессов, входящих в информационный базис, определяется коэффициентом детерминации (R^2), средним квадратом ошибки и F-критерием [2].

На четвертом этапе определялись информационные технологии, влияющие на показатели эффективности бизнес-процессов. Реализация этого этапа предполагает использование методов принятия решений в условиях неопределенности [3].

На заключительном этапе методики определяется оптимальная структура информационных технологий, интегрируемых в бизнес-процессы промышленного предприятия.

Проектирование структуры программно-аппаратных средств $\{S\}$ можно представить в виде совокупности некоторых программных решений множества $\{G\} \in \{It_k\}$ и взаимосвязей между этими программными продуктами из множества $\{Q\}$.

Отсутствие связей между программными продуктами говорит о невозможности влияния на функционирование программного решения множества $\{G\}$ этого же решения. Взаимозаменяемость программных средств при создании структуры программно-аппаратных средств означает, что из подмножества $\{G_i\}$ можно применить любое из решений, а из подмножества $\{Q_i^j\}$ любую реализуемую на практике связь. При этом любое из решений множества $\{G\}$, как и любая связь из множества $\{Q\}$, может отсутствовать в проектируемой структуре программно-аппаратных средств $\{S\}$ [2].

Математические модели M_i функционирования структуры программно-аппаратных средств $\{S\}$ в разных условиях эксплуатации, их сложность, инвариантность, полнота и адекватность во многом определяются свойствами множеств $\{G\}$ и $\{Q\}$:

$M_i = M_i(G_1, \ldots, G_n; Q_1^2, \ldots, Q_1^n; \ldots; Q^1{}_k, \ldots, Q^n{}_k; \ldots; Q^1{}_n, \ldots, Q_n^{n-1})$.

Для решения задачи структурно-параметрической оптимизации в целом предлагается принцип дискретного изменения программных решений структуры программно-аппаратных средств $\{S\}$ из множества $\{G\}$ и связей из множества $\{Q\}$, то есть принцип перебора различных структурных схем и определение оптимального параметрического решения для каждого возможного исполнения структуры программно-аппаратных средств. Дальнейший анализ оптимальных решений для всех возможных конструктивных исполнений структуры $\{S\}$ позволяет окончательно принять научно обоснованное решение, которое на практике можно реализовать с помощью динамического программирования [3].

Для выбора оптимальной структуры программно-аппаратных средств $\{S\}=y_N(x)$ из N вариантов ИТ с показателями эффективности бизнес-процессов $\{x_1, x_2, \ldots, x_N\}$, которые после внедрения ИТ соответственно $\{S_1(x_1), S_2(x_2), \ldots, S_N(x_N)\}$, можно записать:

$\{S\}=y_N(x)=f(x_1, x_2, \ldots, x_N)=$

$=S_1(x_1)+S_2(x_2)+\ldots+S_N(x_N)$.

Сначала согласно принципу оптимальности следует установить оптимальные параметры N-го программно-аппаратного средства, затем (N-1)-го и т.д.

Пусть $f_N(x)$ выражает оптимальную эффективность бизнес-процесса, образующуюся на N-й структуре программно-аппаратных средств. Очевидно, что $f_N(0)=0$, $f_1(x_1)=S_1(x_1)$ для $x>0$ [4]. Тогда основное функциональное уравнение динамического программирования записывается в виде

$\{S\}=y_N(x)=f_N(x)=\text{opt}[S_N(x_N)+S_{N-1}(x_{N-1})+\ldots+S_1(x_1)]=$

$=\text{opt}[S_N(x_N)+f_{N-1}(x-x_N)]$ для N=2, 3, …

Предприятие, на котором апробировалась методика создания системы ИТ, из-за несовершенства учетной политики на ряде бизнес-процессов столкнулось с нерациональным использованием материальных и других производственных ресурсов, что сказалось на его финансово-экономическом положении.

Факторный анализ показателей эффективности бизнес-процессов позволил выделить бизнес-процессы горнопромышленного предприятия, наиболее чувствительные к внедрению ИТ, – добычные работы и транспортировка горной массы.

Для формирования исходного информационного базиса было выбрано более 100 сходных показателей бизнес-процессов, наиболее чувствительных к внедрению ИТ.

В исходный информационный базис вошли показатели, коэффициент корреляции между которыми меньше 0,3, то есть взаимосвязь незначительная [4].

Таким образом, в ходе прикладного исследования системных связей и полученных закономерностей был построен исходный информационный базис значимых показателей эффективности бизнес-процессов, наиболее чувствительных к интеграции ИТ, что позволяет описать влияние этих показателей на эффективность функционирования горнопромышленного предприятия в целом.

Показатели эффективности бизнес-процессов, входящие в информационный базис:

- $\left\{Рэф1_i^{ИБ}\right\}$, *i=1, 2, … , m,* для добычных работ: x_1^1 – объем добычи (факт.), x_2^1 – фактические потери при добыче, x_3^1 – объем добычи (план.), x_4^1 – плановые потери при добыче, x_5^1 – плановое разубоживание, x_6^1 – фактические показатели разубоживания, x_7^1 – себестоимость по добыче (руб.), x_8^1 – удельная себестоимость по добыче (руб./т), x_9^1 – численность промышленно-производственного персонала, x_{10}^1 – производительность по добыче (т/смену), x_{11}^1 – объем подготовительных выработок (п.м), x_{12}^1 – объем нарезных выработок, x_{13}^1 – глубина залегания;
- $\left\{Рэф2_j^{ИБ}\right\}$, *j*=1, 2, …, *k*, для транспортировки горной массы: x_1^2 – списочная численность *автотранспортных средств* (а/с), x_2^2 – загрузка кузова, x_3^2 – коэффициент выхода на линию, x_4^2 – количество а/с в работе, x_5^2 – плановое количество рабочих смен, x_6^2 – коэффициент использования самосвалов, x_7^2 – время в движении а/с, x_8^2 – количество рейсов в смену на 1 а/с, x_9^2 – объем транспортировки, x_{10}^2 – индекс выполнения планового задания.

Методом группового учета аргументов определяется зависимость между показателями эффективности функционирования предприятия (см. табл.). В таблице представлены следующие регрессионные модели:

модель 1: $f_C(X)$=356341,7–3307,9$\cdot x_1^1$+12,32$\cdot(x_1^1)^2$– –0,102$(x_9^1)^2$ + 194,6$\cdot x_9^2$ – 6,83$\cdot 10^{-3}\cdot(x_1^2)^2$ – 0,29$\cdot x_6^1\cdot x_9^2$ – –0,84$\cdot x_4^1\cdot x_{10}^2$;

модель 2: $f_Q(X)$=3958,78+103,05$\cdot x_{13}^1$+0,57$\cdot x_1^1$ – –392,32x_6^2+5,18$\cdot x_6^2\ x_4^2$ + 0,95$\cdot x_6^1$ – 1,37$\cdot 10^{-3}\cdot x_2^1\cdot x_{13}^1$ – –0,0179$\cdot x_3^2\cdot x_5^1$;

модель 3: $f_{T_{БП}}(X)$ =25711–96,58$\cdot x_3^1$–14,45$\cdot x_9^1\ x_{10}^1$ + +3,02$\cdot x_{11}^1\ x_{12}^1$+88,47x_9^2+0,79x_2^2+1,43$\cdot 10^{-3}\cdot x_6^1$.

Зависимость целевых функций от показателей эффективности бизнес-процессов, входящих в информационный базис

Регрессионная модель	Коэффициент детерминации R^2	Проверка F-критерием			СКО
		$F_{набл}$	$F(m; n)_т$	$p_{ур}$	
Модель 1	0,881	m=7; n=70			5,05
		0,061	0,085	0,001	
Модель 2	0,942	m=7; n=70			3,56
		0,059	0,085	0,001	
Модель 3	0,868	m=6; n=70			6,52
		0,059	0,063	0,001	

Методом динамического программирования была определена структура информационных технологий, оказывающих влияние на показатели эффективности бизнес-процессов [5] на четырех шахтах горного предприятия.

Сравнительно-сопоставительный анализ результатов интеграции ИТ показал значительное изменение показателей эффективности функционирования как отдельно взятой шахты, так и горного предприятия в целом.

Реализация данной методики позволила улучшить основные показатели эффективности функционирования горнопромышленного предприятия, такие как время производственного цикла, производительность горнопромышленного предприятия, себестоимость руды, численность персонала.

Литература

1. Гончаренко А.Н. Разработка методики комплексной оценки ИТ-проектов на промышленном предприятии: Методы управления потоками в транспортных системах // Сб. науч. тр. МАДИ (ГТУ). М., 2009. С. 83–94.

2. Федунец Н.И., Гончаренко С.Н. Проблемы повышения производственного потенциала горнорудных предприятий по добыче медно-никелевых руд // Горный информ.-аналит. бюлл. 2006. № 9. С. 189–196.

3. Васильев Ф.П. Численные методы решения экстремальных задач. М.: Наука, 2008.

4. Федунец Н.И., Гончаренко С.Н. Оценка возможности управления производственными параметрами основных технологических циклов горнодобывающего предприятия // Горный информ.-аналит. бюлл. 2007. № 9.

5. Исследование операций в экономике; [под ред. Н.Ш. Кремера]. М.: ЮНИТИ, 2009.

УДК 681.3.06:004.942

МОДЕЛИРОВАНИЕ СКЛАДСКОЙ ЛОГИСТИКИ: РАЗРАБОТКА И КОМПЛЕКСИРОВАНИЕ В ORLANDO TOOLS

О.Ю. Башарина (Иркутский государственный университет, basharinaolga@mail.ru);
С.А. Горский, к.т.н.
(Институт динамики систем и теории управления СО РАН, г. Иркутск, gorsky@icc.ru)

Рассматриваются вопросы моделирования современных логистических складских комплексов на основе многовариантных экспериментов в параллельной вычислительной системе. Пакет моделирования разработан в инструментальном комплексе Orlando Tools, объектно-ориентированная система управления базой знаний которого обеспечивает комплексирование по данным для всех пакетов прикладных программ, создаваемых с помощью этого инструментального комплекса.

Ключевые слова: *складская логистика, моделирование, параллельные вычисления.*

В настоящее время специалистам различных отраслей производства, бизнеса и управления предлагается широкий спектр экономико-математических моделей. Сложность динамической структуры современных экономических объектов, обусловленная большим количеством важных характеристик процессов их функционирования и связей между ними, требует построения согласованного семейства моделей для исследования этих объектов на разных уровнях детализации и зачастую приводит к значительным техническим и методическим трудностям использования такого семейства моделей.

Современные программные комплексы класса Warehouse Management System (WMS) (Фолио WMS, AZ.WMS, SmartStock.WMS, Solvo.WMS, Radio Beacon WMS и др.), а также логистические системы класса Enterprise Resource Planning (ERP) (например Microsoft Axapta) являются мощным инструментом управления складскими процессами. Однако зачастую в этих комплексах и системах до сих пор используются методы и средства последовательного моделирования. С ростом сложности модели время расчетов при последовательном моделировании значительно увеличивается и может занимать от нескольких дней до нескольких месяцев. Вследствие этого, исходя из необходимости принятия оперативных решений по управлению экономическим объектом на основе анализа его математической модели, приоритет получают средства параллельного или распределенного моделирования [1].

В статье рассматривается пакет моделирования современных *логистических складских комплексов* (ЛСК), основанный на методологии модульного программирования и ориентированный на проведение расчетных работ в параллельных вычислительных системах. Использование модульного подхода обеспечивает ряд важных возможностей. Во-первых, достаточно гибкую модификацию и безболезненное развитие математического и программного базиса для моделирования ЛСК посредством добавления или замены модулей (программ) этого базиса новыми модулями, в том числе модулями уже разработанных библиотек программ. Во-вторых, быструю точечную реализацию дополнительных средств моделирования процессов функционирования ЛСК, не представленных в используемых системах управления этими комплексами.

ЛСК предоставляет клиентам $cl_i \in Cl$, i=1, 2, …, M, логистические услуги $u_j \in U$, j=1, 2, …, N, на множестве товаров $f_s \in F$, s=1, 2, …, R. Для элементов множеств Cl, U, F имеются соответствующие наборы атрибутов, значения элементов определяются значениями их атрибутов. Каждому клиенту присваивается категория, в соответствии с которой назначается уровень обслуживания $l_n \in L$, n=1, 2, …, K. Каждая услуга u_j относится к одной из категорий $w_k \in W$, k=1, 2, …, B, а каждый товар f_s – к какой-либо группе товаров $gr_p \in GR$, p=1, 2, …, A.

Заданы матрицы C и V размерности $N \times M$, соответственно определяющие для клиента цены cl_i и объем услуги u_j.

Предприятие несет расходы $Z(\tau)$ по предоставлению услуг: $Z(\tau) = \sum_{k=1}^{m1} \sum_{i=1}^{n} \alpha_{ki}(\tau)$, где $\alpha_{ki}(\tau)$ – количественный показатель k-го компонента расходов на обеспечение услуги u_i в момент времени τ.

Заданы также матрицы ST, $ST1$, $V1$ с размерностью $R \times M$, которые соответственно определяют стоимость, себестоимость и объем товаров f_s клиента cl_i.

ЛСК требуется определять вероятные планы оказания услуг, их сроков и возврата для учета возможных финансовых поступлений. Для анализа эффективности реализации проекта и внесения корректирующих воздействий в ходе его выполнения предприятию необходимо проводить расчеты финансовых показателей по предоставлению услуг за определенный период времени t.

Основные финансовые показатели функционирования ЛСК, такие как оценка потенциальной мощности услуг $E(t)$, мощность освоенных услуг $y(t)$, оценка мощности неосвоенных услуг $E_н(t)$, суммарный доход $H(t)$, суммарные затраты $G(t)$, прибыль $P(t)$ и рентабельность $R(t)$, рассчитыва-

ются методами численного интегрирования и определяются соотношениями следующего вида: $E(t)=\int_{\tau_0}^{t} e(\tau)\,d\tau$, где $e(\tau)$ – оценка потенциальной интенсивности оказания услуг ЛСК в момент времени τ; $y(t)=\int_{\tau_0}^{t}(u(\tau)-w(\tau))\,d\tau$, где $u(\tau)$ и $w(\tau)$ – соответственно объем текущих и завершенных услуг ЛСК в момент времени τ; $E_н(t)=E(t)–y(t)$, $H(t)=\int_{\tau_0}^{t}\sum_{i=1}^{k} h_i(\tau)\,d\tau$, где $h_i(\tau)$ – объем доходов от i-го вида услуги в момент времени τ; $G(t)=\int_{\tau_0}^{t}\sum_{i=1}^{n} z_i(\tau)\,d\tau$, где $z_j(\tau)$ – объем расходов на реализацию j-го вида услуги в момент времени τ; $P(t)=H(t)–G(t)$, $R(t)=P(t)/G(t)$.

Для анализа деятельности ЛСК необходимо оценить финансовые показатели и стабильность его функционирования с позиции долгосрочной перспективы [2]. Финансовые результаты деятельности предприятия характеризуются суммой полученной прибыли и уровнем рентабельности:

– прибыль от реализации услуг P_u;

– внереализационные доходы и расходы P_v;

– показатели рентабельности (рентабельность услуг R_u и рентабельность оборота R_{ob});

– $F(P_u, P_v, R, R_{ob})$.

Причем крайне важно проанализировать динамику прибыли ЛСК от реализации отдельных видов услуг $(P_u)_i$.

Внереализационные доходы и расходы делятся на следующие категории:

– инвестиционные доходы;

– финансовые расходы;

– прочие (прибыль и убытки прошлых лет).

Финансовая устойчивость предприятия связана прежде всего с общей финансовой структурой и степенью его зависимости от кредиторов и инвесторов. Поэтому необходимо провести анализ:

– структуры источников средств;

– расходов, связанных с обслуживанием внешних источников средств;

– источников материальных оборотных средств.

Для анализа структуры источников средств необходимо рассчитать коэффициенты капитализации, такие как коэффициент финансовой автономии $K_{авт}$, коэффициент финансового риска $K_{фр}$ и коэффициент маневренности собственного капитала $K_{ман}$. Расчет этих показателей можно провести на основе данных финансовой отчетности «Баланс» (форма № 1) по формулам: $K_{авт}=П_{соб}/АП$; $K_{фр}=П_{пр}/П_{соб}$; $K_{ман}=А_{об}/П_{соб}$, где $П_{соб}$ – собственный капитал; $АП$ – весь капитал; $П_{пр}$ – привлеченные средства; $А_{об}$ – собственные оборотные средства.

Для оценки расходов по обслуживанию внешних источников финансирования используются так называемые коэффициенты покрытия. К наиболее значимым относятся коэффициент структуры покрытия долгосрочных вложений $K_{пдв}$, коэффициент долгосрочного привлечения заемных средств $K_{дпзс}$, коэффициент финансовой независимости капитализированных источников $K_{фнки}$. Данные коэффициенты определяются следующими соотношениями: $K_{пдв}=П_{дс}/А_{ноб}$; $K_{дпзс}=П_{дс}/(П_{соб}+П_{дс})$; $K_{фнки}=П_{соб}/(П_{соб}+П_{дс})$; $K_{дпзс}+K_{фнки}=1$, где $П_{дс}$ – долгосрочные пассивы; $А_{ноб}$ – необоротные активы; $П_{соб}$ – собственный капитал.

Для характеристики источников формирования запасов используются несколько показателей:

– собственные оборотные средства $K_{соб}$;

– собственные оборотные средства и долгосрочные заемные источники формирования запасов $K_{сдз}=K_{соб}+K_{дз}$;

– общая величина основных источников формирования запасов $K_{общ}=K_{сдз}+K_{кр}–A_{ноб}$, где $K_{кр}$ – краткосрочные кредиты; $А_{ноб}$ – необоротные активы.

Трем показателям наличия источников формирования запасов соответствуют показатели обеспеченности запасов источниками формирования:

– излишек (+) или недостаток (–) собственных оборотных средств: $\pm Ф^{с}=K_{соб}–З$;

– излишек (+) или недостаток (–) собственных оборотных средств и долгосрочных заемных источников формирования запасов: $\pm Ф^{сдз}=K_{сдз}–З$;

– излишек (+) или недостаток (–) общей величины основных источников формирования запасов: $\pm Ф^{общ}=K_{общ}–З$.

В приведенных соотношениях $З$ – стоимость запасов.

С помощью этих показателей определяется трехкомпонентный показатель типа финансовой устойчивости: $S(Ф)=\begin{cases}1, \text{ если } Ф>0,\\ 0, \text{ если } Ф<0.\end{cases}$

Показатели S={1, 1, 1} и S={0, 1, 1} определяют соответственно абсолютную и нормальную финансовую устойчивость, а при S={0, 0, 1} и S={0, 0, 0} наблюдаются соответственно неустойчивое и кризисное финансовые состояния.

Решение поставленных выше задач требует проведения многовариантных расчетов. Причем, учитывая стохастичность моделируемых процессов, для каждого варианта нужно выполнить немало прогонов, чтобы добиться требуемой степени достоверности результатов. Многие параметры исследуемой предметной области представляют массивы данных, допускающих одновременную обработку их элементов. Перечисленные особенности вычислительного эксперимента и описания предметной области актуализируют применение парадигмы параллельного программирования в процессе решения поставленных ранее задач.

Пакет моделирования ЛСК создан на базе инструментального комплекса Orlando Tools [3], предназначенного для разработки пакетов прикладных программ, обеспечивающих возможность параллельной обработки структур данных в процессе проведения многовариантных расчетов при решении прикладных задач математического моделирования на однородных UNIX-кластерах.

Описание предметных областей пакетов, создаваемых с помощью Orlando Tools, осуществляется на основе объектно-ориентированной модели данных и поддерживается на программном уровне встроенной библиотекой классов объектов предметной области. Объектно-ориентированная система управления базой знаний Orlando Tools позволяет проводить комплексирование по данным для всех пакетов прикладных программ, разрабатываемых с помощью этого инструментального комплекса. Тем самым обеспечивается возможность использования в процессе создания нового пакета фрагментов описания предметных областей, функциональных программных модулей, исходных данных и результатов вычислений, имеющихся в других пакетах. В результате сокращаются сроки разработки пакетов прикладных программ и проведения вычислительных экспериментов. В частности, рассматриваемый пакет моделирования создан на основе комплексирования с пакетом R-SIM [4] для моделирования экономических показателей процесса реорганизации объектов коммерческой недвижимости предприятия.

В таблице представлены сравнительные результаты времени решения ряда задач на персональном компьютере и на вычислительном кластере МВС-1000/16 Института динамики систем и теории управления СО РАН (г. Иркутск), узлы которого имеют характеристики вычислительных ресурсов, аналогичные характеристикам персонального компьютера. Очевидно, что с повышением сложности задачи время расчетов на персональном компьютере существенно увеличивается, и поэтому требуется применение других, более производительных вычислительных средств.

Задача моделирования	Время решения задачи	
	на ПЭВМ	на кластере (16 узлов)
Расчет планируемых финансовых показателей работы ЛСК	1–2 мин.	1 мин.
Определение оптимального варианта сдачи в аренду объектов ЛСК	15–30 мин.	1–2 мин.
Планирование расписания обслуживания клиентов с учетом потока случайных заявок	1 ч.–3 сут.	5 мин.–5 ч.

В данной статье показан опыт применения высокоуровневых инструментальных средств разработки интеллектуальных пакетов для моделирования ЛСК, накопленный в лаборатории методов автоматизации научных исследований и учебного процесса Международного института экономики и лингвистики Иркутского государственного университета. Архитектура, принципы работы, способы и средства реализации инструментального комплекса Orlando Tools обеспечивают широкий спектр функциональных возможностей для моделирования других экономических объектов в самых различных сферах деятельности.

Литература

1. Fujimoto R.M. Parallel and Distributed Simulation Systems. NY: John Wiley & Sons, 2000.

2. Кононенко О., Маханько О. Анализ финансовой отчетности. Харьков: Фактор, 2006. 200 с.

3. Инструментальный комплекс ORLANDO TOOLS / Г.А. Опарин [и др.] // Программные продукты и системы. 2007. № 4. С. 63–65.

4. Горский С.А., Феоктистов А.Г. Создание пакетов программ в инструментальном комплексе ORLANDO TOOLS // Управление большими системами: сб. тр. II шк.-сем. молод. ученых. Воронеж: Научная книга, 2007. Т. 2. С. 33–39.

УДК 004.92:62-82

ПРИНЦИПЫ ПОСТРОЕНИЯ САПР ГИДРАВЛИЧЕСКИХ МАШИН И АППАРАТОВ

В.С. Крутиков, к.т.н.; К.А. Лиходед; В.В. Копица
(Южно-Российский государственный технический университет (Новочеркасский политехнический институт), klikh@mail.ru, vadimnpi@mail.ru)

Изложены принципы построения САПР, обеспечивающей комплексное решение задачи создания новых машин и аппаратов.

Ключевые слова: *гидравлические машины и аппараты, система автоматизированного проектирования, комплексное решение задач.*

С восьмидесятых годов прошлого века на высокотехнологичных предприятиях, выпускающих сложную наукоемкую продукцию, интенсивно внедряется информационная поддержка изделия

на всех этапах жизненного цикла (CALS – Continuous Acquisition and Life-cycle Support). Жизненный цикл изделия – перечень этапов, через которые проходит изделие за весь период своего существования. Термин CALS предполагает организацию единого информационного пространства, объединяющего автоматизированные системы. В интегрированной информационной среде, в которой осуществляется безбумажное информационное взаимодействие между всеми участниками жизненного цикла изделия, действуют стандартные правила хранения, обновления, поиска и передачи информации.

Одним из наиболее важных является этап проектирования, осуществляемого САПР. Принято выделять системы функционального проектирования – CAE (Computer Aided Engineering), конструкторского – CAD (Computer Aided Design) и технологического – CAM (Computer Aided Manufacturing).

С помощью интегрированной совокупности CAE/CAD/CAM создаются машины и аппараты, соответствующие современному уровню, при этом решается ряд взаимосвязанных задач:

– расчет системы гидропривода машины и определение параметров нового аппарата;

– расчет размеров деталей нового аппарата;

– анализ динамических характеристик аппарата, проверка его работы в гидроприводе машины и в случае необходимости корректировка размеров деталей;

– разработка конструкторской документации;

– разработка *технологического процесса* (ТП) изготовления деталей аппарата и программ для станков с ЧПУ.

Возможности современной вычислительной техники позволяют проектировщику создавать новый аппарат путем разработки единой САПР, состоящей из подсистем решения отдельных вышеперечисленных задач. В разрабатываемой САПР все вопросы должны решаться программно с минимумом диалоговых режимов работы, за исключением тех случаев, когда программное решение затруднительно или недостаточно информации. Все подсистемы САПР включают необходимые банки данных и знаний, оформленные в виде подпрограмм, обмен информацией между подсистемами осуществляется автоматически.

В этом заключается принципиальное отличие предлагаемого подхода от идеологии САПР, создаваемых в «Компас» и Solid Works, в основу которых положен принцип диалогового режима работы, не позволяющий в дальнейшем переходить от автоматизированного к полностью автоматическому проектированию.

В общем случае разрабатываемая САПР имеет структурную схему, состоящую из пяти подсистем: расчета гидропривода и выбора гидрооборудования – РП, гидропривода – Д, расчета размеров деталей – РА, автоматизированного получения конструкторской документации – КД, автоматизированного проектирования технологического процесса – ТП (рис. 1).

САПР РП состоит из программы rasgp.pas, написанной на языке Object Pascal и работающей в среде Delphi, базирующейся на ПО, содержащем банки данных по стандартному гидрооборудованию [1]. Эта подсистема создает текстовый файл xarob.txt, содержащий характеристику всего выбранного стандартного гидрооборудования, и файл parga.dat параметров нового гидроаппарата для автоматической передачи информации в САПР РА.

САПР РА нового аппарата должна начинаться с создания его расчетной схемы. Разработка САПР схемного решения подробно описана в работе [2]. Расчетная схема принимается по аналогии с существующими подобными аппаратами или создается проектировщиком вне рамок данной системы.

САПР РА состоит из программы rasga.pas, которая базируется на ПО, содержащем базовое ПО по механике жидкости [3], ПО по деталям машин, ГОСТы, математические и геометрические зависимости.

Программа rasga.pas составляется таким образом, что внутри ее создается единое, функционально связанное информационное пространство. В результате изменение любого размера в одной детали приводит к автоматическому пересчету размеров всех функционально связанных деталей, выбору из подключенных банков данных других

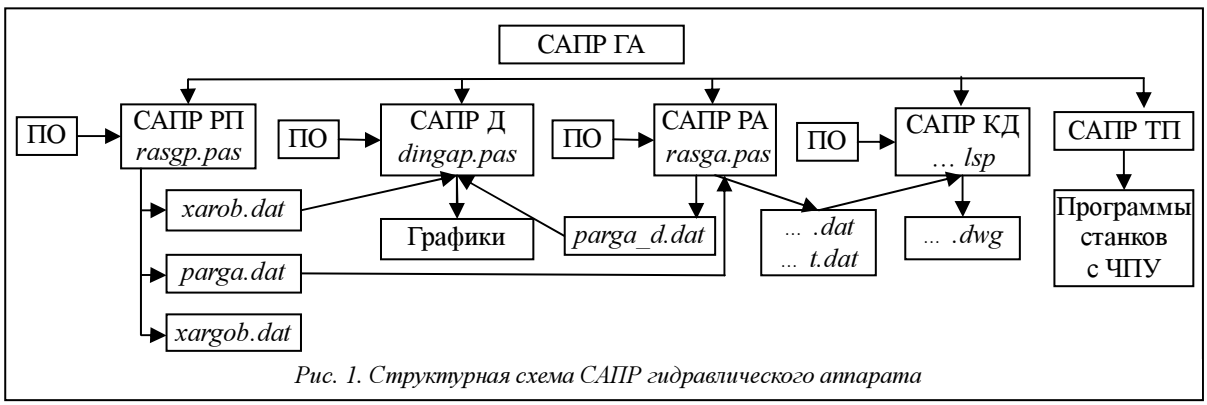


Рис. 1. Структурная схема САПР гидравлического аппарата

стандартных деталей и новой информации о требованиях к поверхностям. Эта подсистема создает: файл parga_d.dat для передачи информации в подсистему анализа динамических характеристик проектируемого гидроаппарата в составе САПР Д; файл .dat, содержащий всю информацию о размерах, допусках, шероховатостях для каждой детали; файл t.dat текстовой информации для автоматического заполнения основной надписи чертежа. Эти файлы передаются в САПР КД. Просмотрев рабочие чертежи, полученные в результате работы САПР КД, проектировщик в случае необходимости может вернуться в подсистему САПР РА, внести необходимую корректировку в расчет размеров какой-либо детали, запустив на выполнение программу rasga.pas, получить файлы измененных размеров всех деталей, по которым подсистема САПР КД в автоматическом режиме создаст рабочие чертежи нового варианта аппарата. В программе rasga.pas можно предусмотреть различные варианты отдельных узлов расчетной схемы. Этим создается возможность проведения экспертной или параметрической оптимизации разрабатываемого аппарата.

Подсистема САПР КД включает функции на языке AutoLISP, работающие в системе AutoCAD и в автоматическом режиме создающие чертежи деталей. Одновременно эти функции создают контуры деталей, которые будут входить в общий вид аппарата. Функция получения общего вида собирает его из готовых блоков деталей, устанавливая их в нужное место чертежа. При необходимости блоки деталей масштабируются и поворачиваются. Такой метод получения общего вида прост в реализации и является удобным и эффективным способом контроля правильности расчета размеров взаимосвязанных деталей и обмена информацией между подсистемами. ПО САПР КД состоит из функций вычерчивания типовых элементов гидравлических аппаратов и элементов чертежа, что значительно упрощает процесс написания функции получения чертежа деталей. Результатом работы САПР КД являются файлы чертежей .dwg и файлы .dat, содержащие координаты характерных точек детали и все технические требования к поверхностям, предназначенные для последующей передачи информации в САПР ТП, которая будет создавать программы работы станков с ЧПУ.

Подсистема анализа динамических характеристик аппарата и привода САПР Д состоит из программы dingap.pas, включающей динамическую модель проектируемого аппарата. Разработанное ПО статических и динамических программных моделей типовых аппаратов упрощает написание программы. Анализ получаемых графиков переходных процессов позволяет оценить динамические характеристики проектируемого аппарата. Если результаты не удовлетворяют проектировщика, он возвращается в САПР РА, изменяет размеры, получает новый файл parga_d.dat и снова запускает программу dingap.pas.

Отметим удобство использования программного комплекса моделирования гидроприводов машин различного назначения HydroCAD при разработке САПР Д [4]. Пользовательский интерфейс системы адаптирован для специалистов по гидроприводу. В HydroCAD использован подход, избавляющий проектировщика от системного программирования, автоматизирующий процесс проработки интерфейсов передачи данных и формирования списков формальных параметров вызова подпрограмм. Создаваемая в HydroCAD компьютерная модель исследуемого гидропривода представляет собой программную реализацию структурно-функциональной математической модели на макроуровне. Эта модель отображает происходящие в гидроприводе физические процессы и позволяет детально смоделировать его устройство и связи между составляющими элементами. Помимо вывода на экран переходных процессов всех необходимых выходных параметров непосредственно во время прогона модели, в HydroCAD реализованы поддержка двух- и трехмерной анимации и режим синхронизации модельного и реального времени.

Основой для создания САПР является наличие ПО решения всех задач проектирования. Специалисты в области каждой задачи, используя принцип декомпозиции сложной задачи и системный подход, могут выполнять разработку сложного и трудоемкого ПО. Примером послужит созданное ПО, которое позволяет полностью автоматизировать решение всех задач расчета, выбора и получения конструкторской документации цилиндрических пружин сжатия, применяемых практически в каждом гидроаппарате.

Разработанное ПО позволило создать САПР предохранительного клапана непрямого действия, гидропневмоаккумулятора с датчиком наличия жидкости, гидроцилиндра с торможением и без торможения, блока дистанционного управления, центробежных насосов (консольных, секционных и с двухсторонним подводом жидкости) [5].

Рисунок 2 представляет общий вид предохранительного клапана типа 510, а рисунок 3 – графики переходных процессов при работе клапана.

Достоинства создаваемой по таким принципам системы проявились при работе САПР центробежных секционных насосов. Система позволяет практически в автоматическом режиме проектировать насос на различные параметры и с различным

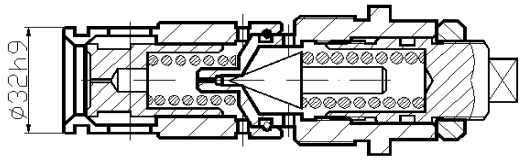


Рис. 2. Общий вид клапана 510

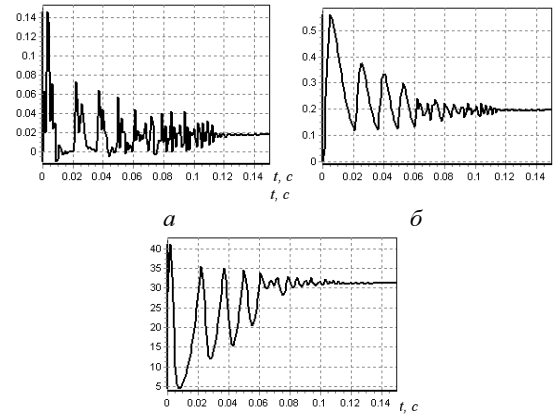


Рис. 3. Динамика работы клапана: а) перемещение клапана управления, мм; б) перемещение основного клапана, мм; в) давление перед клапаном, МПа

числом ступеней. В подсистемах САПР РА и КД реализовано разветвление вариантов конструкции узла фиксации подшипника на валу насоса и вариантов уплотнений крышки, которое позволяет назначать любое их сочетание.

При различных вариантах конструкции система автоматически меняет размеры, набор и форму деталей подшипникового узла. Выбор варианта конструкции узла по запросу системы определяет проектировщик.

Авторы работают над составлением программы функционально-стоимостного анализа вариантов фиксации подшипника на валу, которая определит оптимальный для различных параметров насоса, а подсистема КД в автоматическом режиме выдаст конструкторскую документацию. Запрограммировать проведение такого анализа можно только при совместной работе подсистем РА, КД и ТП, что и реализуется в созданной по изложенным принципам комплексной САПР центробежного секционного насоса.

Подсистема САПР КД, основанная на программированном черчении, позволяет легко собрать общие виды аппарата в разных режимах работы для проверки его нормального функционирования. Это является дополнительным методом проверки правильности расчета размеров деталей.

Следует отметить, что работа САПР РА при расчете аппарата для различных параметров не сводится к простому пропорциональному изменению размеров деталей, они принимаются из нормального ряда согласно ГОСТ, из банка данных автоматически получается информация о других значениях допусков, шероховатостей и прочего для новых параметров аппарата.

На примере САПР *гидроцилиндра* (ГЦ) подтверждена возможность проектирования аппарата не только для различных параметров, но и разной конструкции [6].

Подсистема САПР КД может создавать чертежи сложных конструкций. На рисунке 4 показан

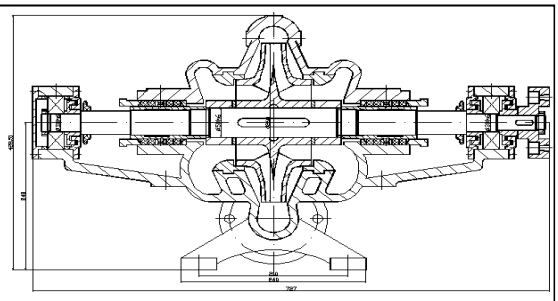


Рис. 4. Центробежный насос с двухсторонним подводом жидкости

общий вид центробежного насоса с двухсторонним подводом жидкости, полученный в результате работы САПР НД.

Приведенные примеры показывают, что разработанная по таким принципам САПР

– дает комплексное решение всех задач создания аппаратов;

– сокращает время разработки за счет параллельного выполнения работ по созданию подсистем РА, КД и ТП;

– позволяет по мере развития переходить от автоматизированного проектирования к автоматическому;

– полностью реализует идеологию безбумажного проектирования: вся информация находится на электронных носителях и передается автоматически файлами в подсистемы САПР, что минимизирует возможность ошибок, связанных с человеческим фактором.

Разработанное ПО делает задачу создания САПР посильной для любого проектировщика и дает ему эффективный инструмент проведения оптимизации разрабатываемого объекта.

Результаты работы созданных для ряда гидравлических машин и аппаратов САПР подтвердили правильность принятых принципов их построения.

Литература

1. Крутиков В.С. Разработка САПР гидросистем объемного гидропривода // Гидромеханика, гидромашины, гидропривод и гидропневмоавтоматика: тез. докл. Междунар. науч.-технич. конф. М.: МЭИ, 1996. С. 112.

2. Даршт Я.А. Методы и модели автоматизированного анализа и синтеза элементов гидропривода. Автореф. дисс... д.т.н. Ковров, 2005.

3. Крутиков В.С., Лиходед К.А. Базовое программное обеспечение компьютерного моделирования гидравлических машин и аппаратов // Изв. вузов, Сев.-Кав. регион. 2010. № 3. С. 15–19 (Технич. науки).

4. Анисимов А.В., Лиходед К.А. Программный комплекс моделирования гидроприводов различного назначения HydroCAD // Изв. вузов, Сев.-Кав. регион. 2010. № 4. С. 21–27 (Технические науки).

5. Крутиков В.С. САПР центробежных насосов общепромышленного назначения. Новые технологии управления движением технических объектов. Вып. 5: сб. стат. по матер. 7-й Междунар. конф. Новочеркасск, 2004. С. 119–122.

6. Крутиков В.С. САПР гидроцилиндра с торможением в начале и конце хода // Гидропневмоавтоматика и гидропривод-2005: сб. науч. тр. Т. 1. Ковров. 2006. С. 234–238.

УДК 004.415.23/25+418

УНИВЕРСАЛЬНАЯ ПРОГРАММА МОНИТОРИНГА ТЕХНОЛОГИЧЕСКИХ ПРОЦЕССОВ

А.Л. Калабин, д.ф.-м.н.; А.В. Козлов
(Тверской государственный технический университет, alex.ka.86@gmail.com);
Э.А. Пакшвер, к.т.н. (ОАО «Тенакс», г. Москва)

Описана архитектура программной системы мониторинга технологических процессов, состоящей из трех модулей: для создания схемы производственного процесса, для работы с хранилищами данных и для обработки и анализа данных. Работа приложения демонстрируется на примере процесса производства полиакрилонитрильных нитей.

Ключевые слова: *управление, мониторинг, производственный процесс, архитектура ПО, структура производственного процесса.*

Целью производства является получение однородной высококачественной продукции. Для этого необходимы четкая организация технологического процесса и его своевременное регулирование с учетом имеющихся обратных связей.

Основным инструментом установления обратной связи является мониторинг, заключающийся в систематическом или непрерывном сборе информации о параметрах объекта для определения тенденций их изменения.

Цель настоящей работы – создание универсального программного инструмента для технолога, осуществляющего сбор, обработку и анализ данных о технологическом процессе. Этот инструмент позволяет создавать из типовых элементов схему производственного процесса, адаптируя к нему систему. На основе схемы производственного процесса создаются хранилище данных и соответствующие инструменты для регистрации параметров и анализа накопленных данных.

Таким образом, для разработки программной системы мониторинга необходимо решить следующие задачи:

1) реализация инструмента для создания схемы производственного процесса на основе типовых элементов;

2) создание хранилища значений параметров производственного процесса и инструментов для регистрации этих значений;

3) создание инструментов для обработки и анализа данных о производственном процессе.

Приложение для мониторинга можно разбить на три модуля, каждый из которых предназначен для решения одной из поставленных задач. Задачи и характер их решения имеют различную природу и мало пересекаются между собой, что позволяет сделать модули минимально зависимыми друг от друга.

С точки зрения пользователя систему мониторинга можно рассматривать как отдельные приложения, используемые различными специалистами в разное время. Первым этапом работы с системой является создание схемы производственного процесса. Это выполняет технолог до начала эксплуатации системы. Затем операторы и обслуживающий персонал регистрируют значения отслеживаемых параметров. У пользователей нет необходимости изменять схему, поэтому данный модуль приложения можно воспринимать как электронный журнал для записи значений параметров. В процессе работы системы может возникнуть потребность изменить схему технологического процесса или свойства ее отдельных эле-

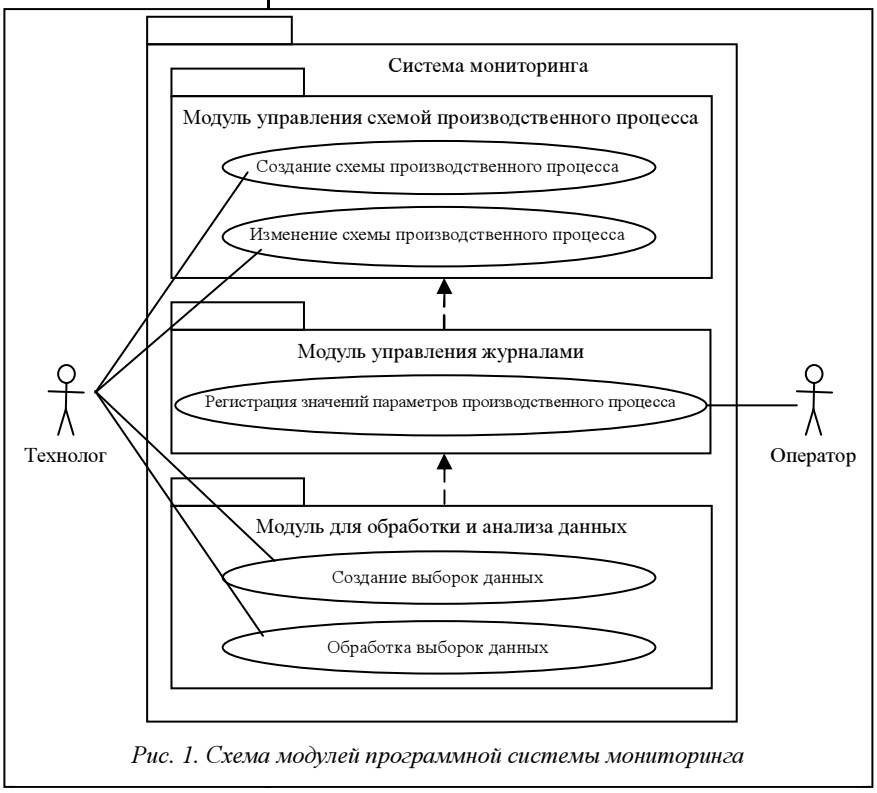


Рис. 1. Схема модулей программной системы мониторинга

ментов. Обработкой и анализом накопленных данных занимается технолог производства или научный консультант. Они используют третий модуль приложения для мониторинга, который представляет собой постоянно расширяемый набор инструментов для анализа данных. Модули приложения показаны на рисунке 1.

Приложение реализовано на платформе .Net на языке C#. Каждый модуль включает в себя несколько проектов, проекты каждого модуля реализуют модель данных, бизнес-логику и интерфейс пользователя. Для проектов, реализующих бизнес-логику различных модулей приложения, следует придерживаться правила, по которому модули приложения не должны зависеть от модулей более высокого уровня. То есть, например, модуль управления схемой производственного процесса не должен зависеть от модуля управления журналами.

Производственный процесс – это совокупность всех действий людей и средств производства, направленных на изготовление продукции [1]. Производственный процесс можно рассматривать как прохождение предмета труда через несколько стадий, в ходе которых он меняет свои свойства. Для обозначения любого предмета труда в приложении мониторинга используется термин «материал». В процессе производства может происходить превращение одного материала в другой или качественное изменение свойств какого-либо материала. Можно выделить три вида материала – сырье, продукция и полуфабрикаты.

Та часть производственного процесса, где происходят обработка и/или изменение материала, называется участком. В роли участка могут выступать аппарат, станок, цех или любое другое структурное подразделение предприятия. Главной характеристикой участка является то, что на нем обрабатывается материал. Каждый участок может включать в себя несколько однотипных аппаратов, станков и т.д., выполняющих одинаковые операции, такая часть участка называется линией.

Участок может иметь входы и выходы, называемые материальными потоками. Каждый поток характеризуется материалом, который перемещается с одного участка на другой. Материальные потоки соединяют участки в единый производственный процесс.

Схема производственного процесса включает в себя классы (см. рис. 2). (Диаграммы, приведенные в статье, являются эскизами и не отражают всех деталей системы. Специализированные коллекции, созданные для хранения объектов, классы исключений, реализуемые объектами интерфейсы и другие детали, не рассматриваемые в статье, опущены.)

Для решения задач мониторинга не нужно выделять все участки и материальные потоки производства. Необходимо отметить материальные потоки только в тех местах, где, во-первых, качество материала интересно с точки зрения влияния на качество конечной продукции, а во-вторых, имеются возможности для измерения параметров материала. Все технологические операции между выделенными материальными потоками объединяются в участки, даже если они включают в себя несколько различных машин и аппаратов.

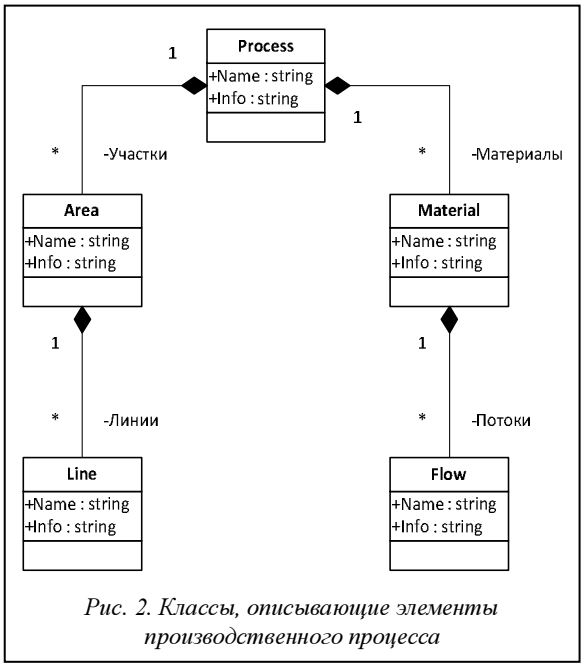


Рис. 2. Классы, описывающие элементы производственного процесса

Например, процесс производства углеродного волокна упрощенно можно представить так. Акрилонитрил смешивается с растворителем и подается в реактор. Там происходит процесс полимеризации, в результате получается прядильный раствор. Этот раствор подается в демономеризатор, в котором его свойства изменяются, а затем в формовочную машину. На выходе получается волокно.

С точки зрения предложенного подхода производственный процесс получения углеродного волокна состоит из трех участков – полимеризация, демономеризация и формование (рис. 3). В ходе производственного процесса обрабатываются четыре материала – акрилонитрил, растворитель, прядильный раствор и волокно. Движение этих материалов образует пять материальных потоков. Если на производстве имеется несколько реакторов полимеризации, то первый участок может состоять из множества линий.

Таким образом, в рамках системы мониторинга производственный процесс представляет собой последовательность участков, на вход которых подаются материалы, а на выходе получаются другие материалы или те же, но с измененными свойствами.

Как видно из диаграммы на рисунке 2, участок и материал – очень похожие объекты. Одним из будущих изменений системы может стать отказ от

Рис. 3. Элементы производственного процесса производства углеродного волокна

интуитивно непонятных терминов «участок», «линия» и «материальный поток». Проектирование схемы может сводиться к тому, что пользователь сначала создает коллекцию элементов схемы производственного процесса, а затем на основе собственных элементов собирает схему.

Основная задача предложенной системы – регистрация параметров производственного процесса. Атрибутами участков и материалов являются параметры, а параметрами линий и материальных потоков – параметры соответствующих участков и материалов. Кроме того, можно реализовать параметры, принадлежащие отдельным линиям или потокам. Параметры могут быть числовыми, строковыми, временными и пр.

В зависимости от типа элемента производственного процесса, которому принадлежит параметр, можно выделить:

1) входные/выходные параметры – параметры материалов и параметры материальных потоков;

2) параметры технологического режима – параметры участков и параметры линий.

По типу данных параметры делятся на целые числа, вещественные числа, текст и дата.

С выделением двух классов параметров возникает типичная проблема объектно-ориентированной разработки на языке, не поддерживающем множественное наследование. При проектировании класса параметра можно использовать наследование для отражения одной классификации и композицию для отражения другой. Решить проблему помогут рекомендации Бертрана Мейера [2].

1. Правило изменений: не используйте наследование для описания отношения, воспринимаемого как «является», если компоненты соответствующего объекта могут изменять тип в процессе выполнения.

2. Правило полиморфизма: наследование подходит для описания отношения, воспринимаемого как «является», если для сущностей может возникнуть потребность присоединения к объектам различных типов.

Удобно иметь возможность изменять тип данных параметра в ходе проектирования схемы производственного процесса. Кроме того, параметр может изменить свой тип данных в процессе эволюции системы. С другой стороны, нецелесообразно изменять объект, которому принадлежит параметр. При последующей обработке удобно использовать параметры различных объектов процесса как полиморфную сущность. Другим модулям приложения нет необходимости отличать участок и материал, им удобнее рассматривать эти сущности как контейнер параметров.

Таким образом, определено, что класс параметра должен иметь наследование, отражающее классификацию параметров с точки зрения их принадлежности различным элементам схемы производственного процесса. Кроме этого, существует иерархия классов, описывающих тип данных параметра (рис. 4).

Реализованы два способа хранения схемы технологического процесса. Схему можно сериализовать в формат xml. Это позволяет сохранять ее, не привязываясь к конкретной БД параметров.

При формировании БД схему можно хранить в таблицах самой базы, тем самым создавая дополнительные поля, имеющие отношение не к конкретной схеме технологического процесса, а к ее реализации на конкретном производстве.

После реализации схемы производственного процесса и описания параметров, значения которых надо обрабатывать и анализировать, необходимо разработать механизм для регистрации и хранения значений параметров производственного процесса.

При реализации второго модуля приложения центральным понятием является журнал – коллекция записей о состоянии параметров одного элемента производственного процесса. Каждая запись в нем привязана к определенному моменту времени. В зависимости от предложенного способа декомпозиции производственного процесса журналы могут быть двух видов – журналы линий и журналы потоков. Принципиально они ничем не

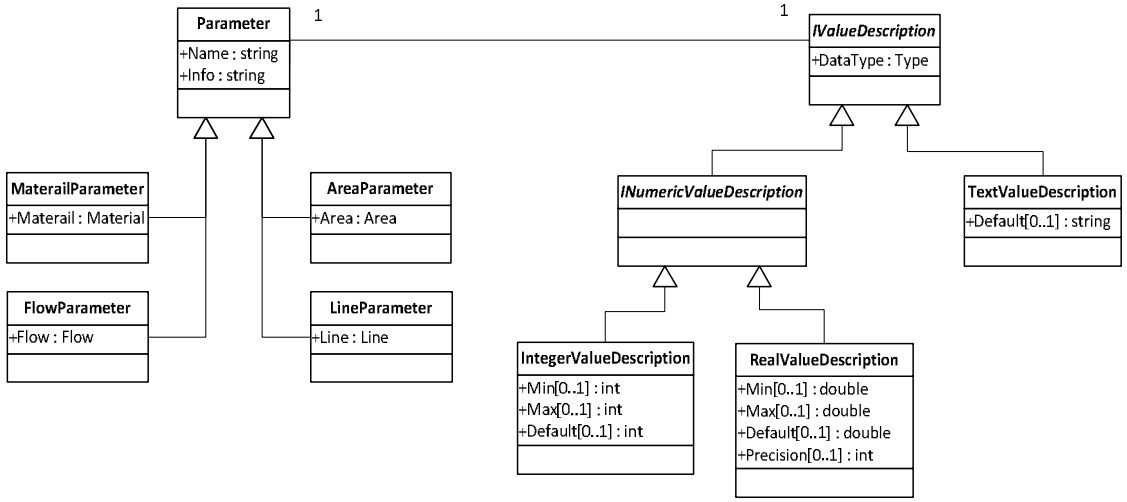


Рис. 4. Классы, описывающие параметр

отличаются друг от друга и поэтому описываются одним классом (Journal). Для удобства созданы классы, наследующие Journal и содержащие ссылки на элементы схемы производственного процесса, для которых реализован журнал.

Журналы линий одного участка или потоков одного материала частично имеют общую структуру, которая характеризуется параметрами участка или материала. Различия между журналами определяются параметрами, принадлежащими отдельным участкам и потокам.

Может возникнуть необходимость работать с параметром как с атрибутом участка или материала в целом, а не с конкретной линией или потоком. Например, если возникнет желание проследить, как изменялось значение параметра на всех линиях участка сразу, следует ввести понятие группы журналов (JournalGroup) (рис. 5).

В качестве объектов, непосредственно хранящих записи о значениях параметров, решено использовать объекты DataTable, являющиеся частью технологии ADO.Net. Авторы реализовали класс JournalTable, наследующий непосредственно сам DataTable, и классы, наследующие некоторые из его компонентов (DataColumn и DataRow). При использовании БД для каждого журнала создается отдельная таблица базы, структура которой соответствует структуре журнала.

Для выборки данных по определенным критериям использована технология LINQ, предоставляющая удобный способ обработки коллекций данных.

На основе созданной выборки приложение строит графики и гистограммы, рассчитывает статистические оценки.

Приложение для мониторинга предоставляет следующие возможности анализа данных: построение графиков изменения параметра во времени и графиков зависимости между параметрами с определением корреляции и регрессии.

Проанализируем данные на примере опытного производства ПАН нити в 80-е годы прошлого

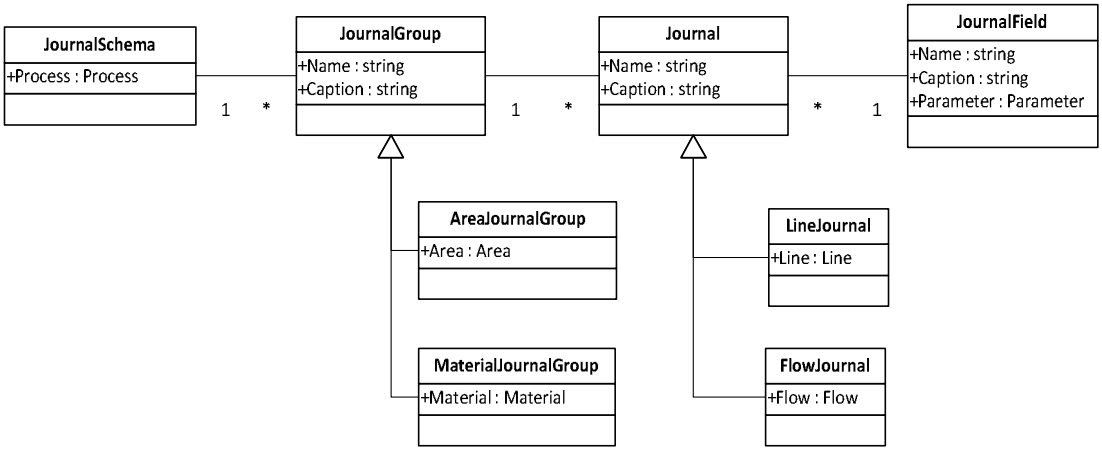


Рис. 5. Классы, описывающие схему журналов производственного процесса

столетия с некоторыми изменениями. В БД созданы два участка (формование и термообработка) с восемью и тремя линиями соответственно.

Параметры участка формования включают: разрывную нагрузку нити – «нагрузка», разрывное удлинение нити – «удлинение», «крутка», «концентрация» прядильного раствора, «вязкость рабочая» прядильного раствора, число сменяемых фильер в сутки – «фильеры», «номер синтеза» полимера, «номер смески» полимера, идущего на растворение, «вязкость удельная» растворяемой смески полимера, «зольность» растворяемой смески полимера, параметры диметилформамида, из которого готовится прядильный раствор «ДМФ», «пыль» – количество пылевидной фракции полимера в данной партии полимера, «текс» – линейная плотность измеряемой нити, «выработка» нити (в кг) на данной линии (машине). Количество параметров в программе можно изменять. В ходе анализа технологического процесса можно установить незначительно влияющие параметры, исключить их из текущей БД и завести новые в поиске более значимых параметров.

На участке термообработки использованы параметры «нагрузка», «удлинение», «крутка» и «текс» термообработанной нити.

Опытная БД составляет более 9 000 записей. Статистическая обработка таких больших выборок позволяет делать выводы с большой достоверностью согласно закону больших чисел. По мере работы производства с применением такой программы происходят все большая отработка и оптимизация технологического процесса.

Приведем пример использования программы для оперативного анализа работы производства. Приложение для мониторинга позволило сравнить работу различных машин. На рисунке 6 показана зависимость удлинения от разрывной нагрузки нити, полученной на четырех различных формовочных машинах.

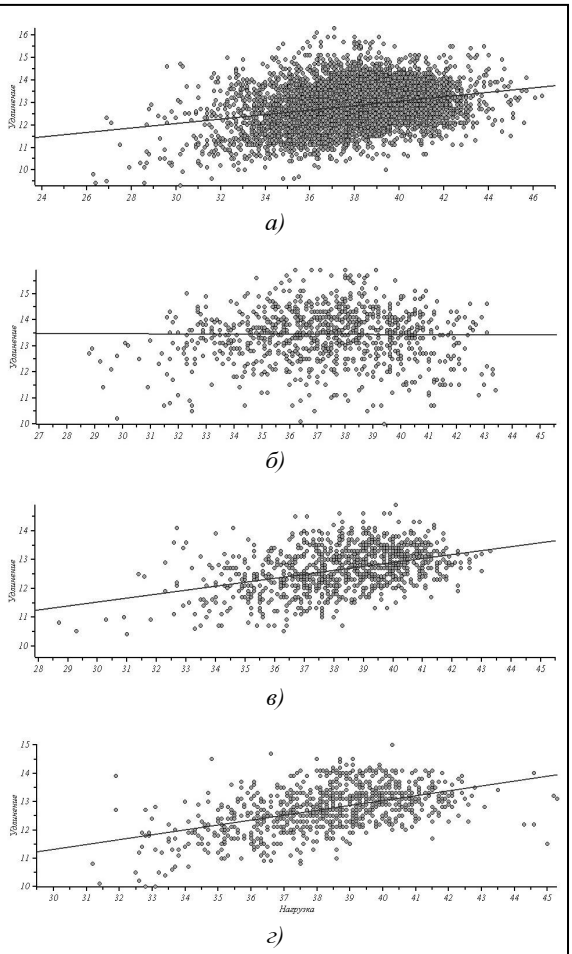
На основе приведенных графиков технологом были сделаны следующие выводы.

Существует зависимость между этими величинами, которая близка к линейной: $y=ax+b$. Это свидетельствует о дефектности структуры нити. Чем больше коэффициент b, тем выше разнородность структуры нити.

Данная зависимость разная на различных машинах. Это говорит о том, что разные машины при вроде бы одинаковом технологическом режиме производят волокно, различное по своей структуре.

Различается и среднеквадратическое отклонение на разных машинах. Оно варьируется от 1,08 до 0,75, то есть в пределах 30 %.

Различны средние значения величин нагрузки и удлинения на разных машинах: они варьируются для нагрузки на 1,2 и для удлинения на 0,6, что составляет примерно 3 % и 5 % соответственно.



а) *б)* *в)* *г)*

Рис. 6. Зависимость между разрывной нагрузкой и разрывным удлинением свежесформованной нити: а) на всех машинах; б) на машине 1; в) на машине 6; г) на машине 8

Отметим, что данное отличие величин наиболее достоверно при большом объеме выборки и определяется различием машин.

Следующим шагом в развитии системы должно стать создание экспертной системы для построения базы правил корректировки параметров технологического процесса на основе интеллектуального анализа данных этого процесса. База правил позволила бы хранить и накапливать опыт и знания эксперта по ведению технологического процесса. Кроме того, ее использование могло бы помочь эксперту-технологу в настройке параметров процесса, оптимизировать эту процедуру с учетом внешних воздействий, например, изменений, связанных с выбором станков или сырья.

Литература

1. Шах А.Д. Организация, планирование и управление предприятием химической промышленности. М.: Высш. школа, 1981.

2. Мейер Б. Объектно-ориентированное конструирование программных систем. М.: Русская Редакция, 2005.

3. Приемы объектно-ориентированного проектирования. Паттерны проектирования / Э. Гамма [и др.]. СПб: Питер, 2007.

УДК 004.4'2+588

МОДЕЛЬ ФОРМИРОВАНИЯ АДАПТИВНОЙ СРЕДЫ ОБУЧЕНИЯ И ОЦЕНКА ЕЕ ЭФФЕКТИВНОСТИ

С.Г. Зайдуллина; Н.Г. Мигранов, д.ф.-м.н.
(Башкирский государственный педагогический университет им. М. Акмуллы, г. Уфа, sv_sa@mail.ru, ufangm@yahoo.co.uk)

Рассматривается модель адаптации представления материала в системе формирования электронной информационной поддержки процесса обучения. Предложена система по созданию интерактивных курсов на базе интернет-технологий с учетом психофизиологических особенностей обучаемых.

Ключевые слова: *инструментальные средства разработки ПО, модель, адаптивные электронные обучающие системы.*

Непрерывный рост общего объема информации, потребность в ее самостоятельном усвоении, с одной стороны, и развитие современных информационных технологий, e-learning, дистанционного обучения, с другой, привели к необходимости и возможности разработки эффективных адаптивных электронных обучающих систем. Такие системы, базирующиеся на моделях обучаемых, позволяют разрабатывать индивидуальные учебные траектории и интерфейсы компьютерных учебных курсов.

Использование адаптивных компьютерных обучающих систем, функционирующих как в локальной, так и в Глобальной сети, создают для обучаемого максимально комфортные условия. Для повышения результативности управления процессом получения знаний, умений, навыков студентами необходимо внедрять в учебный процесс инструментальные среды, способные извлекать из общей базы знаний среды обучения персонифицированную информацию и проектировать индивидуальные учебные траектории.

Предлагаемая система инструментальных средств для разработки динамических многовариативных электронных курсов позволит, на взгляд авторов, спроектировать и реализовать вариативный подход при формировании электронной информационной поддержки курса обучения. Комплекс по проектированию и генерации адаптивных компьютерных средств обучения базируется на когнитивных стилях обучения.

Система инструментальных средств «УНИКУМ» для разработки динамических многовариативных электронных курсов (Свид. о гос. регистрации прогр. для ЭВМ № 2009616960 зарегистр. 15.12.2009 г., авторы: Зайдуллина С.Г., Пинемасов Е.К.) является фундаментом для формирования электронной информационной поддержки процесса обучения. Она позволяет создавать электронный адаптивный курс обучения для проведения дистанционных занятий или занятий в локальной сети, формирует документы для ведения отчетности и дает возможность преподавателю, автору курса следить за динамикой усвоения знаний.

Проектирование адаптивного авторского курса

Для управления контентом в адаптивной компьютерной обучающей системе выделим подмножества, группирующие знания о предметной области обучения и обучаемых: B – множество, база знаний дисциплины (курса), стандарт или единая планка, дидактические единицы по дисциплине; $\overline{K}=(k_1, k_2, \ldots, k_l)$ – множество компонентов обучения и их соответствие дидактическим единицам; $\overline{O}=(o_1, o_2, \ldots, o_n)$ – множество категорий обучаемых.

Дополним систему множеством интерфейсов $\overline{I}=(i_1, i_2, \ldots, i_m)$. F – алгоритм построения индивидуальной учебной траектории и интерфейса можем записать как $F(B, \overline{O}, \overline{I}, \overline{K}) \to \overline{UT}$, где $\overline{UT}=(ut_1, ut_2, \ldots, ut_n)$ – индивидуальные учебные траектории.

Индивидуализация достигается за счет взаимосвязанности отдельных компонентов электронной системы обучения.

Для решения задачи адаптивного структурно-параметрического управления содержанием электронного учебного курса можно выделить следующие этапы:

– идентификация обучаемых на принадлежность к множеству O_n;

– выбор алгоритмов адаптации и управления для каждого из множеств;

– определение структуры отдельных модулей знаний и их настройка;

– оптимизация параметров системы;

– формирование алгоритмов переключения и алгоритмов управления.

Процесс разработки электронного адаптивного авторского курса средствами инструментальной системы представлен на рисунках 1, 2. Будущее электронного проекта во многом определяется разработчиком курса, его знаниями предметной области, эргономических требований, дизайнерскими способностями.

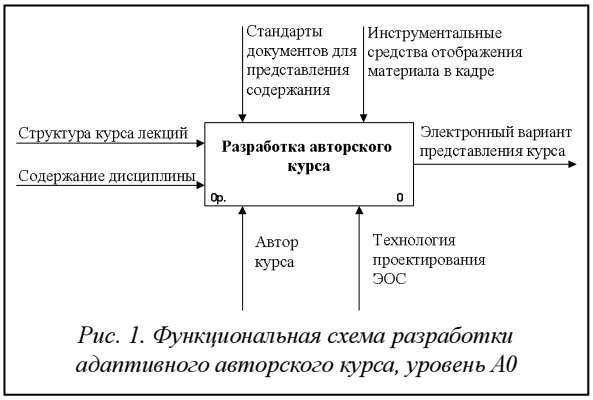


Рис. 1. Функциональная схема разработки адаптивного авторского курса, уровень А0

Комплекс программ формирует две древовидные структуры, описывающие содержание и внешний вид электронного средства обучения. Как правило, адаптация при проектировании таких систем идет по двум направлениям: адаптация навигации и адаптация контента, формы представления материала [1]. Адаптивная навигация – это выбор короткого пути к наиболее актуальной для обучаемого информации из общего объема базы. Адаптация контента, формы представления материала на отдельной странице в системе электронного обучения формируется исходя из индивидуальной модели пользователя (его психологических, личностных особенностей, когнитивного стиля).

При встрече с новым материалом учащийся соотносит идеи, находящиеся в электронных средствах обучения, с теми знаниями, которые у него уже имеются. Успешность такого соотношения обусловливает эффективность учения и определяется тем, насколько психологически обоснованно, логично и согласованно представлена учебная мультимедиа-информация. При создании электронных средств обучения следует учитывать психологические особенности обучаемых. Известно, что лица с образным типом памяти и художественным, гуманитарным складом ума предпочитают активные формы обучения с преобладанием наглядно-образных форм подачи мультимедиа-материала в интересной игровой форме. Лицам с аналитическим, математическим складом ума больше подходят самостоятельная работа с материалом, отработка различных умений с помощью электронных средств обучения, аналитические виды заданий. Данные о пользователе (выбор модели обучаемого) влияют на цветовую гамму кадра, наличие музыки, звукового сопровождения, формирование содержания кадра (приоритет появления компонентов). Важно то, как человек получает информацию, как она преподносится и воспринимается им. С учетом данных о силе подвижности нервных процессов настраиваются свойства текстового компонента, частота смены видов деятельности. Создавая электронные средства обучения, необходимо ориентироваться на некоторую оптимальную скорость подачи разнотипной информации, которая не превышала бы способности человека по ее восприятию, но в то же время была бы достаточной для поддержания активности на высоком уровне. Предлагаемая система настраивает и позволяет изменять цветовое решение и ассоциативный аудиоряд отдельных фрагментов обучающего курса.

В системе формирования электронной информационной поддержки процесса обучения предусматривается возможность самому обучаемому выбрать скорость, объем подачи материала, стратегию обучения в соответствии со своими психологическими особенностями, что также помогает в индивидуальном обучении.

При классификации человека возникает нечеткое множество $\overline{O}$, вводятся переменная represent-canal (ведущий канал восприятия) значения – термы «аудиал», «кинестик», «визуал» и переменная temperment значения – термы «сангвиник», «холерик», «меланхолик», «флегматик». Исходя из результатов тестирования на определение типа темперамента (тест по методике изучения свойств нервной системы, тест для изучения структуры темперамента Я. Стреляу [2]), в модуле идентификации пользователя формируется степень принадлежности обучаемого к термам «H», «F» «S» «M». Аналогично (биас-тест определения репрезента-

Рис. 2. Декомпозиция уровня А0

тивной системы [3]) устанавливается степень принадлежности обучаемого к термам «A», «V», «K». На этапе выбора алгоритмов адаптации и управления для каждого из множеств система обращается к правилам продукции:

If (temperment is H) and (representcanal is A) then (SlideScheme is SH1) (Color is Qt)(Sound is SS)(Sheets is I)
If (temperment is S) and (representcanal is A) then (SlideScheme is SH1) (Color is S)(Sound is SB)(Sheets is P)
If (temperment is F) and (representcanal is A) then (SlideScheme is SH1) (Color is Br)(Sound is SS)(Sheets is P)
If (temperment is M) and (representcanal is A) then (SlideScheme is SH1) (Color is Br)(Sound is SB)(Sheets is I)
If (temperment is H) and (representcanal is V) then (SlideScheme is SH2) (Color is Qt)(Sound is SS)(Sheets is I)
If (temperment is S) and (representcanal is V) then (SlideScheme is SH2) (Color is S)(Sound is SB)(Sheets is P)
If (temperment is F) and (representcanal is V) then (SlideScheme is SH2) (Color is Br)(Sound is SS)(Sheets is P)
If (temperment is M) and (representcanal is V) then (SlideScheme is SH2) (Color is Br)(Sound is SB)(Sheets is I)
If (temperment is H) and (representcanal is K) then (SlideScheme is SH3) (Color is Qt)(Sound is SS)(Sheets is I)
If (temperment is S) and (representcanal is K) then (SlideScheme is SH3) (Color is S)(Sound is SB)(Sheets is P)
If (temperment is F) and (representcanal is K) then (SlideScheme is SH3) (Color is Br)(Sound is SS)(Sheets is P)
If (temperment is M) and (representcanal is K) then (SlideScheme is SH3) (Color is Br)(Sound is SB)(Sheets is I)

Значения параметров SlideScheme и Color приведены в таблицах 1, 2. Значения для Sound: SB – есть негромкая фоновая музыка; SS – фоновая музыка отсутствует или очень тихая. Объем последовательного изложения материала Sheets принимает значения I – ограничение объема системой (4–5 слайдов) или P – произвольное количество кадров, запланированных преподавателем.

Таблица 1

Обозначение значений переменной SlideScheme

SlideScheme	Порядок выбора формы представления материала на отдельной странице (слайде)
SH1	Текст, рисунок/схема, видео/анимация, обязательно звуковой ряд (голосовое сопровождение слайда)
SH2	Рисунок/схема, видео/анимация, текст
SH3	Видео/анимация, рисунок/схема, текст, выбор пользователем вкл./выкл. звукового ряда

Таблица 2

Обозначение значений переменной Color

Color	Преобладание цветовой гаммы для текстовых компонентов и подложки кадра
Br	Яркая гамма (наличие красного и желтого цветов)
S	Стандартная (черный, белый, синий цвета, минимальное присутствие оттенков желтого цвета)
Qt	Спокойная цветовая гамма (наличие синего, зеленого цветов)

Для адаптации контента, формы представления материала (текст, анимация, рисунок, фоновый звук, голосовое сопровождение) на отдельной странице (слайде) созданы классы объектов [4]. В настоящий момент в системе не предусмотрено создание отдельных элементов учебного материала (рисунков, коллажей, анимации и т.д.). Представляется целесообразным подключение, загрузка в систему уже готового контента (например, информации, полученной из Сети) или контента, созданного в привычно используемых разработчиком курса средах. Таким образом, автор курса может воспользоваться возможностями, предлагаемыми в Интернете, или собственными наработками. По умолчанию каждому добавляемому элементу (будь то текст, рисунок, анимация или звуковой фрагмент) присваивается нулевой базовый уровень, однако на слайде должен присутствовать хотя бы один элемент, значение которого в поле base_Element равно единице. Это элемент, являющийся, на взгляд разработчика курса, основополагающим в объяснении. Для реализации взаимозаменяемости у каждого компонента существует свойство «связка» (bound), в котором указывается имя аналога компонента.

Для определения отдельных модулей знаний и их настройки была разработана структура хранения материала, изображенная на рисунке 3.

Результаты самооценивания пользователем уровня усвоения отдельных кадров, рубежное тестирование влияют на корректировку сценария компьютерного учебного курса.

Устанавливается связь между отдельными компонентами среды обработки и формированием многовариативных курсов.

Представление отдельных кадров, их интерфейс формируются исходя из набора объектов авторского курса, результатов тестирования обучаемых и индивидуальной накапливаемой базы самооценок.

Формирование сводных ведомостей итоговых и промежуточных результатов обучения позволяет проследить динамику усвоения знаний и создает базу для статистики.

Модель адаптации представления материала реализована в системе «УНИКУМ» для формирования электронной информационной поддержки процесса обучения. Система имеет интуитивно понятный интерфейс с продуманной навигацией. В конечном итоге эффективность процесса обучения зависит не только от содержания электронных учебных курсов и деятельности преподавателя, но и от эффективности интерфейса и функциональных возможностей системы электронного обучения. Формирование кадров происходит при помощи панели компонентов, включающей в себя кнопки для добавления изображений, анимации, текста, фонового звукового ряда. В каждом кадре конструктора слайдов присутствует кнопка «Привязка кадра к терминологическому словарю, определение атомарных знаний». Добавление тестовых

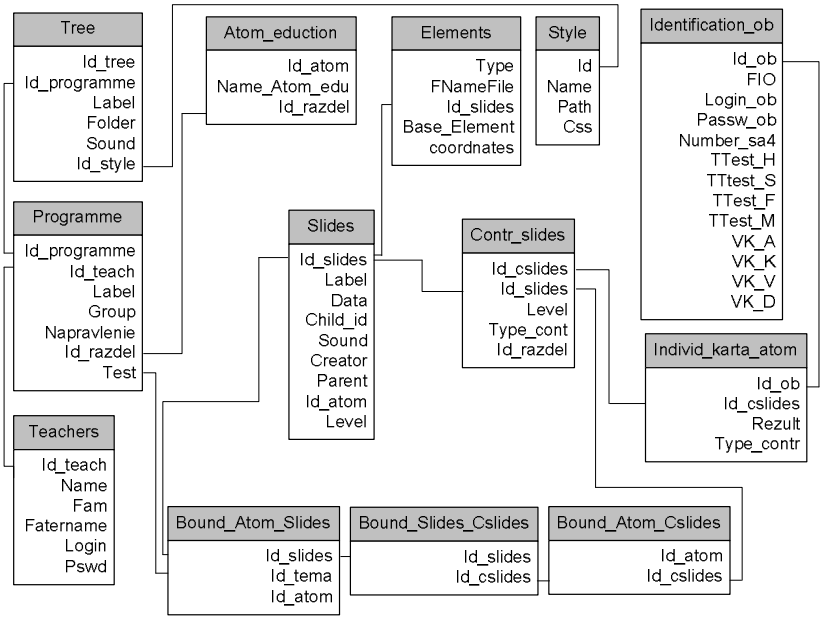


Рис. 3. Структура таблиц формирования курса, хранения индивидуальных результатов и параметров

заданий, задание уровня сложности преподносимой в кадре информации осуществляются с помощью двух соответствующих пиктограмм панели инструментов кадра. Для формирования адаптивного электронного курса обучения с помощью кнопки «Добавить N-е представление слайда» на панели инструментов к каждому кадру желательно добавлять до двух-трех альтернативных представлений информации.

Комплекс программ формирования адаптивных курсов был реализован в Adobe Flex Builder 3 с использованием языка программирования PHP5, СУБД MySQL. Adobe Flex 3 предоставляет полные мультимедийные возможности Flash Platform, включая потоковое видео и звук. Для работы программы необходимо наличие веб-сервера Apache, сервера управления БД My SQL и интернет-браузера.

Анализ результатов обучения студентов с применением комплекса

На базе Башкирского государственного педагогического университета им. М. Акмуллы и Башкирского государственного университета был в два этапа проведен эксперимент с участием 103 студентов.

На первом этапе изучался ряд разделов дисциплины в двух группах. Отобранные студенты не были знакомы с изучаемым материалом. Первая группа студентов проходила обучение с помощью электронного курса, сформированного комплексом программ генерации обучающих компонентов без учета психофизиологических различий. Вторая группа изучала материал, подготовленный с применением адаптационных механизмов электронной системы обучения.

На втором этапе применялся еще один параметр дифференциации – разный начальный уровень знаний предметной области. При обучении использовалась электронная информационная поддержка курса, сформированная на базе комплекса программ генерации обучающих компонентов с применением адаптации формы, контента и навигации.

Проводилось входное и итоговое тестирование. Результаты эксперимента записывались в БД с указанием Ф.И.О., времени на освоение темы, результатов тестирования.

Эффективность электронной адаптивной системы с закладываемым однотипным по объему содержанием учебного контента оценивалась по таким показателям, как

- уровень освоения обучаемым учебного материала;
- время, необходимое для освоения отдельного учебного модуля;
- количество остаточных знаний по прошествии времени (полгода).

Оценка эффективности электронного обучения при различных способах представления учебного материала приведена в таблице 3.

Таблица 3

Результаты по контрольному учебному модулю

Подход	Освоение материала		Количество остаточных знаний, %
	Уровень, %	Время, час.	
I	66	11	35
II	70	9	44
III	78	8	57

Таким образом, выявлено сокращение времени усвоения при использовании электронного адаптивного материала в среднем на 20 %. Анализ результатов обучения студентов с применением комплекса показал рост интереса к обучению, повышение его качества.

Литература

1. Исследование и анализ методов и моделей интеллектуальных систем непрерывного обучения / О.О. Гагарин [и др.]. URL: http://www.setlab.net/?view=Tytenko_KPIj (дата обращения: 02.11.2010).

2. Методика изучения свойств нервной системы учащихся. URL: http://vsetesti.ru/396/ (дата обращения: 28.04.2011).

3. Биас-тест определения репрезентативных систем. URL: http://dic.academic.ru/dic.nsf/enc_psychotherapeutic/39/БИАС (дата обращения: 28.04.2011).

4. Зайдуллина С.Г., Мигранов Н.Г. Комплекс программ генерации обучающих компонентов для обеспечения вариативного подхода в формировании электронных информационных ресурсов // Вестн. Поморского ун-та: сер. Естественные науки. Архангельск: Изд-во ПГУ. 2009. № 3.

УДК 004.78

МОДЕЛИРОВАНИЕ ПОКАЗАТЕЛЕЙ НАУЧНОЙ ДЕЯТЕЛЬНОСТИ ПРИ СОЗДАНИИ ИНФОРМАЦИОННО-АНАЛИТИЧЕСКОЙ СИСТЕМЫ ВУЗА

Р.В. Малецкий; В.В. Пикулин, к.т.н.
(Пензенская государственная технологическая академия, pvv@pgta.ru)

Рассматриваются вопросы моделирования показателей научной деятельности вуза, математические и структурные модели, используемые в процессе создания информационно-аналитической системы.

Ключевые слова: *научная деятельность, показатели эффективности, автоматизация, информационно-аналитическая система.*

Важнейшей составляющей работы как вуза в целом, так и отдельных его подразделений и сотрудников является *научная деятельность* (НД). По ее интенсивности и результативности, наличию в ней инновационной составляющей можно судить о соответствии вуза своему статусу. Научно-исследовательская деятельность вуза – главное средство повышения квалификации преподавательского состава и лучший способ привлечения студентов к нестандартной, творческой работе по изучаемому предмету. Исследовательская деятельность рассматривается как составная часть обязанностей всех членов коллектива вуза.

Для поддержки и развития НД вуза необходимо применять адекватные методы и современные технологии управления. К эффективным инструментальным средствам управления относятся *информационно-аналитические системы* (ИАС). ИАС поддержки научной деятельности, входящая в корпоративную информационную среду вуза, является наиболее совершенным средством управления, поскольку способна оперировать огромными массивами информации, связанными с объектом управления, интегрировать информационно-аналитическую поддержку научной деятельности с поддержкой других видов деятельности вуза – учебной, экономической и др. [1].

Вопросы структурно-технической организации, функционального состава, технологии применения АСУ научно-исследовательской работой в вузах рассматриваются во многих публикациях, например в [1–4].

Одним из центральных вопросов, решаемых при создании ИАС НД, является определение показателей, которые должны использоваться для оценки продуктивности, качества и эффективности НД вуза в целом, а также его подразделений и сотрудников. От состава показателей зависят набор данных, которые должны храниться в БД системы, технология и алгоритмы их обработки, содержание пользовательских интерфейсов и ряд других проектных решений, поэтому, по мнению авторов данной публикации, вопросы моделирования предметной области следует рассматривать во взаимосвязи с составом и содержанием показателей эффективности НД в вузе.

После определения состава показателей НД, которые будут оцениваться средствами создаваемой ИАС, требуется выполнить их формализацию и проработать вопросы технологии подготовки соответствующих первичных данных и их обработки, для чего необходимо создание ряда функциональных, информационных, математических моделей.

К показателям оценивания НД предъявляются противоречивые требования, которые следует учитывать при выборе или синтезе показателей, а также при разработке средств информационной поддержки НД:

– достаточно полная характеристика направления работ;

– минимальная трудоемкость процессов формирования значений показателей;

– возможность оценивания НД, выполняемой отдельными сотрудниками, подразделениями, группами подразделений, вузом.

Для подготовки методики создания информационного и программного обеспечения ИАС на основе анализа показателей НД следует использовать совокупность моделей, позволяющую создавать корректные компоненты всех видов обеспечения системы: аналитические и графовые модели, а также модели «сущность–связь» и функциональные модели в формате IDEF0 (вместо

последних могут использоваться объектно-ориентированные модели).

Модели могут быть разработаны индивидуально для каждого показателя или для группы логически связанных показателей. Кроме того, следует проанализировать особенности вычислительных процессов и выработать предложения по их оптимизации.

Рассмотрим примеры моделей для некоторых типичных показателей оценки НД вуза с учетом Постановления Правительства РФ «Об оценке результативности деятельности научных организаций…».

1) Общий объем НИР ($V_{общ}$) и объем НИР за счет собственных средств ($V_{внутр}$) (тыс. руб.); значения показателей вычисляются для каждого объекта оценивания – подразделения, факультета, вуза:

$$V_{общ} = \sum_{i=1}^{K} V_i , \qquad (1)$$

$$V_{внутр} = \sum_{i=1}^{K_{внутр}} V_{внутр,i} , \qquad (2)$$

где V_i – объем i-й НИР (тыс. руб.); K, $K_{внутр}$ – соответственно общее количество договоров и количество внутренних договоров на выполнение НИР по объекту исследования.

2) Удельный вес внутренних затрат на исследования и разработки в общем объеме выполненных научной организацией работ, услуг (%):

$$V_{внутр,уд} = \frac{V_{внутр}}{V_{общ}} 100\ (\%) . \qquad (3)$$

3) Внутренние затраты на исследования и разработки, отнесенные к численности исследователей (тыс. руб./чел.):

$$V_{внутр,1} = \frac{V_{внутр}}{N_{иссл}} . \qquad (4)$$

4) Среднегодовой объем НИР на одного исследователя за последние пять лет (тыс. руб./чел.):

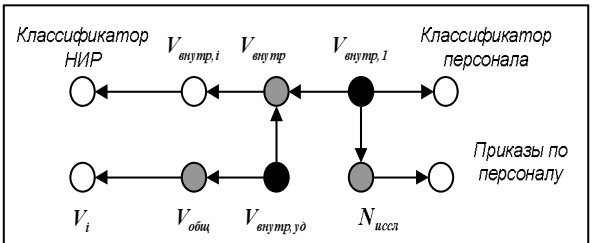
$$p_{5,1} = \frac{1}{5\, N_{иссл,5}} \sum_{i=1}^{K_5} V_i . \qquad (5)$$

5) Среднегодовой объем внутренних НИР за последние пять лет:

$$p_{внутр,5,1} = \frac{1}{5\, N_{иссл,5}} \sum_{i=1}^{K_{внутр,5}} V_{внутр,i} , \qquad (6)$$

где K_5, $K_{внутр,5}$ – соответственно общее количество и количество внутренних выполненных и профинансированных за пять лет НИР; $N_{иссл,\,5}$ – среднегодовое количество сотрудников, отнесенных к категории исследователей.

Для первых четырех показателей граф зависимости промежуточных и результатных показателей включает первичные оперативные данные и классификаторы (фрагмент графа зависимостей показателей см. на рис. 1). Граф показывает, какие данные должны хранить в БД системы для вычисления определенных показателей НД и от каких первичных и промежуточных данных зависят результатные показатели.



Примечание: ○ – первичные, ◉ – промежуточные, ● – результатные показатели

Рис. 1

На основе декомпозиции и анализа содержания сформулированных целей и задач процесса управления НД, а также определенных формализованных показателей разработаны функциональные модели, используемые для синтеза ПО ИАС, например, функциональная модель типичного процесса учета НИР в вузе без учета возможных подпроцессов по работе с рекламациями заказчиков включает четыре подпроцесса (рис. 2, 3). Для внутренних НИР вместо выписки из банковского счета должны использоваться другие или дополнительные документы, например, ведомости оплаты.

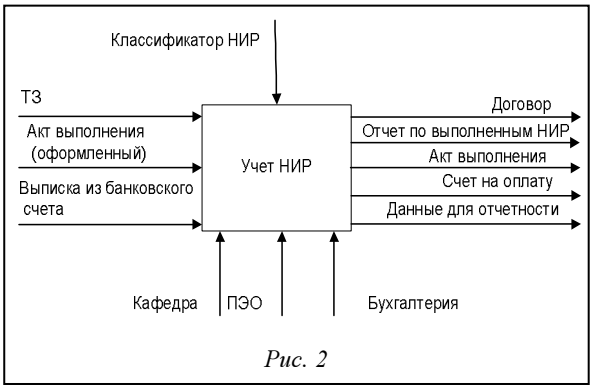


Рис. 2

На основе графовой и функциональных моделей разработана диаграмма классов (рис. 4). В нее входят следующие группы классов: справочники (Кафедра, Тип_участника_НИР, Источник_финансирования, НТ_направление, Объект_финансирования, Статус_НИР, ЛКСЭЦ, Характер_НИР, Область_науки, Классификатор_ГРНТИ, Код_ГРНТИ, Вид_приказа), персональные данные участников (Студент, Сотрудник, Приказ, Содержание_приказа) и данные по НИР (НИР, Участник_НИР, Этап_НИР, Оплата_НИР, Статистика).

Вычисление показателей, характеризующих НД за пять лет и включающих данные о количестве исследователей, может выполняться по различным правилам, содержание которых влияет на производительность и надежность процессов обработки данных.

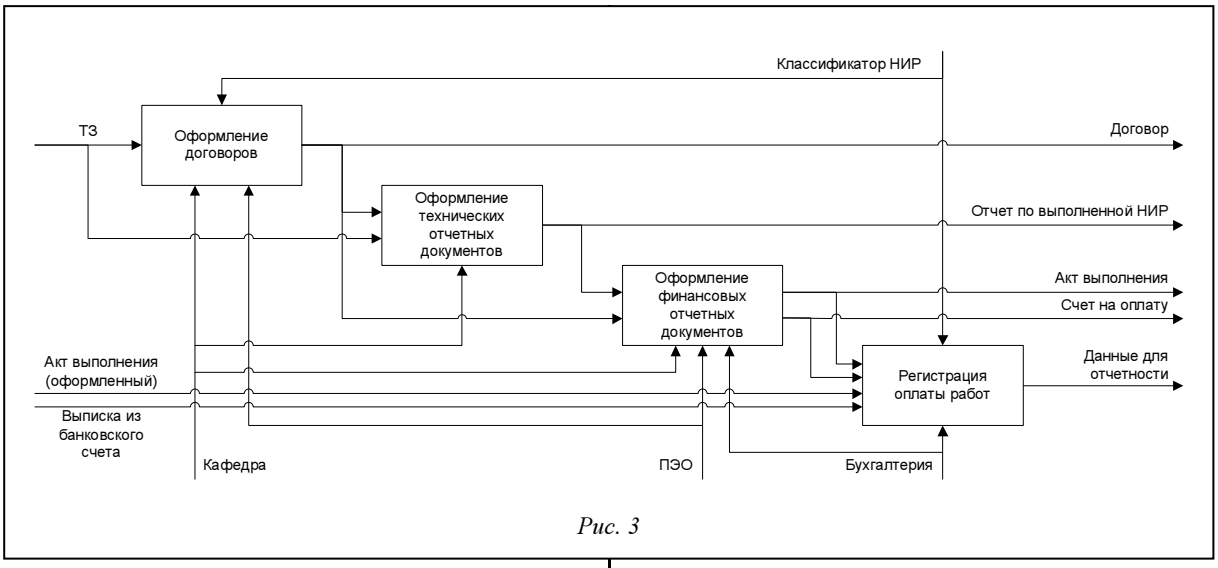


Рис. 3

Непосредственное использование (1) и (2) требует наличия в БД ИАС НД первичных данных за пятилетний период, количество которых велико, и достаточно сложных и многократно повторяемых процедур обработки данных, что требует существенных затрат времени. Кроме того, за такой период накопления данных возможна их частичная потеря (это могут быть и документы НИР, и приказы на сотрудников); продолжительная обработка накопленных данных сопряжена с вероятностью сбоев системы. Для снижения риска проявления негативных факторов можно использовать обработку данных, накапливаемых в течение определенного периода (одного года или полугодия). Обработка данных, накопленных за полугодие, для вуза предпочтительнее, так как требуется формирование отчетных данных за учебный и за календарный год.

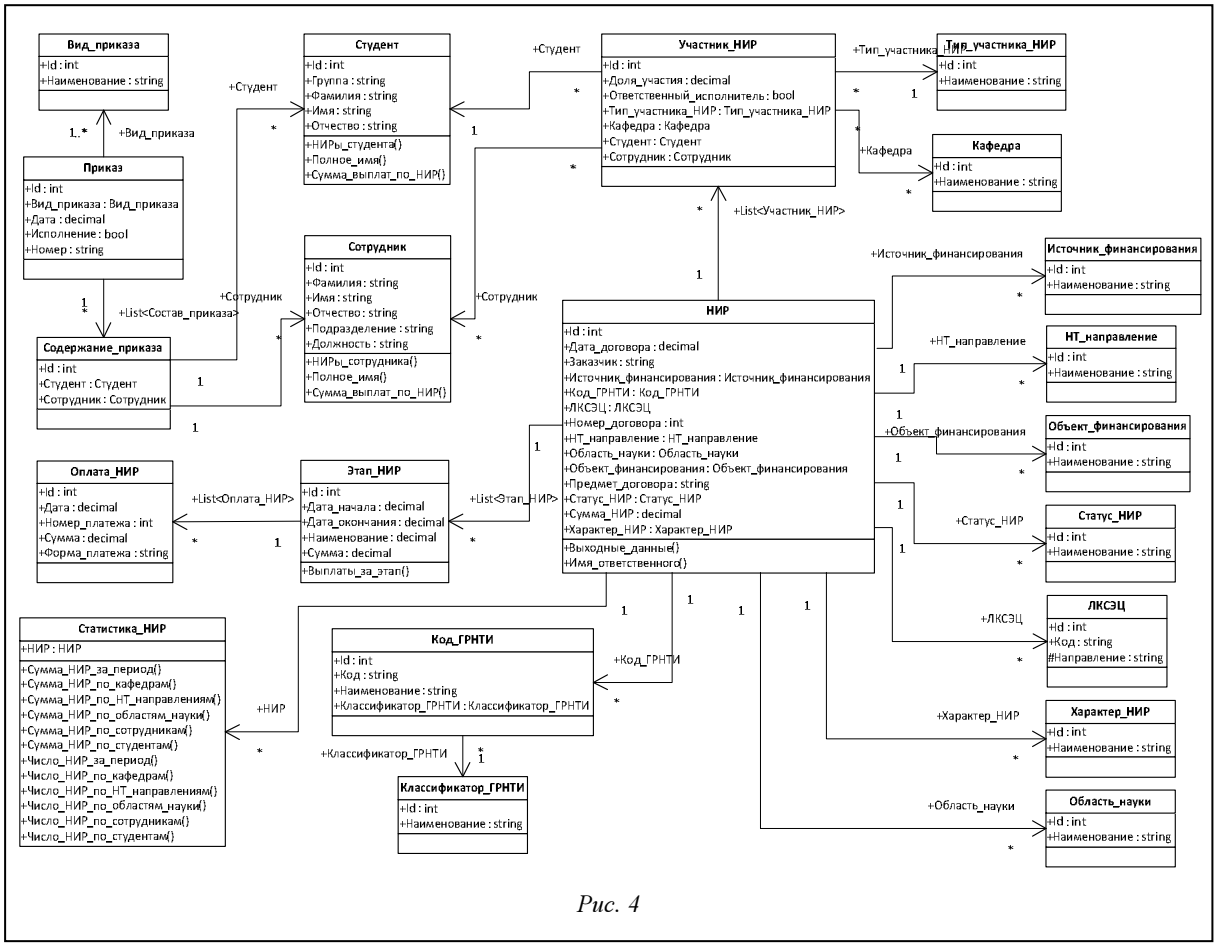


Рис. 4

Поэтому выражения (5) и (6), использующиеся для вычисления $P_{5,1}$ накапливаемых за полугодие данных, следует преобразовать в (3) и (4):

$$p_{5,1} = \frac{0,1}{N_{иссл,5}} \sum_{i=1}^{10} V_{НИР,i}, \quad (7)$$

где $V_{НИР,i}$ – объем НИР, выполненных за i-е полугодие в рамках рассматриваемого пятилетия,

$$p_{внутр,5,1} = \frac{0,1}{N_{иссл,5}} \sum_{i=1}^{10} V_{внутр\ НИР,i}. \quad (8)$$

Величину $N_{иссл,\ 5}$ наиболее просто можно оценить на основе среднесписочных значений за i-е полугодие ($\overline{N}_{иссл,i}$):

$$N_{иссл,\ 5} = 0,1 \sum_{i=1}^{10} \overline{N}_{иссл,i}, \quad (9)$$

$$\overline{N}_{иссл,i} = 0,5(N_{иссл,i,H} + N_{иссл,i,K}), \quad (10)$$

где $N_{иссл,i,H}$, $N_{иссл,i,K}$ – количество сотрудников, отнесенных к категории исследователей, на начало и окончание i-го полугодия.

Для повышения оперативности получения данных о результатах НИР в качестве периодов накопления и промежуточной обработки можно использовать квартал или месяц. В этом случае следует внести соответствующие изменения в (7) и (8) для вычисления значений показателей за пятилетку.

Использование рассмотренной технологии моделирования показателей НД позволит обеспечить создание корректных компонентов информационного, программного, методического обеспечения ИАС вуза.

Литература

1. Новиков Д.А., Суханов А.Л. Модели и механизмы управления научными проектами в вузах М.: Ин-т управления образованием РАО, 2005. 80 с.

2. Говорков А.С. Автоматизация организационно-управленческих аспектов научной деятельности вуза // Университетское управление. 2009. № 6. С. 13–18.

3. Котляров И.Д. Управление продуктивностью научной работы профессорско-преподавательского состава // Университетское управление. 2009. № 5. С. 41–48.

4. Мелехин В.А., Хеннер Е.К. Структурно-информационная модель научной деятельности классического университета // Университетское управление. 2008. № 6. С. 85–95.

УДК 681.518.2

ОБРАБОТКА ПОКАЗАТЕЛЕЙ В КОМПЬЮТЕРНЫХ МЕТОДИКАХ ОЦЕНКИ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА

А.И. Долгов, д.т.н.; А.Ф. Мартыненко; В.В. Преснухин, к.т.н.
(Ростовский военный институт Ракетных войск, dolgov-ai@yandex.ru)

Рассматриваются методы обработки в компьютерных коэффициентных методиках исходных (входных) и промежуточных показателей для получения выходных показателей рейтинговой оценки образовательного процесса (оцениваемого объекта) с учетом нормативных логических условий оценивания.

Ключевые слова: *компьютерная коэффициентная методика, образовательный процесс, изменяемый весовой коэффициент, сумматор, умножитель.*

В практике оценивания образовательного процесса наиболее востребованы программно реализованные коэффициентные методики благодаря их простоте и приемлемой точности получаемых результатов.

Под коэффициентной методикой понимается методика вычисления значения интегрального выходного показателя (или некоторого множества выходных показателей) оцениваемого объекта (процесса) путем суммирования значений вполне определенного множества входных показателей, учитываемых при этом с соответствующими весовыми коэффициентами [1].

Разработка компьютерной коэффициентной методики оценки образовательного процесса (далее – коэффициентной методики), предшествующая ее программной реализации, включает два этапа – математическое описание методов обработки показателей и синтез структурной схемы коэффициентной методики.

Математическое описание методов, необходимых для получения рейтинговой оценки образовательного процесса с учетом нормативных логических условий оценивания, включает:

– выделение предварительного значения рейтинговой оценки в виде среднего балла, соответствующего комбинациям частных оценок, определяемой логическими условиями оценивания для данного рейтингового интервала;

– определение наличия оценок, снижающих значение предварительной рейтинговой оценки;

– получение итогового (окончательного) значения рейтинговой оценки путем коррекции ее предварительного значения.

Приведем определения основных используемых понятий. *Оценка* – процесс определения (оценивания) значения показателя, характеризующего результат образовательного процесса. *Показатель* – качественная и (или) количественная характеристика, вводимая для оценивания отдельного свойства либо совокупности свойств объекта (процесса). В отличие от оценки показатель обычно имеет наименование, обозначение и значение, а при количественной оценке во многих случаях и размерность. *Шкала* – инструмент для разделения оцениваемых значений рассматриваемых однородных показателей на различимые между собой группы, где отношения между значениями показателей выражены свойствами числового ряда либо ряда наименований [2].

В образовательном процессе в зависимости от вида используемой шкалы различают балльные, качественные (шкала наименований), а иногда и балльно-качественные оценки.

Значениями балльной оценки являются целые числа – количество баллов (например, 5, 4, 3, 2).

Значения качественной оценки выражаются наименованиями («отлично», «хорошо», «удовлетворительно», «неудовлетворительно»).

В последнее время в оценивании образовательного процесса все чаще применяются рейтинговые оценки – числовые оценки, задаваемые с некоторой точностью в (рейтинговых) интервалах численных значений рейтинговой шкалы.

Наиболее часто используемая 5-балльная система оценки, а также иногда применяемые системы оценок с несколько большим количеством баллов (например 12-балльная) имеют существенный недостаток – используют ограниченное количество значений (градаций) оценивания, внутри которых объекты оказываются неразличимыми.

В связи с этим рассмотрим методы обработки показателей образовательного процесса с использованием как 5-балльных оценок, так и рейтинговых оценок с градацией чисел в рейтинговых интервалах:

4,5≤«отлично»≤5; (1)

3,5≤«хорошо»<4,5; (2)

2,5≤«удовлетворительно»<3,5; (3)

2,0≤«неудовлетворительно»<2,5.

В традиционно используемых коэффициентных методиках значения весовых коэффициентов постоянны по величине. Вместе с тем при вычислении значения того или иного показателя могут учитываться вполне определенные логические условия оценивания, учитываемые только с помощью изменяемых весовых коэффициентов [3].

Выделение предварительного значения рейтинговой оценки

Рассмотрим данный метод на примере оценивания подготовленности специалиста в рейтинговом интервале «хорошо» при нормативном требовании, задающем логические условия получения итогового значения *Rxop* рейтинговой оценки «хорошо» лишь в случае следующих комбинаций частных оценок: оценка за практические навыки должна быть не ниже «хорошо», а оценка за теоретические знания не ниже «удовлетворительно».

Предварительное значение *R'xop* рейтинговой оценки *Rxop* выделяется в виде среднего балла частных оценок практических навыков ($X_П$) и теоретических знаний ($X_Т$) лишь в случае выполнения нормативного требования, при этом

$$R'xop = \begin{cases} \dfrac{X_П + X_Т}{2}, \text{ если } X_П \ge 3,5 \text{ и } X_Т \ge 2,5, \\ 0 \text{ в ином случае.} \end{cases}$$

Конкретный пример иллюстрируется данными, представленными в таблице.

Оцениваемый объект	**Результаты оценивания**						
	Практика (П)		**Теория (Т)**		**Оценка подготовленности специалиста**		**Подготовленность специалиста с учетом *V1*, *R* (*R'xop*)**
	оценка $X_П$	Пр1,2	оценка $X_Т$	Пр1,2	интегральная оценка	X2,1	
1-й специалист	3,4 (уд)	– 100	5,0 (отл)	0	4,2 (хор)	< 0	0
2-й специалист	4,0 (хор)	0	4,4 (хор)	0	4,2 (хор)	> 0	4,2 (хор)

Первый специалист получил за практику рейтинговую оценку $X_П$=3,4 («удовлетворительно»), а за теорию $X_Т$=5 («отлично»), средняя оценка его подготовленности составила $\frac{3,4+5,0}{2}$ =4,2 балла.

Второй специалист получил оценку за практику $X_П$=4,0 («хорошо»), за теорию $X_Т$=4,4 («хорошо»), и средний балл за его подготовленность такой же: $\frac{4,0+4,4}{2}$ =4,2.

В этой ситуации в соответствии с соотношением 3,5≤«хорошо»<4,5 для среднего балла оба специалиста претендуют на оценку «хорошо».

Однако (при несоблюдении условий получения итоговой рейтинговой оценки «отлично») по логическим условиям оценивания итоговая рейтинговая оценка *R'xop* первого специалиста (ввиду наличия оценки «удовлетворительно» за практику) должна быть исключена из рейтингового интервала «хорошо», что соответствует *R'xop*=0.

Таким образом, надо определить наличие комбинации $X_П$≥3,5 и $X_Т$≥2,5, соответствующей рейтинговой оценке «хорошо», что в коэффициентной методике нереализуемо с использованием одного весового коэффициента, но осуществимо с применением трех изменяемых весовых коэффициентов.

Сформируем признак *Пр*1,2 принадлежности оценки специалиста к рейтинговому интервалу «хорошо» в соответствии с соотношением

$Пр1,2 = Хп V п2 + Хт V т2,$

где $V п2 = \begin{cases} 0, \text{ если } Хп \geq 3,5, \\ 1 \text{ в ином случае} \end{cases}$

и $V т2 = \begin{cases} 0, \text{ если } Хт \geq 2,5, \\ 1 \text{ в ином случае.} \end{cases}$

Условием наличия комбинации $Хп \geq 3,5$ и $Хт \geq 2,5$, соответствующей рейтинговой оценке «хорошо» (см. табл.), является значение показателя $Пр1,2=0$, а признаком отсутствия этой комбинации – значение показателя $Пр1,2>0$

$$R'хор = X2,1 \cdot V1,R, \quad (4)$$

где $X2,1=(Хп+Хт)\cdot 0,5+Пр1,2;$

$$V1,R = \begin{cases} 1, \text{ если } X2,1 > 0, \\ 0 \text{ в ином случае.} \end{cases}$$

Таким образом, весовой коэффициент $V1,R$ совместно с признаком принадлежности ($Пр1,2$) на основании соотношения (4) при наличии комбинации, соответствующей рейтинговой оценке «хорошо» ($Пр1,2=0$, $X2,1>0$), обеспечивает формирование значений $V1,R=1$ и как результат – $R'хор = (Хп+Хт)\cdot 0,5$, а при отсутствии комбинации ($Пр1,2=1$, $X2,1<0$) – значений $V1,R=0$ и $R'хор=0$.

Фрагмент синтезированной в соответствии с математическим соотношением (4) структурной схемы коэффициентной методики приведен на рисунке 1. Внутри прямоугольников, обозначающих сумматоры входных показателей нижестоящего уровня, указываются обозначения показателей, значения которых равны результату суммирования входных показателей, умноженному на соответствующий весовой коэффициент.

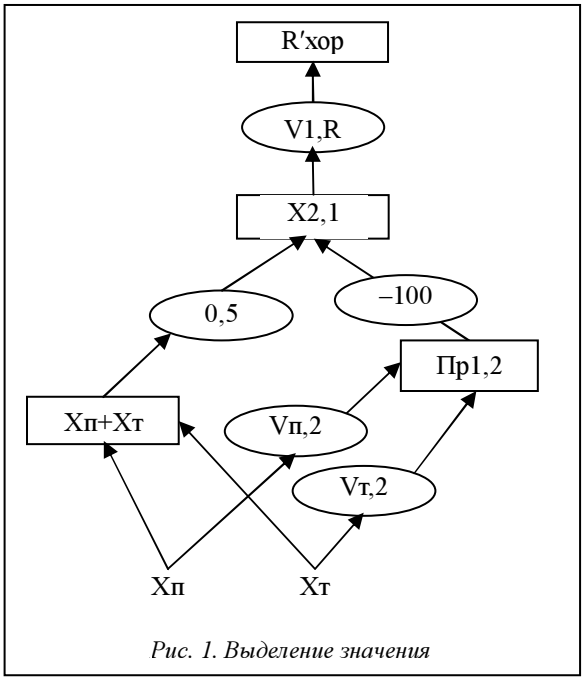


Рис. 1. Выделение значения

В овалах, обозначающих для неизменяемых весовых коэффициентов умножители, а для изменяемых весовых коэффициентов умножители с аддитивной поправкой, для неизменяемых весовых коэффициентов приводятся их значения, а для изменяемых – символические обозначения. Связи между сумматорами и умножителями указаны стрелками.

Аналогично строятся и фрагменты структурных схем для других итоговых интегральных рейтинговых оценок подготовленности специалиста.

Определение наличия снижающих оценок

При определении предварительных рейтинговых оценок могут встречаться комбинации частных оценок, приводящие к одинаковым предварительным значениям рейтинговой оценки, однако комбинации по своему составу могут быть настолько неравноценными, что в соответствие им следует поставить разные рейтинговые значения предварительной оценки.

Так, например, в интервалах оценки «хорошо» при двух результатах оценки подготовленности специалиста – в первом случае оценка за практику 4,4 («хорошо»), за теорию 4,0 («хорошо»), а во втором случае оценка за практику 5,0 («отлично»), а за теорию 3,4 («удовлетворительно») – полученные предварительные рейтинговые оценки, средний балл которых равен 4,2, оказываются одинаковыми и по нормативным условиям оценивания соответствуют рейтинговому интервалу 3,5≤«хорошо»<4,5. Однако рейтинговую оценку, соответствующую комбинации частных оценок, включающей оценку «удовлетворительно», вполне справедливо и необходимо откорректировать таким образом, чтобы сделать ее ниже рейтинговой оценки комбинации без частной оценки «удовлетворительно».

Коррекцию предварительных рейтинговых оценок можно осуществить следующим образом. Рейтинговый интервал 3,5≤«хорошо»<4,5 разделяется на два диапазона:

4,0≤«хорошо»<4,5 (верхний диапазон), (5)

3,5≤«хорошо»<4,0 (нижний диапазон), (6)

при этом полученное значение предварительной рейтинговой оценки $R'хор$ преобразуется так, чтобы в случае возникновения комбинации частных оценок за теорию и практику, не включающей оценку «удовлетворительно», выделить значение $R''хор1$, принадлежащее верхнему диапазону, а в случае комбинации, включающей такую оценку, выделить значение $R''хор2$, принадлежащее нижнему диапазону.

Выполнение подобного преобразования для оценок, принадлежащих рейтинговому интервалу «хорошо», осуществимо с использованием признака $Пр1,3$ наличия снижающей оценки (в данном случае «удовлетворительно»):

$Пр1,3 = Xп \cdot Vп,3 + Xт \cdot Vт,3,$

где $Vп,3 = \begin{cases} 0, \text{ если } Xп \geq 3,5, \\ 1 \text{ в ином случае;} \end{cases}$

$Vт,3 = \begin{cases} 0, \text{ если } Xт \geq 3,5, \\ 1 \text{ в ином случае.} \end{cases}$

Наличие среди частных оценок $Xп$, $Xт$ хотя бы одной оценки «удовлетворительно» приводит к положительному значению признака наличия снижающей оценки ($Пр1,3$) за счет умножения хотя бы на один изменяемый весовой коэффициент $Vп,3$ или $Vт,3$, равный в этом случае единице, либо к нулю при отсутствии снижающих оценок «удовлетворительно» за счет умножения на изменяемые весовые коэффициенты $Vп,3$ и $Vт,3$, равные в этом случае нулю.

С помощью изменяемых весовых коэффициентов $V3,1$ и $V3,2$ и признака наличия снижающей оценки ($Пр1,3$) предварительное значение оценки подготовленности специалиста может быть получено как $R''хор1$ или $R''хор2$, причем в зависимости от значений изменяемых весовых коэффициентов $V3,1$ и $V3,2$ либо в виде доминирующего отрицательного числа, либо в виде значения $R'хор$.

Математическими соотношениями, описывающими метод определения снижающих оценок, являются следующие: $R''хор1 = Rхор + Пр1,3V3,1$ и $R''хор2 = Rхор + Пр1,3V3,2$, где

$V3,1 = \begin{cases} -100, \text{ если } Пр1,3 > 0, \\ 0 \text{ в ином случае;} \end{cases}$

$V3,2 = \begin{cases} 0, \text{ если } Пр1,3 > 0, \\ -100 \text{ в ином случае.} \end{cases}$

Реализация соотношений $R''хор1$ и $R''хор2$ представлена на рисунке 2.

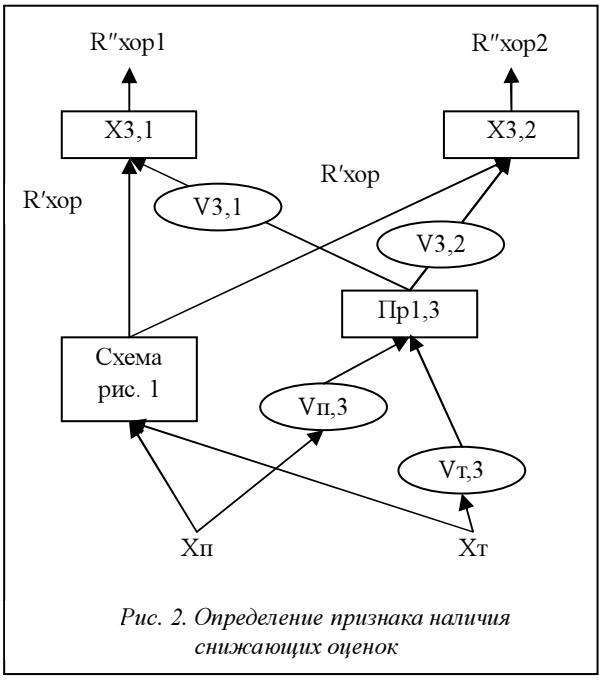


Рис. 2. Определение признака наличия снижающих оценок

Аналогично может быть реализован метод определения признака наличия снижающих оценок и в интервале «удовлетворительно».

Получение итогового значения рейтинговой оценки

Как при отсутствии, так и при наличии снижающих оценок формируется только одно предварительное значение рейтинговой оценки для соответствующего диапазона рейтингового интервала оценки (для рассматриваемого примера – интервал 3,5≤«хорошо»<4,5).

Однако зачастую предварительное значение $R''хор1$ или $R''хор2$, хотя и попадает в нужный диапазон соответствующего рейтингового интервала, из-за логического условия оценивания (при разных сочетаниях входных оценок) по абсолютной величине может выйти за границы установленного рейтингового интервала.

Для приведения полученного промежуточного значения рейтинговой оценки к установленным границам интервалов (диапазонов) воспользуемся методом линейных преобразований получаемых промежуточных значений рейтинговой оценки.

Для интервала «хорошо» максимальные оценки за теорию и практику могут быть равными соответственно «отлично» и «хорошо», а минимальные – «удовлетворительно» и «хорошо». Исходя из устанавливаемых рейтинговых интервалов (1) и (2) получения оценки «хорошо», возможные достигаемые максимальное и минимальное значения среднего балла оценок специалиста соответственно определяются как

$$\frac{5 + 4,49(9)}{2} = 4,749(9), \quad (7)$$

$$\frac{2,5 + 3,5}{2} = 3,0. \quad (8)$$

Таким образом, реально вычисляемые в виде среднего балла возможные значения итоговой рейтинговой оценки «хорошо» могут оказаться в интервале 3,0≤ «хорошо» ≤ 4,749(9), в то время как установленный рейтинговый интервал (2) «хорошо» должен быть равен 3,5≤ «хорошо» < 4,5.

Линейные преобразования получаемых предварительных значений рейтинговых оценок должны обеспечить такую коррекцию их значений, чтобы они не выходили за границы установленных рейтинговых интервалов. При этом, если оценки за теорию и практику окажутся не ниже 4,0 (чему соответствует $R''хор1 \neq 0$ и $R''хор2 = 0$), в соответствии с (5) итоговая рейтинговая оценка специалиста должна быть в верхнем диапазоне рейтингового интервала «хорошо», а если хотя бы одна из оценок окажется ниже 4,0 (чему соответствует $R''хор1 = 0$ и $R''хор2 \neq 0$), рейтинговая оценка специалиста должна быть в нижнем диапазоне рейтингового интервала (6) «хорошо».

Коррекция обозначаемых через x полученных предварительных значений рейтинговых оценок $R''хор1$ и $R''хор2$ выполнима с помощью линейного преобразования, описываемого функцией (y) одной переменной (x) вида:

$$y=kx+b. \quad (9)$$

Учитывая, что из логических условий оценивания достигаемые максимально возможные значения y и x в случае верхнего диапазона (5) равны соответственно 4,499(9) и 4,749(9), а минимально возможные равны 4,0 и 4,0, составим систему двух линейных уравнений вида

$$\begin{cases} 4,499(9) = k1 \cdot 4,749(9) + b1, \\ 4,0 = k1 \cdot 4,0 + b1. \end{cases} \quad (10)$$

Результаты решения системы уравнений – $k1 \approx 0,665342204$ и $b1 \approx 1,338631184$. На основании этого соотношение (9) для верхнего диапазона рейтингового интервала «хорошо» принимает вид

$$y=0,665342204x+1,338631184. \quad (11)$$

Значения $k2$ и $b2$ для нижнего диапазона рейтингового интервала «хорошо» определяются аналогично.

Убедимся на конкретном примере, что с использованием линейного преобразования (11) реализуется логическое условие получения оценки «хорошо».

Пусть частная оценка практических навыков специалиста $X_П=4,49$ («хорошо»), а теории – $X_Т=5,0$ («отлично»). Средний балл его подготовленности $R'хор=R''хор1=4,745$, что, согласно (1), соответствует рейтинговому интервалу $4,5 \le$«отлично»≤ 5.

Однако по нормативным условиям получения оценки требуется, чтобы интегральная итоговая оценка $Rxop$ специалиста была не выше оценки за практику, в данном примере равной «хорошо».

Подставив полученную промежуточную рейтинговую оценку $R''хор1=4,745$ в (11), получим $Rxop=4,745 \times 0,665342204+1,338631184 \approx 4,496$, то есть итоговая рейтинговая оценка специалиста находится в требуемом верхнем диапазоне рейтингового интервала «хорошо», что соответствует заданным нормативным условиям оценивания.

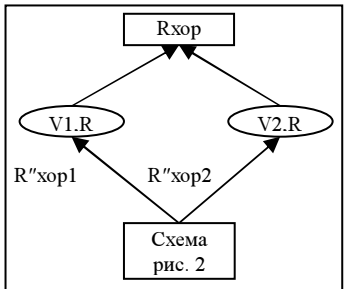


Рис. 3. Получение итоговой рейтинговой оценки

Реализация коррекции выполнима с помощью весовых коэффициентов:

$$V1,R = \begin{cases} (k1\ X3,1 + b1), \text{ если } R''хор2 > 0, \\ 0 \text{ в ином случае;} \end{cases}$$

$$V2,R = \begin{cases} 0, \text{ если } R''хор1 > 0, \\ (k2\ X3,1 + b2) \text{ в ином случае.} \end{cases}$$

Учитывая, что промежуточное значение оценки находится в одном из диапазонов рассматриваемого рейтингового интервала, математическое соотношение, описывающее метод получения итоговой рейтинговой оценки «хорошо» ($Rxop$), будет иметь вид

$$Rxop = V1,R \cdot R''хор1 + V2,R \cdot R''хор2. \quad (12)$$

Математическое соотношение (12) может быть представлено фрагментом структурной схемы, отображенной на рисунке 3.

Аналогично могут быть построены и реализованы соотношения для вычисления рейтинговых оценок R(*отл*), R(*уд*) и R(*неуд*).

Литература

1. Долгов А.И., Мартыненко А.Ф., Преснухин В.В. Научные основы построения коэффициентных методик оценки объектов // Системы управления и информационные технологии. 2008. № 4 (34). С. 61–66.

2. Государственная система обеспечения единства измерений. Шкалы измерений. М.: Стандартинформ, 2008.

3. Долгов А.И., Мартыненко А.Ф., Преснухин В.В. Коэффициентная методика с изменяемыми весовыми коэффициентами // Искусственный интеллект. 2008. Т. 4.

УДК 007:519.816

РАСПРЕДЕЛЕННАЯ ИНФОРМАЦИОННАЯ СИСТЕМА ПОИСКА ЛИТЕРАТУРЫ

М.М. Маран, к.т.н.; Лвин Маунг Со

(Национальный исследовательский университет «Московский энергетический институт», mm@appmat.ru, lwinmgsoe@gmail.com)

Рассматриваются возможности построения и реализация распределенной информационной системы поиска литературы. Реализация выполнена согласно концепции сервис-ориентированной архитектуры на платформах Windows Communication Foundation, Microsoft SQL Server 2008 и Visual Studio 2010.

Ключевые слова: *информационная система, поиск литературы, SOA, WCF, ADO NET MVC.*

Предлагаемая информационная система обеспечивает удобный доступ через веб-интерфейс к каталогу полнотекстовых документов и мультимедийных ресурсов, а также полнотекстовый поиск и

поиск по их атрибутам [1]. Кроме того, администраторы библиотеки имеют возможность редактирования ресурсов библиотеки.

Реализация выполнена в среде Microsoft Visual Studio 2010 с использованием технологий Asp.net MVC 3, ADO.NET Entity Framework, Windows Communication Foundation (WCF). Рассмотрим эти технологии подробнее, отметив их преимущества.

Технологии реализации

Платформа ASP.NET MVC представляет собой удобную среду создания web-приложений, интегрированную с существующими функциями ASP.NET и дополненную средствами MVC (model–view–controller) (*модель–представление–контроллер*):

– *модель (model)* предоставляет данные для *представления* в ответ на запросы *контроллера*, содержит бизнес-логику приложения;

– *представление (view)* отвечает за пользовательский интерфейс, отображает данные, полученные от *модели;*

– *контроллер (controller)* обрабатывает команды пользователя, определяет *модель* для работы и связывает ее с *представлением*.

Бизнес-логика, расположенная в *модели*, включает все правила и алгоритмы, связанные с предметной областью решаемой задачи. Проще говоря, это ядро создаваемого приложения. Рассматриваемая архитектура подразумевает, что изменения в любом из компонентов оказывают минимальные воздействия на остальные части. Несколько упрощая, работу MVC-приложения можно описать следующим образом:

– команда (уведомление о нажатии кнопки, запрос адреса сайта и т.д.) передается *контроллеру*;

– *контроллер*, исходя из полученных данных, определяет и вызывает *модель*;

– *модель* на основе заложенной в ней бизнес-логики формирует набор данных;

– *контроллер* выбирает *представление* и связывает его с данными (*моделью*);

– *представление* отображает данные пользователю.

Контроллер играет роль связующего звена между *моделью* и *представлением*. При этом он стремится как можно меньше знать о подробностях их реализаций. Его задача – определить *модель* для обработки полученной команды и *представление*, которому будут отправляться итоговые данные.

Представление зависит от *модели*, так как использует получаемые от нее данные. *Модель* не зависит ни от *представления,* ни от *контроллера.* Это позволяет вести разработку *модели* независимо, а также создавать для нее несколько *представлений*.

Платформа ASP.NET MVC имеет следующие преимущества:

– облегчает управление сложными структурами путем разделения приложения на *модель*, *представление* и *контроллер*;

– не использует состояние просмотра и серверные формы, что делает эту платформу идеальной для разработчиков, которым необходим полный контроль над поведением приложения;

– использует схему основного *контроллера*, при которой запросы web-приложения обрабатываются через один *контроллер*, что позволяет создавать приложения, поддерживающие расширенную инфраструктуру маршрутизации, а также обеспечивать расширенную поддержку разработки на основе тестирования;

– хорошо подходит для web-приложений, поддерживаемых крупными коллективами разработчиков, которым необходим высокий уровень контроля над поведением приложения.

Платформа ADO.NET Entity Framework (EF) – объектно-ориентированная технология доступа к данным, являющаяся объектно-реляционным отображением (object-relational mapping, ORM) решений для .NET Framework от Microsoft. Она предоставляет возможность взаимодействовать с объектами как посредством LINQ в виде LINQ to Entities, так и с использованием Entity SQL. Для облегчения построения web-решений используются и ADO.NET Data Services (Astoria), и связка из Windows Communication Foundation и Windows Presentation Foundation, позволяющая строить многоуровневые приложения, реализуя один из шаблонов проектирования – MVC, MVP или MVVM [2]. Платформа ADO.NET Entity Framework дает возможность разработчикам создавать приложения для доступа к данным, работающие с концептуальной моделью приложения, а не напрямую с реляционной схемой хранения. В результате можно добиться уменьшения объема кода и снижения затрат на сопровождение приложений, ориентированных на обработку данных.

Таким образом, основной задачей ORM является установка соответствия между объектами, используемыми в приложении, и таблицами, хранящимися в реляционных БД.

Создание SQL-запросов. При запросе приложением объектов ORM-библиотека самостоятельно создает SQL-код запросов и передает его в СУБД. При необходимости разработчик может вмешаться в данный процесс с целью тонкой оптимизации производительности. При этом возможно возникновение проблемы несоответствия типов. В качестве решения ORM при записи значения может использовать свойство объекта.

Абстракция используемой БД. В процессе работы с ORM-библиотекой приложение оперирует привычными для него объектами. Но при этом для хранения информации могут использоваться раз-

личные реляционные СУБД, в частности, SQL Server, SQL Server Express, SQL Server Compact, mySQL и т.д. Это вносит дополнительный уровень гибкости в архитектуру приложения.

Приложения Entity Framework имеют следующие преимущества:

– могут работать с концептуальной моделью в терминах предметной области, в том числе с наследуемыми типами, сложными элементами и связями;

– освобождаются от жестких зависимостей от конкретного ядра СУБД или схемы хранения;

– сопоставления между концептуальной моделью и схемой, специфичной для конкретного хранилища, могут меняться без изменения кода приложения;

– разработчики имеют возможность работать с согласованной моделью объектов приложения, которая может быть сопоставлена с разными схемами хранения, возможно, реализованными в различных СУБД;

– несколько концептуальных моделей могут быть сопоставлены с единой схемой хранения;

– поддержка запросов LINQ обеспечивает проверку синтаксиса во время компиляции для запросов к концептуальной модели.

Технология Windows Communication Foundation (WCF) [3] предоставляет единую инфраструктуру разработки, повышающую производительность и снижающую затраты на создание безопасных, надежных и транзакционных web-служб нового поколения.

WCF – это технология для построения сервис-ориентированной архитектуры приложений (SOA – service-oriented architecture), что позволяет абстрагироваться от конкретной технологии, на которой этот сервис реализован, и пользоваться им из других приложений, написанных на любых других платформе, языке, технологии, – главное, чтобы реализация клиента отвечала определенным правилам. Кроме того, логика самого сервиса и его реализация полностью отделены от коммуникационной составляющей и существует возможность декларативно изменять способ взаимодействия с сервисом путем изменения конфигурационного файла. Можно изменить протокол взаимодействия, адрес, настроить максимальное количество подключений, ограничить размер пакетов, тайм-аут подключения к сервису и выполнения операции и многое другое.

В основе технологии WCF лежит принцип связи с помощью обмена сообщениями, и любые объекты, моделируемые в виде сообщений (например, HTTP-запрос или сообщение очереди сообщений, MSMQ), можно представить единым образом в модели программирования. Это обеспечивает универсальный интерфейс API для разных транспортных механизмов. Сообщения можно отправлять через интрасети или через Интернет общими транспортами, такими как HTTP и TCP. С помощью встроенных точек расширения WCF добавляются дополнительные транспортные механизмы. Служба WCF поддерживает несколько шаблонов обмена сообщениями, включая запрос–ответ, одностороннюю и дуплексную связь. Разные транспорты поддерживают разные шаблоны обмена сообщениями и таким образом влияют на типы поддерживаемых взаимодействий. Интерфейсы API и среда выполнения WCF также помогают отправлять сообщения безопасно и быстро.

При создании приложений для .NET разработчики пользуются средой Visual Studio. В WCF и в Visual Studio есть инструменты для реализации служб. В WCF встроена модель размещения, позволяющая размещать службы в IIS или в Managed Services на платформе Windows. WCF поддерживает развитую модель многопоточности и ограничения пропускной способности (throttling), которая позволяет управлять созданием экземпляров с минимальными усилиями. Вне зависимости от того, обрабатываются ли поступающие запросы одновременно в одном или в нескольких потоках, модель программирования остается одинаковой, так что разработчик может не вдаваться в детали (которые, однако, остаются ему доступными). WCF поддерживает различные способы обмена сообщениями, например: запрос–ответ, односторонний и дуплексный поток. Поддерживаются также пиринговые сети, в которых клиенты могут обнаруживать друг друга и обмениваться данными при отсутствии централизованного механизма управления. Словом, технология WCF важна потому, что современные приложения немыслимы без служб, а именно это и составляет назначение и смысл WCF.

Структура программы eLibrary

eLibrary – распределенная информационная система, состоящая из взаимодействующих в локальной сети подсистем (рис. 1). Каждая подсистема функционирует как собственная информационная система, у подсистем имеются своя СУБД и свои приложения. Подсистемами в рассматривае-

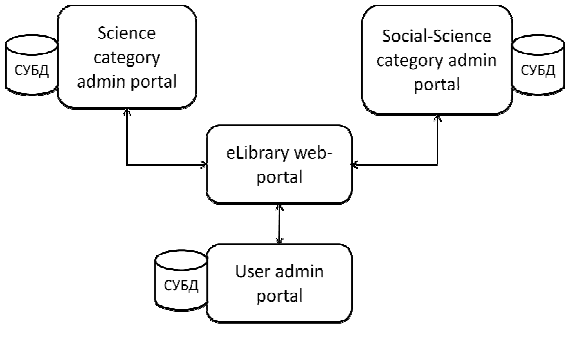


Рис. 1. Архитектура системы eLibrary

мом случае являются Science category admin portal, Social-Science category admin portal, User admin portal, eLibrary web-portal.

Рассмотрим кратко структуру и назначение подсистем.

Подсистема Science category admin portal (см. рис. 2) создана для выполнения действия с документами по категории «Научный» в библиотеке. В ней функционирует СУБД Microsoft SQL Server 2008. В подсистеме имеется одно web-предложение, с помощью которого администраторы могут работать с данными. Для взаимодействия с другими подсистемами в ней также размещен реализованный на WCF web-сервис. Этот сервис поддерживает способ обмена данными и выполнения операций в формате сообщений SOAP.

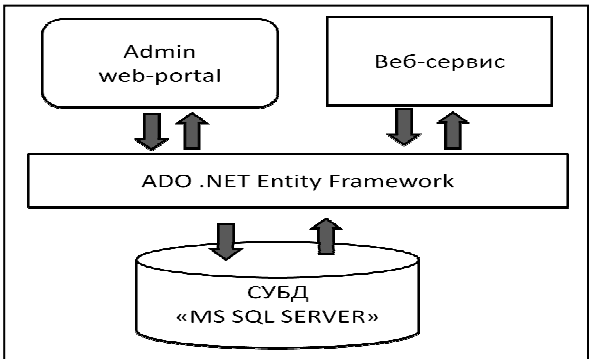


Рис. 2. Архитектура подсистемы Science category admin portal

Как показано на рисунке 2, для взаимодействия с данными СУБД Microsoft SQL Server 2008 использована платформа ADO.NET Entity Framework. Для ее функционирования разработано представление реляционных таблиц в виде классов. Web-приложение создано на основе технологии ASP.NET MVC Framework на языке C# в среде Microsoft Visual Studio 2010.

Подсистема Social-Science category admin portal. Архитектура и функции данной подсистемы такие же, как у подсистемы Science category admin portal. В БД этой подсистемы хранятся документы по общественным наукам.

Подсистема User admin portal предназначена для управления данными об администраторах и пользователях библиотеки. Ее архитектура аналогична описанной выше, различаются структура данных и их представлений в виде классов.

Подсистема eLibrary web-portal (рис. 3) является клиентом для пользователей библиотеки. В ней только одно web-приложение, позволяющее пользователям найти все документы в библиотеке. В этой подсистеме нет собственной БД, нужные данные можно получить из других подсистем с помощью их web-сервисов. Для этого в системе создан специальный слой, который управляет всеми работами с данными.

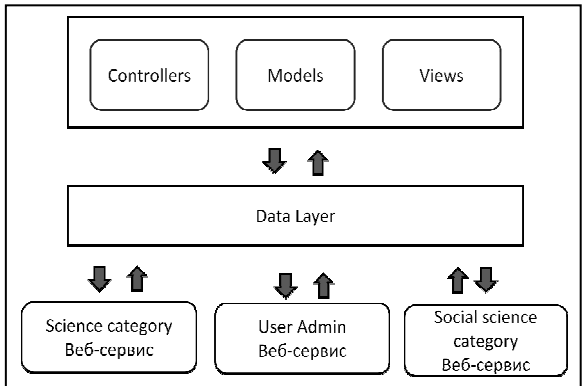


Рис. 3. Архитектура подсистемы eLibrary web-portal

При поступлении запроса пользователя Data Layer посылает запросы в web-сервисы, а при получении ответов из web-сервисов передает их в Controller, затем пользователю (рис. 4). С помощью этого сайта пользователь может найти материалы по названию, по издателю, по авторам и по ключевым словам. Кроме этого, полученные результаты могут фильтроваться, например, по книгам или по статьям. Для просмотра документов пользователю необязательно авторизоваться в системе, а для получения материалов авторизация обязательна.

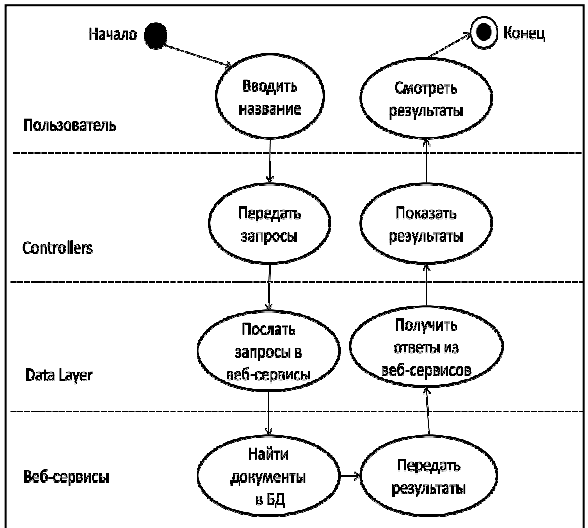


Рис. 4. Диаграмма поиска документов по названию

Предложенная информационная система реализована на широко используемых платформах и дает возможность пользователям вести поиск литературы по различным критериям.

Литература

1. Таненбаум Э., Ван Стеен М. Распределенные системы. Принципы и парадигмы. СПб: Питер, 2003. 845 с.

2. Josuttis N.M. SOA in Practice: The Art of Distributed System Design (Theory in Practice). O'Reilly Media. 2007. 759 p.

3. WCF 4 для профессионалов / Сибарро П. [и др.]. М.: Диалектика, 2011. 464 с.

УДК 004.8

МЕХАНИЗМ РЕАЛИЗАЦИИ МАНДАТНОЙ ПОЛИТИКИ БЕЗОПАСНОСТИ В АСУП НА БАЗЕ ORACLE ECM 11G

М.А. Проскуряков; Б.В. Палюх, д.т.н.; В.В. Мельникова
(Тверской государственный технический университет);
С.Л. Котов, к.э.н. (Государственный испытательный центр ПС ВТ, г. Тверь, gic@tvcom.ru)

Рассматривается проблема обеспечения соответствия подсистемы мандатного контроля доступа в Oracle ECM 11g требованиям РД СВТ, предъявляемым к данной подсистеме. Предложен механизм реализации мандатной политики разграничения доступа в Oracle ECM 11g.

Ключевые слова: *Oracle ECM 11g, РД СВТ, мандатная модель Белла–ЛаПадулы, аттестация, контент-сервер, АСУП, сертификационные испытания.*

Проблему обеспечения безопасности *информационных систем* (ИС) можно рассматривать с учетом двух компонент – автоматизации обработки информации и общей безопасности.

Отметим, что основная аксиома защиты информации формулируется следующим образом: все вопросы безопасности информации описываются доступом субъектов к объектам. Поэтому политика безопасности задается в виде правил, в соответствии с которыми должно осуществляться взаимодействие между субъектами и объектами. Взаимодействия, приводящие к нарушениям этих правил, необходимо пресекать средствами контроля доступа. Среди моделей политик безопасности можно выделить два основных класса: дискреционные (произвольные) и мандатные (нормативные). В данной статье рассматривается фундаментальная нормативная модель Белла–ЛаПадулы [1].

При описании мандатной модели разграничения доступа будем использовать такой подход к формальному моделированию безопасности ИС, при котором исследуемая система представляется в виде абстрактной системы (автомата), каждое состояние которой описывается доступами, реализуемыми субъектами к сущностям, а переходы ИС из состояния в состояние – командами или правилами преобразования состояний, выполнение которых обычно инициируется субъектами [2].

Классическая модель Белла–ЛаПадулы, используемая, в частности, в *Руководящем документе «Средства вычислительной техники…»* (РД СВТ) (*http://www. fstec.ru/_docs/doc_3_3_003.htm*), является автоматной моделью, в которой моделируемая ИС представляется абстрактной системой, каждое ее состояние описывается с использованием следующих обязательных составляющих:

- множества текущих доступов субъектов к объектам системы;
- функций, задающих для каждого субъекта уровень доступа и текущий уровень доступа, для каждого объекта уровень конфиденциальности;
- матрицы доступов, позволяющей в дополнение к мандатному управлению доступом использовать дискреционное управление им.

Приведем формальное описание мандатной модели разграничения доступа, которая должна быть реализована в АСУП в соответствии с требованиями РД СВТ: S – множество субъектов s_1, s_2, s_3, …, s_n; O – множество объектов o_1, o_2, o_3, …, o_m, $S \subset O$; $R=\{r, w, d\}$ – множество прав доступа, где r – доступ на чтение, w – на запись, d – на удаление; $L=\{U, SEC, TOPS, GRS\}$ – множество уровней секретности (иерархических категорий), где U – несекретно, SEC – секретно, $TOPS$ – совершенно секретно, GRS – особой важности; $\Lambda=\{L, \leq, \bullet, \otimes\}$ – решетка уровней секретности, где $\leq$ – оператор, определяющий частичное нестрогое отношение порядка для уровней секретности, $\bullet$ – оператор наименьшей верхней границы, $\otimes$ – оператор наибольшей нижней границы; V – множество состояний системы, представляемое в виде набора упорядоченных пар (F, M), где $F : S \cup O \rightarrow L$ – функция уровней секретности, ставящая в соответствие каждому объекту и субъекту в системе определенный уровень секретности, M – матрица текущих прав доступа $\{S_i, O_i\}$.

Таким образом, система $\Sigma=(v_0, R, T)$ в модели Белла–ЛаПадулы состоит из следующих элементов: v_0 – начальное состояние системы, R – множество прав доступа, T: $V \times R \rightarrow V$ – функция перехода, которая при выполнении запросов переводит систему из одного состояния в другое [2].

Рассмотрим реализацию модели Белла–ЛаПадулы на примере системы управления контентом Oracle Enterprise Content Management Suite 11g (далее – ECM 11g). В ECM 11g для настройки мандатных *правил разграничения доступа* (ПРД) стандартными средствами в соответствии с требованиями РД СВТ необходимо выполнить следующие основные настройки.

1. Прежде всего необходимо перейти в апплет «Управление пользователями» → «Защита» → «Предварительно определенные учетные записи», создать неиерархические и иерархические категории, указанные в таблице 1 (в ECM 11g категории называются учетными записями).

Таблица 1

Описание неиерархических и иерархических категорий

Категория	Примечание
grs	Иерархическая категория «Особой важности» внутри неиерархической категории grs
grs/tops	Иерархическая категория «Совершенно секретно» внутри неиерархической категории grs
grs/tops/sec	Иерархическая категория «Секретно» внутри неиерархической категории grs
test1	Неиерархическая категория test1, которая не содержит иерархических категорий

Графическое представление неиерархических и иерархических категорий в ECM 11g дано на рисунке 1.

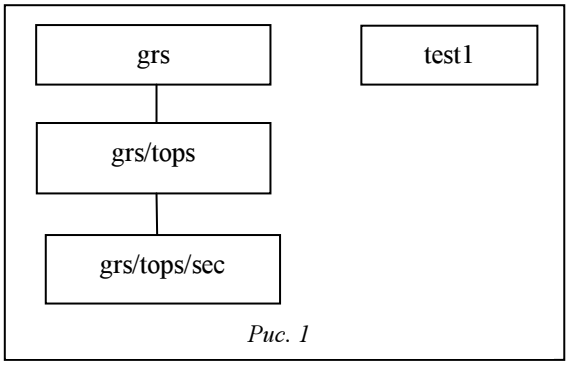


Рис. 1

2. В LDAP-каталог сервера приложений Oracle Weblogic 11g следует добавить необходимые группы, соответствующие правам доступа пользователей к иерархическим категориям. Структура имен создаваемых групп следующая:

@[Имя категории из таблицы 1]([Права доступа к данной иерархической категории])

Примеры: @grs(*W*), @grs/tops/sec(*RWD*), где @ указывает на то, что данная группа в Weblogic соответствует категории в ECM 11g; имя категории между символами @ и (должно точно совпадать с названием учетной записи в ECM11g; *R*, *W*, *D* – параметры полномочий пользователей к документам, относящимся к текущей иерархической категории (*R* – чтение, *W* – запись, *D* – удаление); права пользователей к категориям задаются в круглых скобках сразу после имени категории.

3. Далее необходимо включить пользователей в созданные на предыдущем шаге группы пользователей. Модель мандатных ПРД, согласно РД СВТ, запрещает нисходящий информационный поток в соответствии с моделью Белла–ЛаПадулы, то есть пользователю запрещается запись данных в объекты с более низким уровнем доступа, чем у него самого (уровень определяется набором неиерархической и иерархической категорий), а также чтение данных из объектов с более высоким уровнем доступа, чем у него.

Для обеспечения данного требования необходимо включить пользователей в группы, представленные в таблице 2.

Таблица 2

Настройка групп пользователей в соответствии с их уровнем доступа

Категория пользователя в ECM 11g	Группы Weblogic	Описание
grs/tops/sec	@grs(W) @grs/tops/sec(RWD)	Ожидается, что пользователь с иерархической категорией «Секретно» имеет право на запись документов в объекты с более высокой меткой (в иерархические категории «Совершенно секретно» и «Особой важности») и в объекты с такой же меткой, как у него («Секретно»), а читать и удалять данные может только из объектов с категорией «Секретно»
grs/tops	@grs(W) @grs/tops(RWD) @grs/tops/sec(RD)	Ожидается, что пользователь с иерархической категорией «Совершенно секретно» имеет право на запись документов в объекты с более высокой меткой (категории «Особой важности») и в объекты с такой же меткой, как у него («Совершенно секретно»), а читать и удалять данные может из объектов с категориями «Секретно» и «Совершенно секретно»
grs	@grs(RWD) @grs/tops(RD) @grs/tops/sec(RD)	Ожидается, что пользователь с иерархической категорией «Особой важности» имеет право на запись данных только в объекты с такой же меткой, как у него («Особой важности»), а читать и удалять данные может из объектов с более низкими иерархическими категориями «Совершенно секретно» и «Секретно»

В действительности описанных выше настроек мандатных ПРД в ECM 11g недостаточно для реализации всех требований к мандатной модели согласно РД СВТ, поскольку иерархические категории в ECM устроены таким образом, что права, назначенные корневой категории (в данном случае корневой категорией является категория grs – «Особой важности»), автоматически назначаются всем вложенным категориям (в данном случае категориям grs/tops и grs/tops/se). При описанных выше настройках начальное состояние (F, M) мандатной модели разграничения доступа в ECM 11g будет безопасно по чтению: $\forall s \in S$, $\forall o \in O$, $r \in M[s, o] \rightarrow F(o) \leq F(s)$, но не будет безопасно по записи, то есть не будет соблюдаться следующее правило: $\forall s \in S$, $\forall o \in O$, $w \in M[s, o] \rightarrow F(s) \leq F(o)$. Таким обра-

зом, стандартная мандатная модель разграничения доступа в ECM 11g не соответствует требованиям РД СВТ.

Для обеспечения безопасности мандатной системы разграничения доступа в ECM 11g как по чтению, так и по записи необходимо модернизировать стандартную мандатную модель разграничения доступа в ECM 11g, добавив пользователям во встроенном LDAP-каталоге Weblogic дополнительный атрибут, имеющий условную метку и по сути дублирующий иерархическую категорию пользователя. Согласно этой метке, в профиле регистрации новых документов на контент-сервере ECM 11g по заданному алгоритму в списке выбора категорий для регистрируемого пользователем документа будут отображаться только те иерархические категории, которые не ниже иерархической категории пользователя.

Модернизация мандатной системы разграничения доступа в ECM 11g будет состоять из следующих основных этапов.

1. Добавление и настройка пользовательского JPS-атрибута.

1.1. Выбрать в LDAP-каталоге JPS-атрибут для пользователя, в котором будет храниться дублирующая информация об уровне доступа пользователя согласно мандатной модели разграничения доступа (комбинация неиерархической и иерархической категорий). В качестве примера возьмем пользовательский атрибут Employeenumber во встроенном LDAP-каталоге Weblogic (атрибут JPS).

1.2. Создать новое информационное поле uEmployeenumber на вкладке «Информационные поля» апплета «Управление пользователями», который должен принимать значение пользовательского атрибута Employeenumber из встроенного LDAP-каталога Oracle Weblogic 11g. Для этого в настройках поставщика пользователей JPS (на вкладке «Поставщики» в ECM 11g) необходимо настроить соответствие между пользовательским атрибутом во встроенном LDAP-каталоге и информационным полем в апплете «Управление пользователями» в ECM 11g.

2. Настройка соответствия значения пользовательского JPS-атрибута уровню доступа пользователя в мандатной модели разграничения доступа.

2.1. В апплете «Диспетчер конфигураций» ECM 11g необходимо создать таблицу Checklist, содержащую информационные поля, отраженные в таблице 3.

Таблица 3

Информационные поля таблицы Checklist

Имя	Тип	Длина	Первичный
docAccount	varchar	50	
ID	int		Да
employeenumber	varchar	50	

2.2. Создать представление Checklist для таблицы Checklist и заполнить его данными, приведенными в таблице 4.

Таблица 4

Содержание представления Checklist

ID	docAccount	employeenumber
1	grs	0
2	grs/tops	0
3	grs/tops/sec	0
4	grs	1
5	grs/tops	1
6	grs	2
7	test1	3

2.3. Перейти на вкладку «Сервер администратора» → «Диспетчер компонентов» → «Расширенный диспетчер компонентов» и установить новые компоненты RemoveProfileStandardMenus.zip для удаления стандартного профиля регистрации документов и UserDataSecurityFilter.zip для пользовательского модуля реализации безопасности с целью обеспечения фильтрации доступных пользователям категорий безопасности при регистрации документов на контент-сервере согласно значению атрибута Employeenumber.

2.4. Открыть вкладку «Защита» созданного ранее представления Checklist и указать имя пользовательского класса intradoc.server.schema.UserDataSecurityFilter.

Рис. 2. Схема выполнения регистрации документов

2.5. Найти файл [cs_home]/data/schema/views/CheckList.hda и добавить строчки:

– schSecurityImplementorUserDataField = employeenumber;

– schSecurityImplementorUserDataValue = uEmployeenumber.

3. Настройка профиля для регистрации документов согласно заданным на предыдущих шагах правилам.

3.1. Создать информационное поле xwCategory на вкладке «Информационные поля» апплета «Диспетчер конфигураций» и настроить для данного информационного поля список вариантов со ссылкой на представление Checklist.

3.2. В правиле gStandardFields, отвечающем за отображение и обработку стандартных атрибутов для загружаемых на сервер документов, обязательному атрибуту xwCategory необходимо указать следующее значение для параметра «Является производным полем» – <$dprDerivedValue=#xwCategory.docAccount$> и скрытому атрибуту dDocAccount указать следующее значение для параметра «Является производным полем» – <$dprDerivedValue=#getFieldViewValue("xwCategory",#active.xwCategory,"docAccount")$>.

Схема работы модернизированной мандатной модели разграничения доступа в ECM 11g при выполнении регистрации документов пользователями на контент-сервере представлена на рисунке 2.

Таким образом, начальное состояние (F, M) модернизированной мандатной модели разграничения доступа в ECM 11g будет безопасно как по чтению: $\forall s \in S, \forall o \in O, r \in M[s, o] \rightarrow F(o) \leq F(s)$, так и по записи: $\forall s \in S, \forall o \in O, w \in M[s, o] \rightarrow F(s) \leq F(o)$ [2]. Безопасность всех остальных состояний модернизированной системы $\Sigma=(v_0, R, T)$ мандатного принципа разграничения доступа в ECM 11g будет обеспечиваться тем, что в ECM 11g изменять уровни безопасности пользователей – группы и значения служебных атрибутов пользователей в LDAP-каталоге Weblogic – разрешено только пользователям, играющим роль администратора Weblogic, другие пользователи ECM не имеют прав доступа к консоли администрирования Weblogic. Доступ к апплетам администрирования ECM 11g также разрешен только пользователям, играющим роль администрирования ECM 11g. Следовательно, можно утверждать, что модернизированная система $\Sigma=(v_0, R, T)$ мандатного принципа разграничения доступа в ECM 11g является безопасной, поскольку ее начальное состояние v_0 и другие состояния системы, достижимые из начального состояния v_0 путем применения конечной последовательности запросов из R, безопасны. Практическое применение модернизированной мандатной модели разграничения доступа в ECM 11g, несмотря на наличие сертификата безопасности, требует, чтобы модель безопасности на аттестуемых АСУП также была настроена в соответствии с разработанными в данной статье требованиями.

Литература

1. Палюх Б.В., Котов С.Л., Демирский А.А. Тестирование ПС: учеб. пособие. Тверь: ТГТУ, 2011. 134 с.

2. Девянин П.Н. Модели безопасности компьютерных систем: учеб. пособие для вузов. М.: Изд-во «Академия», 2005. 144 с.

3. Игнатьев В.А. Информационная безопасность современного коммерческого предприятия : монография. Старый Оскол: ООО «ТНТ», 2005. 448 с.

4. Котов С.Л., Палюх Б.В., Федченко С.Л. Методы оценки, тестирования и выбора рациональных характеристик корпоративных информационных систем: учеб. пособие. Тверь: ТГТУ, 2008.

УДК 004.942+896

ПРОГРАММНАЯ ИМИТАЦИОННАЯ МОДЕЛЬ MACHINA SPECULATRIX УОЛТЕРА ГРЕЯ

П.А. Колос; Н.С. Волкова

(Черкасский национальный университет им. Богдана Хмельницкого, dr_peter@i.ua, manunya@i.ua)

Статья посвящена особенностям моделирования поведения живых существ при создании технических систем. Описываются биоморфные кибернетические робототехнические системы machina speculatrix Уолтера Грея, а также исследуются особенности разработки и использования имитационных моделей таких систем.

Ключевые слова: *бионика, кибернетические черепашки, робототехнические системы, биоморфные роботы, программные имитационные модели.*

Управление техническими системами – главная задача кибернетики. Современные кибернетические технологии с этой целью очень часто используют бионический подход.

Бионика (в англоязычной литературе – биомиметика) – научно-технологическое направление по заимствованию у природы ценных идей и реализации их в виде конструкторских и дизайнерских

решений, а также новых информационных технологий [1].

Различают следующие виды бионики:

– биологическую, занимающуюся изучением происходящих в биологических системах процессов;

– теоретическую, строящую математические модели этих процессов;

– техническую, применяющую модели теоретической бионики для решения инженерных задач.

Лучше всего бионический подход проявляет себя при решении задач управления в ситуации неопределенности, в постоянно меняющихся условиях.

Бионический подход для управления техническими системами впервые нашел применение при конструировании робототехнических систем на заре возникновения такой науки, как кибернетика. Современные информационные технологии так же широко начали использовать особенности моделирования живой природы при разработке разного рода интеллектуальных информационных систем. Программная имитация первых воплощавших идеи бионики технических систем позволила развивать их и совершенствовать системы, которые в своем функционировании используют такие же принципы.

Черепашки Уолтера Грея

Бионический подход при создании технических систем впервые использовал английский нейрофизиолог Вильям Грей Уолтер, разрабатывая machina speculatrix – биоморфные кибернетические роботы, которые по своему внешнему виду и медлительности очень напоминали черепах, а потому в историю робототехники вошли под названием «черепашки» [2]. Свои исследования над machina speculatrix Уоллтер Грей проводил с 1948 по 1951 гг.

Черепашки представляли собой самодвижущиеся электромеханические тележки, способные ползти на свет или от него, обходить препятствия, заходить в «кормушку» для подзарядки разрядившихся аккумуляторов.

Всего Уолтер Грей создал более восьми черепашек. Первая из них, Элмер [2], была выполнена в виде небольшой трехколесной тележки с установленными на ней двумя электродвигателями, питающимися от аккумуляторов. Один двигатель обеспечивал поступательное движение устройства, другой – менял направление движения. Управление двигателями осуществлялось с помощью электромагнитных реле. Чувствительными элементами были фотоэлементы и механический контакт, замыкающийся при наезде на препятствие. Внешний вид черепашки Элмер представлен на рисунке 1.

Рис. 1. Внешний вид черепашки Элмер

Несмотря на достаточно простую конструкцию, черепашка демонстрировала осмысленное и порой очень интересное поведение, базирующееся на трех состояниях: поиск света (голод), поворот к свету, избегание яркого света и препятствий (боль).

При отсутствии источника света черепашка медленно передвигалась, а если сталкивалась с препятствиями, сворачивала в сторону и обходила их. Если появлялся свет, Элмер обязательно замечала его и направлялась к свету (положительный тропизм). Однако, слишком приблизившись, отворачивалась от него, словно боясь ослепления (отрицательный тропизм). Затем двигалась вокруг источника света, находя для себя оптимальные условия существования (гомеостазис). Аккумулятор черепашки постепенно разряжался, и она проявляла все больший интерес к свету, так как его источник обозначал место подзарядки (кормушку). Когда аккумулятор был разряжен настолько, что нуждался в подзарядке, черепашка смело направлялась к источнику света и подключалась к питающим контактам зарядного устройства. Получив так называемую пищу – новый запас электроэнергии, она отходила от зарядного устройства и снова бродила по комнате в поисках оптимальных условий существования.

На рисунке 2 показаны траектории движения черепашек. Изображения получены Уолтером Греем путем фотосъемки с большой выдержкой. Траектория передвижения черепашки хорошо видна, поскольку на ее носу исследователь устанавливал зажженную лампочку.

Наиболее интересна черепашка Кора [2]. Это кибернетическое устройство, кроме зрения и осязания, имело еще и слух, так как было оборудовано микрофоном. Кроме того, его можно было обучать, вырабатывая условный рефлекс.

Как известно, условный рефлекс – результат обучения, привычки. Уолтер Грей учил Кору останавливаться перед препятствием и сворачивать в сторону по звуковому сигналу – свистку. Для этого

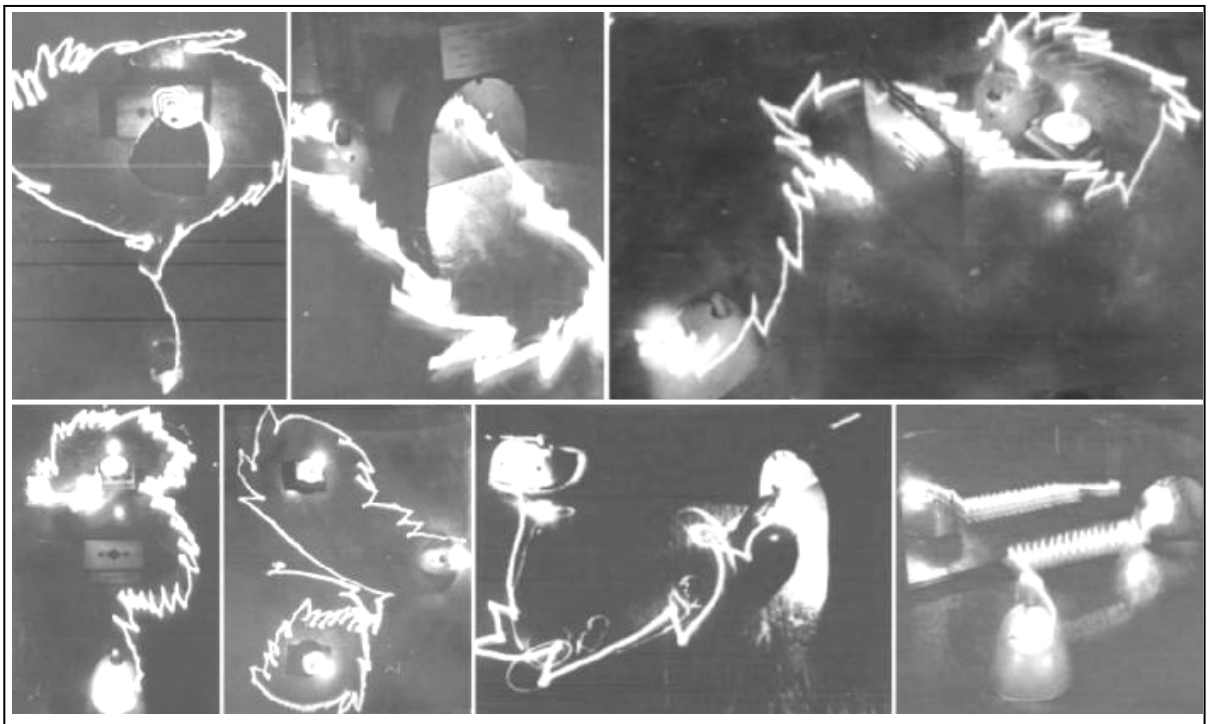


Рис. 2. Траектории движения machina speculatrix Уолтера Грея

он подавал сигнал каждый раз, когда черепашка при своем движении по комнате натыкалась на какое-либо препятствие. Сначала она не обращала внимания на свистки. Однако вскоре у нее вырабатывался условный рефлекс: по сигналу свистка она останавливалась, отступала назад и сворачивала в сторону, даже если перед ней никакого препятствия не было. Но производимый таким образом условный рефлекс вскоре исчезал, если черепашку часто обманывали, подавая сигнал свистка при отсутствии перед ней препятствия.

Поведение, которое демонстрировали робототехнические системы Уолтера Грея, было очень похожим на поведение живых существ, отличительной особенностью которых является именно умение действовать целесообразно, с учетом окружающей обстановки. Взаимодействие между нервной системой его черепашек и окружающей средой создавало неожиданное и сложное поведение. Они никогда точно не повторяли своих действий, но всегда действовали в рамках общего поведенческого образца так, как это делают живые существа.

В дальнейшем устройства, моделирующие поведение живых существ, стали предметом пристального изучения. И в 1989 году Марком Тилденом была создана BEAM (Biology Electronics Aesthetics Mechanics)-технология – новый подход к построению современных робототехнических систем [3]. Именно BEAM-робототехника базируется на рефлексах, которые реализуются на низком аппаратном уровне.

BEAM-концепция заключается в том, что реакция на внешние факторы должна обеспечиваться на первом этапе самой машиной, без участия какого-либо мозга, как это происходило в живой природе, на пути от простейших до человека. По этому же пути должны идти совершенствование и создание более сложных систем, своего рода робогенетика через робобиологию.

Итак, исследования многих лет в управлении техническими системами доказывают эффективность применения с этой целью бионического подхода, которая достигается за счет того, что подобный подход предусматривает использование кибернетической обратной связи Норберта Виннера, а именно, это необходимо для адаптивного управления в постоянно меняющихся условиях неопределенности.

Особенности программной модели machina speculatrix

С целью изучения особенностей моделирования биологических процессов живых организмов, которое осуществил Уолтер Грей при создании своих черепашек, а также применения их при создании других систем управления техническими и автоматизированными объектами с использованием программных средств было принято решение о создании компьютерной имитационной модели биоморфных кибернетических machine speculatrix. Модель представляет собой компьютерную программу, которая имитирует работу черепашек Элмер и Кора в программных кодах. Внешний вид программы представлен на рисунке 3.

Программа обеспечивает следующий набор основных функций:

– создание поля для перемещения черепашек;
– установка количества препятствий на поле;
– установка количества источников света;
– выбор вида черепашки (Элмер или Кора);
– установка количества черепашек, которые будут перемещаться на поле;
– управление источниками света (включение/выключение);
– настройка черепашек (изменение технических параметров);
– мониторинг изменения динамических параметров черепашек (например, заряд аккумулятора);
– фиксирование и отображение траекторий движения черепашек.

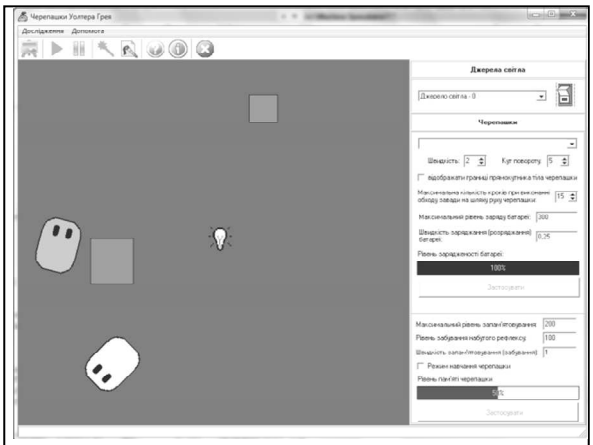


Рис. 3. Общий внешний вид программной имитационной модели machina speculatrix

Для реализации данной имитационной модели была избрана объектно-ориентированная технология программирования, и в связи с этим ее основу составила иерархия классов [4]. На рисунке 4 представлена диаграмма классов для основных компонентов данной имитационной модели, а именно – структура классов, описывающих непосредственно machina speculatrix. Другие классы, которые описывают поле, препятствия, источники света и т.п., опускаются как менее важные.

На вершине находится абстрактный класс TTurtle, реализующий черепашку вообще. В нем воплощены основные функции по отображению черепашки, ее перемещению по полю, поворачиванию в сторону на определенный угол и т.п.

От класса TTurtle наследуются классы TElemerTurtle и TCoraTurtle, которые соответственно реализуют модели черепашек Элмер и Кора.

Класс TElemerTurtle реализует особенности поведения черепашки Элмер в соответствии с описанием, приведенным выше. Модель Элмер также способна реагировать на столкновение с препятствием – сворачивать в сторону и обходить его, способна реагировать на свет, испытывать голод и боль, осуществлять поиск наиболее благоприятных условий и т.п.

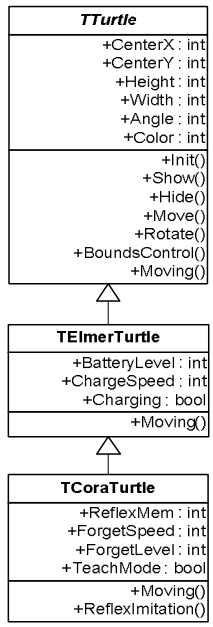


Рис. 4. Диаграмма основных классов имитационной модели machina speculatrix

Класс TCoraTurtle, кроме поведенческих особенностей черепашки Элмер, имеет еще и дополнительные функции по моделированию слуха и выработке условного рефлекса в соответствии с особенностями черепашки Кора. Если включить режим обучения, черепашка при каждом столкновении с препятствием начинает запоминать сигнал – свисток. Когда это происходит достаточное количество раз, у нее вырабатывается условный рефлекс, и тогда, если подать сигнал свистка, Кора будет сворачивать в сторону даже при отсутствии препятствия перед ней.

Исследования, проведенные с моделью, показали, что она полностью адекватно описывает объект моделирования. Виртуальные черепашки Уолтера Грея ведут себя точно так же, как и их реальные технические аналоги. Доказательством этому служит запись траекторий их движения. На рисунке 5 представлены результаты такой записи для одного из сеансов исследования.

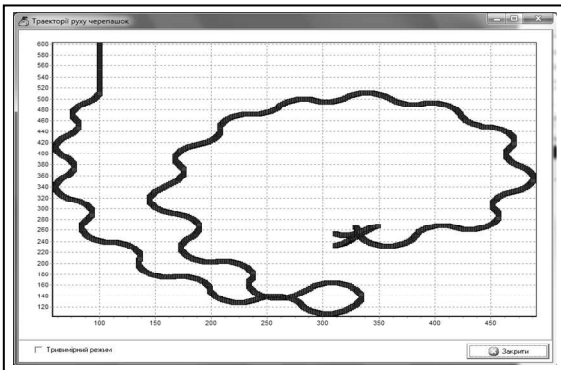


Рис. 5. Траектория движения модели черепашки Уолтера Грея

Если сопоставить изображения на рисунках 2 и 5, то можно легко сделать вывод о том, что аппаратные и программные черепашки Уолтера Грея соответствуют друг другу. Программные черепашки также никогда в точности не повторяют свои действия от сеанса к сеансу, но всегда ведут себя в соответствии с поведенческой моделью. Иными словами, модель черепашек Уолтера Грея воплощает основные принципы бионического подхода к управлению кибернетическими системами.

Таким образом, исследования показали, что созданная модель полностью имитирует поведение черепашек Элмер и Кора, а также позволяет совершенствовать структуру таких бионических кибернетических систем без их технической реализации в железе.

Идеи моделирования рефлексов живых существ нашли свое отражение в разработке рефлекторного подхода для создания интеллектуальных систем [5]. Сегодня он эффективно используется для создания различных автоматизированных систем. Особенно популярен при разработке систем естественно-речевого общения. Последующие исследования предусматривают объединение особенностей модели machina speculatrix Уолтера Грея с рефлекторным подходом для создания интеллектуальных систем. Это позволит усовершенствовать структуру модели таких кибернетических систем и сделать управление ими более эффективным. Новая модель ляжет в основу ее технического аналога.

Литература

1. Крайзмер Л.П., Сочивко В.П. Бионика. М.: Энергия, 1968. 115 с.
2. Owen E. Grey Walter: The Pioneer of Real Artificial Life; Proc. of the 5th International Workshop on Artificial Life. MIT Press, Cambridge, 1997, pp. 34–44.
3. Smit Michael C. Beam Robotics / Smit Michael C., Tilden Mark W.; Algorithm, 1991. Vol. 2. №. 2, pp. 15–19.
4. Кендал С. UML. Основные концепции. М.: Изд. дом «Вильямс», 2002. 144 с.
5. Тесля Ю.М. Несиловое взаимодействие: монография. Киев: Кондор, 2005. 195 с.

УДК 004.942

КОМПАКТНАЯ МОДЕЛЬ ГРАФЕНОВОГО ПОЛЕВОГО ТРАНЗИСТОРА НА ЯЗЫКЕ VERILOG-A

А.А. Целыковский; И.А. Данилов
(Научно-исследовательский институт системных исследований РАН, atsel@niisi.msk.ru, danilov@niisi.msk.ru);
Г.И. Зебрев, д.т.н. (Национальный исследовательский ядерный университет «МИФИ», gizebrev@mephi.ru)

Представлены компактная модель графенового полевого транзистора и ее реализация на языке Verilog-A. На примере моделирования ряда аналоговых схем на основе графеновых транзисторов продемонстрирована возможность использования модели в промышленных САПР.

Ключевые слова: *графеновый полевой транзистор, компактная модель, моделирование, ВАХ, Verilog-A, САПР, двухполупериодное выпрямление тока, умножение частоты, двухпозиционная фазовая манипуляция, амбиполярная электроника.*

Графен, представляющий собой моноатомный слой углерода, рассматривается как возможная альтернатива кремнию для будущей наноэлектроники благодаря высокой подвижности носителей заряда и хорошим перспективам геометрического масштабирования [1]. Отсутствие запрещенной зоны, приводящее к большим токам утечки транзисторов в закрытом состоянии, препятствует применению графена в цифровых схемах, но не является ограничением для его использования в СВЧ-электронике [2].

Недавно были продемонстрированы графеновые транзисторы, частоты отсечки которых выше, чем у лучших кремниевых транзисторов с соотносимыми длинами затворов [1]. Продолжающееся совершенствование технологий получения графена на большой площади делает реальной перспективу создания графеновых интегральных схем. В связи с этим встает задача схемотехнического проектирования с использованием графеновых транзисторов. Для этого требуется создание простых аналитических компактных моделей, подобных существующим для кремниевых транзисторов. Примером последних являются модели типа BSIM, являющиеся промышленным стандартом. Такие модели хорошо подходят для быстрых компьютерных расчетов и интеграции в существующие САПР, которые имеют набор стандартных инструментов моделирования, например, переходных процессов, облегчающих разработку и верификацию компактной модели. При этом компактные модели по возможности отражают процессы на физическом, феноменологическом или эмпирическом уровнях.

В последнее десятилетие в качестве одного из основных средств создания компактных моделей стал использоваться язык Verilog-A, который предназначен для описания аналоговой аппаратуры и позволяет реализовывать модели объектов на разных уровнях абстракции. Одним из главных преимуществ Verilog-A с точки зрения разработки компактных моделей является его широкое распространение как составной части языка Verilog-AMS, являющегося расширением обычного Verilog на случай аналого-цифровой аппаратуры. Это обусловливает наличие средств, необходимых для работы с Verilog-A, в большинстве современных САПР.

Помимо широкого распространения, к достоинствам языка Verilog-A следует отнести простоту его использования и интеграции написанных на нем моделей в программы моделирования типа SPICE. Кроме того, данный язык позволяет работать с частными производными в символьном виде. Все это делает Verilog-A крайне эффективным при написании компактных моделей.

Предлагаемая в данной работе диффузионно-дрейфовая компактная модель графенового полевого транзистора была реализована на языке Verilog-A. Модель основана на явном аналитическом решении уравнения непрерывности тока в канале и позволяет непрерывно описывать в аналитической форме вольт-амперные характеристики транзистора во всех режимах работы. Следует отметить, что традиционно существующие модели, например Level-3, пренебрегают решением уравнения непрерывности тока в канале, что приводит к нефизичной кусочной форме описания вольт-амперных характеристик отдельно для линейного режима и режима насыщения тока, а также для подпорогового и надпорогового участков. Представленная авторами модель учитывает физические особенности графеновых транзисторов (специфическую электростатику, важную роль квантовой емкости и т.д.) и экспериментально наблюдаемые специфические эффекты.

В работе рассматривается графеновый полевой транзистор с двумя затворами, схематическое изображение которого дано на рисунке 1. На рисунке 2 показана эквивалентная емкостная схема данной полевой структуры. Модель такого транзистора можно свести к модели [3] однозатворного транзистора с эффективным напряжением на затворе:

$$V_G = \frac{C_1 V_1 + C_2 V_2}{C_1 + C_2} \qquad (1)$$

и эффективной удельной емкостью подзатворного диэлектрика $C_{ox}=C_1+C_2$ (рис. 3). Здесь $V_{1(2)}$ – напряжение на верхнем (нижнем) затворе, $C_{1(2)}=\varepsilon_{1(2)}\varepsilon_0/d_{1(2)}$ – удельная емкость верхнего (нижнего) подзатворного диэлектрика с диэлектрической проницаемостью $\varepsilon_{1(2)}$ и толщиной $d_{1(2)}$. При малой емкости нижнего диэлектрика и заземлении нижнего затвора (V_2=0) модель транзистора с двумя затворами превращается в модель однозатворного транзистора с $V_G=V_1$ и $C_{ox}=C_1$.

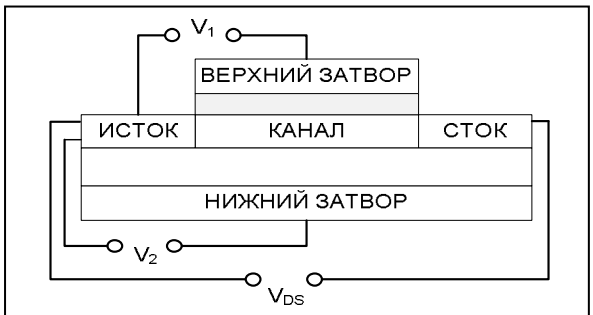


Рис. 1. Схематическое изображение графенового полевого транзистора с двумя затворами

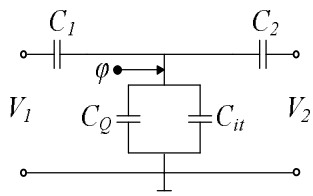


Рис. 2. Эквивалентная емкостная схема графенового полевого транзистора с двумя затворами: C_Q – удельная квантовая емкость, C_{it} – удельная емкость поверхностных состояний на границе раздела графена и диэлектрика

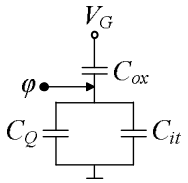
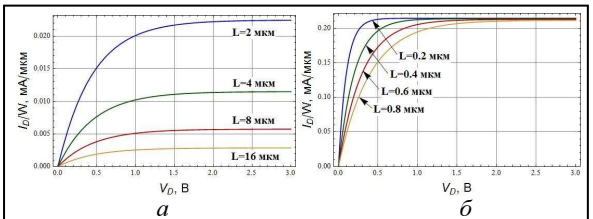


Рис. 3. Эквивалентная емкостная схема графенового транзистора с одним затвором

Параметрами предлагаемой модели являются геометрические размеры канала транзистора (длина L и ширина W), характеристики подзатворных диэлектриков (их диэлектрические проницаемости и толщины), удельная емкость поверхностных состояний C_{it}, соответствующее электронейтральности в канале напряжение V_{NP}, подвижность носителей заряда в графене μ_0. При расчетах принимается во внимание существенная роль квантовой емкости в графеновых полевых структурах. В модели учитываются два типа насыщения тока в канале графенового транзистора: электростатическое запирание канала, при котором ток насыщения обратно пропорционален длине канала (рис. 4а), и насыщение дрейфовой скорости на значении v_{opt} (также являющемся параметром модели), при котором ток насыщения не зависит от длины канала (рис. 4б).



а *б*

Рис. 4. Выходные ВАХ графенового полевого транзистора; эффект генерации носителей подавлен

За счет учета размножения носителей заряда при сильных электрических полях в канале [4] модель позволяет описывать ряд экспериментально наблюдаемых эффектов, специфичных для графеновых полевых транзисторов [5], среди которых излом и быстрый рост тока на выходных *вольт-амперных характеристиках* (ВАХ) графенового транзистора после участка насыщения при увеличении напряжения на стоке (рис. 5).

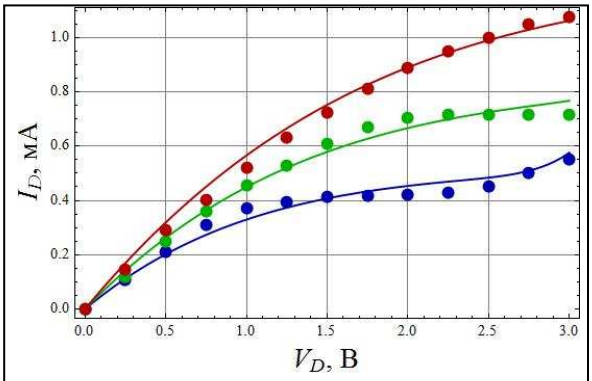


Рис. 5. Сравнение результатов моделирования тока стока (линии) с экспериментом (точки) при различных напряжениях на затворе; моделирование проведено с учетом сопротивления контактов величиной 300 Ом в цепи сток–исток

Компактная модель реализована на языке Verilog-A в виде отдельного модуля, имеющего выводы, соответствующие электродам *D* (сток), *G*1 (верхний затвор), *G*2 (нижний затвор), *S* (исток) транзистора. В модуле были объявлены константы, отсутствующие в стандартной библиотеке физических констант для языка Verilog-A, поставляемой в составе пакета Cadence IC Design (например, скорость Ферми и постоянная тонкой структуры в графене). Также были объявлены параметры модели, среди которых геометрические размеры канала, толщины и диэлектрические проницаемости подзатворных диэлектриков и т.д. Для параметров были указаны значения по умолчанию, которые используются, если в программе-симуляторе они дополнительно не определены. Вычисление величин, не меняющихся в процессе моделирования (зависящих только от параметров), помещены в блок вычислений, выполняемых один раз в начале моделирования (*@initial_step*). Остальные вычисления выполняются на каждом шаге моделирования. Напряжения на выводах модуля обозначаются как *V*(*D*), *V*(*G*1), *V*(*G*2), *V*(*S*), ток от истока к стоку – как *I*(*D*, *S*). Задание электрических величин на выводах модуля производится при помощи специального оператора <+. Описанное проиллюстрируем листингом.

Листинг. Элементы компактной модели графенового транзистора на языке Verilog-A:

```
`include "constants.vams"
`include "disciplines.vams"
module GFET(D, G1, G2, S);
inout D, G1, G2, S;
electrical D, G1, G2, S;
...
`define v_F 1M
...
parameter real L = 1u;
...
real Id0;
...
analog
        begin
                @(initial_step)
                    begin
                                ...
                                m=1+Cit/Cox;
                                ...
                    end
                ...
                Id0=W*`P_Q*ns*vsat*(1-exp(-u0*(V(D)-
                    -V(S))/(vsat*L)));
                I(D,S) <+ Id0/(1-M);
    end
```

Реализация модели на языке Verilog-A обеспечила возможность ее использования в стандартных SPICE-подобных симуляторах.

На рисунке 6 представлена электрическая схема, состоящая из двух инверторов на резисторах и графеновых транзисторах. Она была промоделирована в симуляторе Cadence Spectre. Для резистора использована идеальная модель, для графенового транзистора – модель, представленная в настоящей работе. Результаты моделирования отражены на рисунке 7.

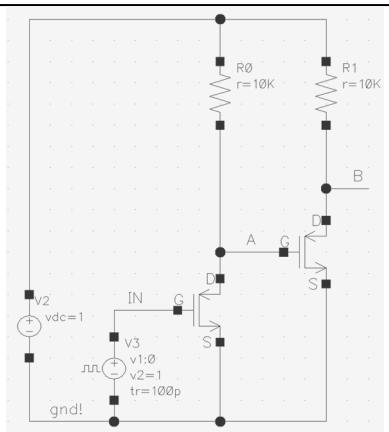


Рис. 6. Схема с двумя инверторами на графеновых транзисторах

Важнейшей особенностью графеновых полевых транзисторов является их амбиполярность, то есть способность изменять тип проводимости в зависимости от смещения на затворе. Представленная модель описывает амбиполярный характер проводимости графенового транзистора, который отражается в его передаточной характеристике (рис. 8), симметричной и близкой к передаточной характеристике выпрямителя [6].

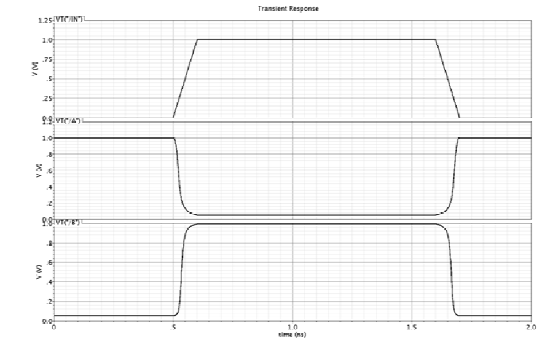


Рис. 7. Результаты моделирования схемы, представленной на рисунке 6. Сверху вниз: напряжение на входе, выходе первого инвертора, выходе второго инвертора

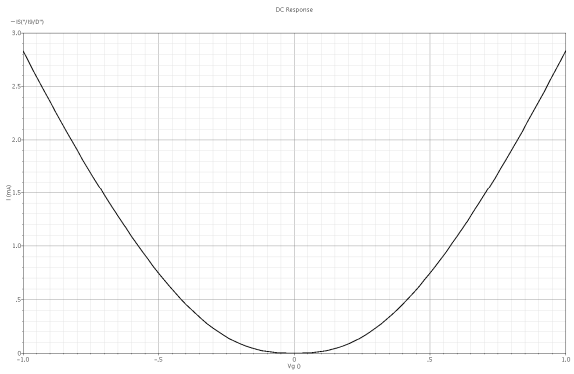


Рис. 8. Передаточная характеристика графенового полевого транзистора (моделирование в Cadence Spectre). Левая ветвь соответствует проводимости p-типа, правая – n-типа

Устройства амбиполярной электроники на основе графена, среди которых двухполупериодный выпрямитель тока и умножитель частоты [6], схема двухпозиционной фазовой манипуляции [2] и другие, характеризуются значительным упрощением схемотехники из-за сокращения числа используемых транзисторов по сравнению с кремниевыми аналогами. На рисунке 9 представлена моделируемая схема, которая в зависимости от типа подаваемых на затвор сигналов может играть роль как выпрямителя тока и удвоителя частоты, так и схемы двухпозиционной фазовой манипуляции.

При подаче на затвор транзистора только синусоидального сигнала (если напряжение электронейтральности V_{NP} отлично от нуля, следует также сместить канал транзистора в состояние электронейтральности постоянным напряжением V_{NP}, которое удобно подавать на нижний затвор) схема на рисунке 9 представляет собой двухполупериодный выпрямитель тока и умножитель частоты со значительно более простой схемотехникой [6] по сравнению с традиционными устройствами. Напряжения на входе (затворе транзистора) и выходе (стоке транзистора) данной схемы, а также ток стока транзистора представлены на рисунке 10.

При подаче на затворы транзистора двух управляющих сигналов – сигнала прямоугольной формы, который, меняя полярность, определяет тип проводимости в канале, и синусоидального сигнала меньшей амплитуды – моделируемая схема играет роль схемы двухпозиционной фазовой манипуляции (рис. 11).

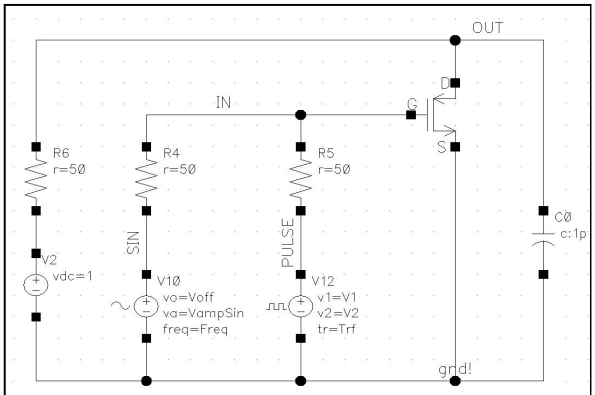


Рис. 9. Включение графенового транзистора при моделировании двухполупериодного выпрямителя тока, умножителя частоты и схемы двухпозиционной фазовой манипуляции

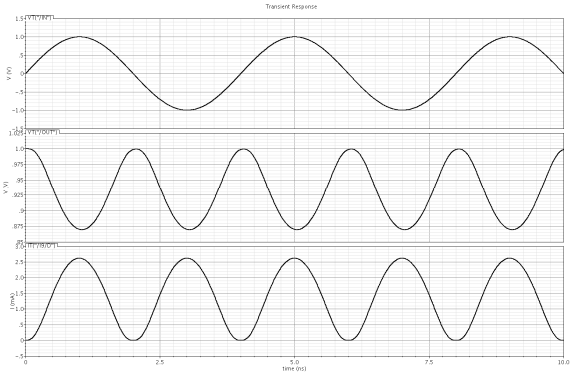


Рис. 10. Двухполупериодное выпрямление тока и умножение частоты схемой на основе графенового транзистора (сверху вниз: напряжение на входе схемы, напряжение на выходе схемы, ток стока транзистора)

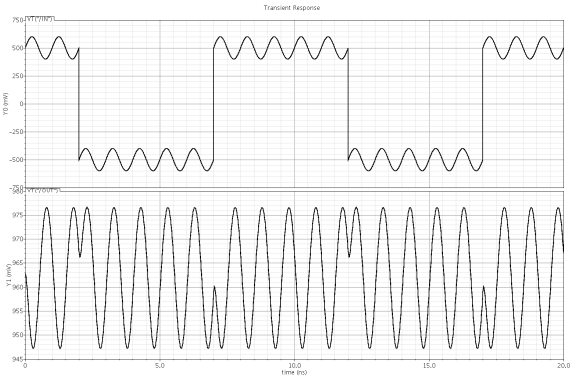


Рис. 11. Входной (вверху) и выходной сигналы схемы двухпозиционной фазовой манипуляции на основе графенового транзистора

Амбиполярная электроника является одним из активно развивающихся направлений электроники, открывающих новые перспективы для построения привычных устройств электроники. Представленную в данной работе компактную модель на языке Verilog-A можно использовать в промышленных САПР для моделирования как статических, так и динамических характеристик схем на основе амбиполярных графеновых транзисторов.

Литература

1. Schwierz F. Graphene transistors. Nature Nanotech. 2010, № 5, pp. 487–496.

2. Hsu A. [et al.]. High Frequency Performance of Graphene Transistors Grown by Chemical Vapor Deposition for Mixed Signal Applications. Japanese Journal of Applied Physics, 2011. Vol. 50.

3. Zebrev G.I. Graphene Field Effect Transistors: Diffusion-Drift Theory, in Physics and Applications of Graphene – Theory, InTech, 2011. URL: http://www.intechopen.com/articles/show/title/graphene-field-effect-transistors-diffusion-drift-theory (дата обращения: 14.10.2011).

4. Целыковский А.А. Генерация носителей заряда в канале графенового транзистора при сильных тянущих электрических полях // Ломоносов-2011: Сб. тез. Междунар. конф.: в 2 т. М.: МГУ. 2011. Т. 2. С. 90–92.

5. Meric I. [et al.] Current saturation in zero-bandgap, top-gated graphene field- effect transistors, Nature Nanotech., 2008. № 3, pp. 654–659.

6. Wang H. [et al.]. Graphene frequency multipliers, IEEE Electron Device Letters. 2009. Vol. 30. № 5, pp. 547–549.

УДК 004.32:550.3

ОБЗОР СИСТЕМ СБОРА И ПЕРВИЧНОЙ ОБРАБОТКИ ИНФОРМАЦИИ ДЛЯ СТАНЦИЙ ГЕОЛОГО-ТЕХНОЛОГИЧЕСКИХ ИССЛЕДОВАНИЙ

С.А. Калёнов (Институт кибернетики, информатики и связи Тюменского государственного нефтегазового университета, polinochcka86@rambler.ru)

Рассматриваются распространенные системы сбора информации, применяемые на станциях геолого-технологических исследований СИРИУС и «Геосфера» для сопровождения бурения скважин. Сравниваются технические характеристики, надежность и удобство эксплуатации систем.

Ключевые слова: *станции СИРИУС, «Разрез-2», «Геосфера», геолого-технологические исследования, устройство сбора информации.*

Информационное обеспечение процесса бурения нефтяных и газовых скважин является наиболее важным звеном на этапе строительства скважин, особенно при введении в разработку и освоении новых нефтегазовых месторождений.

В информационном обеспечении процесса строительства скважин наиболее важную роль играют *геолого-технологические исследования* (ГТИ). Основными задачами службы ГТИ являются изучение геологического строения разреза скважин, выявление и оценка продуктивных пластов и повышение качества строительства скважин на основе получаемой в ходе бурения геолого-геохимической, геофизической и технологической информации. Оперативная информация, находящаяся в службе ГТИ, имеет большое значение при бурении разведочных скважин в малоизученных регионах со сложными горно-геологическими условиями, а также при проводке наклонно направленных и горизонтальных скважин [1].

Для решения этих задач разработаны системы непрерывного сбора, обработки, контроля, передачи и хранения информации, находящиеся на станциях ГТИ СИРИУС и «Геосфера» и на буровой и сопровождающие весь процесс бурения скважины.

Компьютеризированная станция ГТИ нефтегазовых скважин СИРИУС включает в себя систему сбора данных, состоящую из блока управления БУ-201-06 и двух распределительных блоков БР1-206 и БР2-207. Блок управления является центральным блоком системы сбора данных ГТИ и предназначен для обеспечения питания сети цифровых датчиков, расположенных на буровой, а также для опроса датчиков, сбора информации об измеренных параметрах и передачи их в ПЭВМ. Распределительные блоки предназначены для объединения терминальных устройств (датчиков) и компьютерной станции в единую систему сбора данных ГТИ (далее система сбора) для контроля, регулирования и управления технологическими процессами. Они являются связующим звеном в распределенной системе сбора, к которому подключаются терминальные устройства, формируют требуемые информационные каналы и позволяют выполнять следующие функции:

– прием и первичную обработку информации от аналоговых и цифровых терминальных устройств (датчиков);

– управление цифровыми и аналоговыми датчиками;

– обеспечение электропитания терминальных устройств и их гальванической развязки по питающим и информационным линиям;

– обеспечение связи по кабельной линии с оборудованием верхнего уровня системы сбора.

Компьютеризированная станция ГТИ нефтегазовых скважин «Геосфера» в своем составе может иметь одну из трех предлагаемых производителем систем сбора:

– система сбора технологической информации ТВР 60-8А;

– выносная система сбора технологической информации ВССИ;

– устройство сбора технологической информации ССИ-3.

ТВР 60-8А используется в станциях ГТИ скважин и предназначена для

– аналого-цифрового преобразования изменяющихся электрических величин по шестидесяти аналоговым каналам;

– измерения углового перемещения двухфазного датчика положения талевого блока и преобразования полученного значения в ток или напряжение;

– измерения частоты по восьми цифровым каналам (имп./с) с последующим преобразованием полученных значений в ток или напряжение;

– определения состояний по восьми каналам;

– включения-выключения нормально разомкнутых сухих контактов по восьми каналам;

– выдачи питания для датчиков или барьеров искробезопасности (=24 В);

– цифровой передачи измеренной информации в технологический компьютер по интерфейсу RS422/485 в соответствии с протоколом обмена;

– цифровой передачи информации из технологического компьютера по интерфейсу RS-422/485 и выдачи питания (+24 В) на табло бурильщика.

Выносная система сбора ВССИ производит сбор информации от технологических датчиков, ее преобразование, предварительную обработку, фильтрацию, а также обеспечивает индивидуальным стабилизированным питанием с защитой от короткого замыкания и визуализацией подключенных к ней технологических датчиков.

Выносная система сбора ВССИ используется в станциях ГТИ скважин и предназначена для

– аналого-цифрового преобразования изменяющихся электрических величин напряжения и тока;

– счета импульсов в соответствии с направлением счета;

– измерения частоты;

– цифровой передачи информации на станцию ГТИ;

– обеспечения индивидуальным стабилизированным питанием технологических датчиков.

ВССИ устанавливается на буровой в месте, максимально приближенном к объектам измерения, и соединяется со станцией ГТИ четырехжильным кабелем. Изделие выполнено в стальном пылевлагозащищенном корпусе с классом защиты IP65.

Устройство сбора технологической информации ССИ-3 используется в станциях ГТИ скважин с целью

– аналого-цифрового преобразования изменяющихся электрических величин напряжения и тока;

– счета импульсов в соответствии с направлением счета;

– измерения частоты;

– выдачи питающего напряжения для датчиков.

Система сбора ССИ-3 представляет собой измерительную плату и источник питания датчиков, установленные в общем корпусе, и включает клеммную панель для подключения магистрального кабеля от датчиков. Подключение системы сбора ССИ-3 к компьютеру производится через интерфейс USB1.1 или RS-232.

По своим техническим параметрам рассматриваемые системы сбора информации ориентированы на различные по сложности задачи контроля процесса бурения и характеризуются разными способами и оперативностью монтажа [2].

Сравнивая технические характеристики систем сбора, стоит отметить, что при необходимости контроля большого числа параметров предпочтительной является ТВР 60-8А, имеющая 60 каналов для подключения аналоговых датчиков и дающая возможность автоматизировать процесс с помощью как дискретных, так и аналоговых управляющих сигналов. Это выделяет систему, позволяя использовать ее в качестве *устройства сопряжения с объектом* (УСО) для частичной автоматизации технологических процессов бурения при условии применения совместимого с ней оборудования. Недостатком является необходимость прокладки всех кабелей от датчиков до системы сбора в одну точку на буровой, что требует больших затрат времени и сил и снижает оперативность развертывания станции на буровой установке.

Системы сбора ССИ-3 и ВССИ станции ГТИ «Геосфера» не имеют возможностей автоматизации процессов и обладают минимально необходимым количеством контролируемых параметров, позволяющих контролировать процесс бурения промысловых скважин без каких-либо затруднений и отличающихся лишь способом монтажа. Стоит отметить, что подключение датчиков к системе ССИ-3 производится по магистральному кабелю, упрощая монтаж датчиков на буровой, но при этом увеличивая шанс отказа части (либо всего комплекса) датчиков в случае разрыва одного из соединений [3].

Система сбора станции ГТИ СИРИУС в отличие от рассмотренных имеет распределенную структуру. Два распределительных блока монтируются на буровой, что позволяет более оперативно проводить монтаж датчиков. Существенным фактором является возможность подключения цифровых датчиков и оперативной адаптации количества цифровых и аналоговых измерительных линий путем подбора плат обработки сигналов распределительных блоков. На распределительном блоке БР1-206 возможно подключение до восемнадцати аналоговых датчиков с информационными сигналами тока или напряжения, на БР2-207 – до шестнадцати, что превосходит возможности ВССИ и ССИ-3 (по пятнадцать каналов измерения напряжения и по пять тока) [4].

Кроме того, данная система имеет повышенную отказоустойчивость за счет возможности частичной замены функций блока БР2-207 блоком БР1-206. При этом блок управления находится в станции с более благоприятными условиями работы и из строя выходит крайне редко.

Системы сбора станции ГТИ «Геосфера» измеряют постоянные пороговые значения (ТВР 60-8А: +2,5 В, +1,25 В, +625 мВ, +312 мВ, +156 мВ, +78 мВ, +39 мВ, +20 мВ; ВССИ и ССИ-3: +10 В, +5 В, +312 мВ, +78 мВ), в то время как распределительные блоки системы сбора станции ГТИ СИРИУС способны измерять изменяющийся информационный сигнал напряжения в пределах от 0 до 10 В, позволяя получить большую информативность от канала связи с датчиком. По токовому сигналу принципиальных отличий между системами нет. Все системы сбора включают в себя защиту от перенапряжений и смены полярности входного напряжения, что исключает повреждение электронных компонентов в случае неправильной эксплуатации либо выхода из строя подключаемого оборудования.

Немаловажное значение имеет интерфейс взаимодействия системы сбора с компьютером, так как станции ГТИ комплектуются технологическими компьютерами и ноутбуками промышленного производства, которые не всегда обеспечивают аппаратную поддержку для обмена данных. На решение этой проблемы в полевых условиях могут потребоваться дополнительные затраты времени и средств, обусловленные необходимостью доставки и установки устройств либо плат расширения. К тому же операторы станций не всегда могут корректно установить дополнительное оборудование без помощи специалиста.

Системы сбора станции ГТИ «Геосфера» ССИ-3 и блок управления системы сбора станции ГТИ СИРИУС БУ-201-06 лишены данного недостатка, поскольку ССИ-3 взаимодействует с компьютером с помощью широко распространенного интерфейса USB1.1 либо RS-232, а БУ-201-06 – только посредствам RS-232. Подключение системы сбора ТВР 60-8А требует наличия интерфейса RS-422 либо RS-485 (как и ВССИ).

Станции ГТИ СИРИУС («Разрез-2») и «Геосфера» получили широкое распространение в России и за рубежом благодаря широким возможностям оперативных ГТИ, мобильности и высокой надежности и точности систем сбора информации, адаптируемых под задачи различной сложности.

Литература

1. Акбулатов Т.О., Акчурин Х.И. Информационное обеспечение процесса бурения: учеб. пособие. Уфа: Изд-во УГНТУ, 2002. 55 с.

2. Булатов А.И., Демихов В.И., Макаренко П.П. Контроль процессов бурения нефтяных и газовых скважин. М.: Изд-во «Недра», 1998. 345 с.

3. URL: http://geosferatver.ru/unit/unit.php (дата обращения: 16.07.2011).

4. URL: http://gelstver.ru (дата обращения: 16.07.2011).

УДК 519.767

РАЗРАБОТКА ИНСТРУМЕНТОВ ВЕРИФИКАЦИИ ДРАЙВЕРОВ НА ОСНОВЕ СЕМАНТИЧЕСКИХ МОДЕЛЕЙ

Ю.П. Кораблин, д.т.н.; Е.Г. Павлов
(Российский государственный социальный университет, г. Москва, y.p.k@mail.ru, lucenticus@gmail.com)

Проводится обзор основных инструментов, применяемых для верификации драйверов. Указаны преимущества и недостатки, а также структурные особенности каждого верификатора. Описан разработанный метод верификации драйверов на основе семантических моделей. Приводится пример верификации драйвера Linux.

Ключевые слова: *верификация, драйверы устройств, процессная семантика, эквациональная характеристика, алгебраическая семантика.*

Современные *операционные системы* (ОС) поддерживают множество типов устройств, изготовленных различными производителями. Эти устройства могут иметь специфичные особенно-

сти, которые известны лишь самим компаниям. Поэтому разработкой драйверов для ОС, за исключением, может быть, стандартных, занимается, как правило, сам производитель устройства. Однако во всех современных системах драйвер является частью ОС и может иметь доступ к привилегированному режиму (режиму ядра). Ошибка в драйвере может привести к краху всей системы. Кроме того, в некоторых случаях драйвер одного устройства вызывает сбой в драйвере другого устройства, так как довольно сложно протестировать взаимодействие между специфичными устройствами. Именно поэтому разработка драйверов относится к одной из самых сложных задач в программировании.

Драйверы являются частью ядра по вполне обоснованным причинам, так как большинство из них используют низкоуровневые команды и системные вызовы ядра ОС. Однако, если в драйвере присутствует ошибка (например, разыменование указателя на NULL), это может разрушить всю систему. Именно в драйверах кроется основная причина нестабильности ОС. Как показывают исследования, около 80 % сбоев в Microsoft Windows XP происходят из-за ошибок в драйверах сторонних производителей. Чтобы обезопасить свои системы, Microsoft ввела цифровую подпись WHQL (Windows Hardware Quality Laboratory): драйвер отправляется в тестовую лабораторию Microsoft, и, если он удовлетворяет требованиям надежности и безопасности, ему выдается уникальная цифровая подпись. Помимо этого, в современных драйверных системах предлагается использовать драйверы режима пользователя там, где нет необходимости в низкоуровневом взаимодействии с устройством. Примером такой модели может служить Windows Driver Foundation.

В последние годы активно исследуется повышение надежности ОС. Работы ведутся в самых различных направлениях – от проектирования ОС, устойчивых к сбоям, до автоматической генерации кода драйверов. Однако сегодня практическое применение нашел лишь один подход – верификация драйверов устройств. На основе данного подхода созданы инструментальные пакеты для статического анализа кода драйверов. Эти пакеты разработаны для наиболее популярных ОС семейств Microsoft Windows и Linux.

Инструменты для верификации драйверов ОС

Для верификации драйверов ОС семейства Windows компания Microsoft разработала программный пакет Static Driver Verifier, поставляемый с 2006 года как часть Microsoft Windows Driver Development Kit. Данный верификатор основан на технологии SLAM, разработанной в Microsoft Research [1].

Инструмент для статического анализа Static Driver Verifier предназначен для автоматической проверки во время компиляции кода драйвера Windows, написанного на языке C, с целью выявления нарушений правил драйверной модели.

Static Driver Verifier исследует ветви исполнения кода драйвера устройства, символически исполняя его исходный код. Static Driver Verifier помещает драйвер в агрессивную среду и систематически проверяет все ветви исполнения кода, выискивая нарушения правил использования драйверной модели. Для проверки драйвера Static Driver Verifier выполняет компиляцию, компоновку и сборку драйвера, сканирует исходный код в поисках точек входа, проводит подготовку к верификации, после чего проверяет, следует ли драйвер правилам текущего сеанса верификации.

Для выполнения верификации Static Driver Verifier предоставляет свою модель ОС. Движок верификации всесторонне анализирует ветви исполнения кода и пытается доказать, что драйвер нарушает правила, некорректно взаимодействуя с моделью ОС, предоставленной Static Driver Verifier.

При выполнении проверки драйвера Static Driver Verifier объединяет его исходный код с моделью ОС, получая таким образом своеобразный бутерброд из кода, верхний слой которого представляет собой управляющую оснастку, средний – исходный код драйвера, нижний содержит заглушки вместо фактических функций драйверной модели.

Static Driver Verifier вводит этот комбинированный код на языке C в верификационную машину SLAM. Она оснащает его правилами, выбранными при запуске процесса верификации, и выполняет всестороннюю проверку. По результатам проведенного анализа генерируются статистические данные о запуске и детальные трассы ошибок, для анализа которых разработаны специализированные графические инструменты.

В работе [2] авторы провели портирование техники верификации драйверов Windows на ОС Linux. Основываясь на технологии Microsoft, реализованной в проекте Static Driver Verifier, расширен язык спецификаций SLIC и вместо SLAM используется верификатор на основе проверки моделей CBMC. На базе документации ядра были сформулированы некоторые правила для драйверов Linux и реализованы на языке SLICx. Таким образом, эта работа позволила использовать стиль Static Driver Verifier для верификации драйверов Linux. Данный инструментарий реализован в виде плагина для среды разработки Eclipse. Следуя парадигме аспектно-ориентированного программирования, правила SLICx внедряются в исходный код драйвера, а далее модифицированные исходники проверяются во время компиляции верификатором CBMC.

Иной подход реализован в предикативном верификаторе DDVerify [3]. Он является полностью автоматизированным инструментом проверки кода, который, учитывая исходный код Linux-драйвера устройства, генерирует соответствующую обертку для драйвера и с использованием SATABS проверяет место нарушения драйвером пред- и постусловия модели ядра. Правила корректности задаются как часть модели ядра. Код ограничений, накладываемых правилами, задается вместе с кодом, описывающим семантику функции. В данном подходе можно проверять только те функции, которые привязываются к драйверу с помощью линковки. Поэтому для использования DDVerify требуется существенно изменять заголовочные файлы ядра.

Исследования в области верификации драйверов активно ведутся и в России. В 2009 году на базе Института системного программирования РАН (г. Москва) создан Центр верификации ОС Linux, в основе которой лежит инструментарий Linux Driver Verification [4]. Процесс верификации драйвера в этом инструментарии начинается с инициализации системы LDV-Core. Перед началом работы при необходимости на основе архива с исходным кодом ядра Linux создается копия этого ядра со специально модифицированной подсистемой сборки. Далее LDV-Core запускает процесс компиляции драйверов и одновременно с этим на основе модификаций сборки читается поток команд компиляции и линковки с выделением из них тех, которые относятся к верифицируемым драйверам. Затем для каждого типа драйвера создается одна или несколько моделей окружения и добавляется код моделей и проверок указанных правил корректности к соответствующим файлам драйвера. Далее код отправляется на вход верификатору (BLAST, CPAchecker и другие), который уже не имеет представления о том, что поданный ему на вход код является драйвером ядра ОС Linux и может использоваться в неизменном виде для верификации достижимости программ на языке программирования С.

На данный момент в проекте Linux Driver Verification используются только верификаторы достижимости, то есть инструменты, предназначенные для выявления нарушений правил корректности.

После всех проверок инструмент верификации выносит вердикт, который может принимать одно из трех значений: SAFE, UNSAFE и UNKNOWN. По полученному вердикту все описанные выше стадии проходят в обратном порядке. По ходу этого процесса вердикт дополняется отчетами о работе других компонентов, после чего формируется финальный отчет о проверке всего задания.

Для удобства использования Linux Driver Verification было разработано несколько пользовательских интерфейсов (командной строки, веб-интерфейс), предназначенных для проведения верификации и последующего анализа полученных результатов по некоторому набору ядер и правил.

Верификация на основе семантических моделей

В работе [5] для доказательства свойств программ предлагается использовать семантические модели. Семантика языков распределенного программирования исследуется посредством сопоставления программам множества вычислительных последовательностей и анализа семантических значений в заданной алгебраической модели вычислительных последовательностей (под вычислительной последовательностью понимается последовательность выполненных команд и тестов программы).

Методика исследования семантики программ разработана для модели языка распределенного программирования L и расширена для языка *асинхронных функциональных схем* (АФС). Перед анализом свойств программы сначала необходимо преобразовать ее в соответствующую программу на языке L или АФС.

Рассмотрим более подробно язык АФС. Пусть заданными являются следующие синтаксические области, то есть множества:

Com – элементарных команд с типичным элементом *a*;

Exp – булевских выражений с типичным элементом *b*;

BConst – булевских констант с константами *tt* (тождественно истинное значение) и *ff* (тождественно ложное значение);

RCom – команд ввода с типичным элементом *read*;

WCom – команд вывода с типичным элементом *write*;

CPLab – меток каналов и процессов с типичными элементами *i* и *j*;

CNom – номеров входов/выходов каналов с типичными элементами *k* и *l*;

Cmd – команд с типичным элементом *c*;

CWait – команд ожидания с типичным элементом *wait*;

TExp – временных выражений с типичным элементом *time*.

Множество АФС-программ *Prog* с типичным элементом *pr* определяется как

pr ::= *NET* {*can*} *BEGIN fproc END*

can ::= *CHAN j*::*type*(*k*):*type*(*l*) | can_1 ; can_2

type ::= *ALL* | *ANY*

fproc ::= *FUN i*::*c* | $fproc_1$; $fproc_2$

c ::= *a* | *skip* | *exit* | *break* | *wait*(*time*) | *read*(*i*, *l*) | *write*(*i*, *k*) | *SEQ*(*c* {, *c*}) |

PAR(*c* {, *c*}) | *ALT*(*gc*) | *LOOP*(*ALT*(*gc*))

gc ::= $g \rightarrow c$ | *gc* {; *gc*}

g ::= *tt* | *ff* | *b* | *wait*(*time*) | *read*(*i*, *l*) | *write*(*i*, *k*)

Неформально семантика АФС-программ задается следующим образом:

– под элементарной командой *a* понимается обычное присваивание;

– пустая команда *skip* не приводит к выполнению каких-либо действий;

– выполнение команды *exit* приводит к завершению функционального процесса;

– выполнение команды *break* приводит к завершению выполнения цикла;

– команда *wait*(*time*) задерживает выполнение функционального процесса на время, задаваемое выражением *time*;

– команда чтения *read*(*i*, *l*) осуществляет запрос на прием данных из *l*-го выхода *i*-го канала, а команда *write*(*i*, *k*) – на передачу данных на *k*-й вход *i*-го канала; если *l*-й выход (*k*-вход) канала *i* готов к выдаче (приему) данных, осуществляется чтение данных из канала (запись данных в канал), после чего продолжается выполнение функционального процесса; в противном случае его выполнение задерживается;

– команда *SEQ*(*c*{, *c*}) означает последовательное выполнение команд, перечисленных внутри скобок;

– параллельная команда *PAR*(*c*{, *c*}) означает параллельное выполнение команд, перечисленных внутри скобок;

– защищенная команда $g \to c$ может выполняться лишь в случае истинности защиты *g*; значение защиты *g* является истинным в следующих случаях: 1) $g \equiv tt$; 2) $g \equiv b$, причем значение *b* есть истина; 3) $g \equiv wait(time)$ через интервал времени, задаваемый выражением *time*; 4) $g \equiv read\ (i, l)$ либо $g \equiv write(i, l)$, и соответствующий выход (вход) канала *i* готов к обмену; в последнем случае, если канал *i* не готов к обмену, выполнение защищенной команды приостанавливается;

– альтернативная команда *ALT*(*gc*{; *gc*}) означает выполнение одной из защищенных команд, перечисленных в скобках, для которой значения защит являются истинными;

– команда цикла *LOOP*(*ALT*(*gc*{; *gc*})) означает многократное выполнение альтернативной команды; завершение цикла осуществляется, если все защиты *ALT*-команд являются булевскими выражениями и ни одно из значений защит не является истинным либо при выполнении команды *break*;

– канал типа *ALL*(*k*):*ALL*(*l*) вначале принимает данные по всем своим *k* входам, после чего становится готовым для передачи данных по всем *l* выходам; канал освобождается после передачи данных по всем своим выходам;

– канал типа *ALL*(*k*):*ANY*(*l*) аналогичен по входу каналу типа *ALL*(*k*):*ALL*(*l*); канал освобождается после передачи данных по любому из своих выходов;

– канал типа *ANY*(*k*):*ALL*(*l*) становится готовым для передачи данных, если он принял данные по одному из входов, и освобождается после передачи данных по всем своим выходам;

– канал типа *ANY*(*k*):*ANY*(*l*) становится готовым для передачи данных, если он принял данные по одному из входов, и освобождается после передачи данных по любому из своих выходов.

Выполнение АФС-программы предполагает одновременный запуск на выполнение всех функциональных процессов. В начале выполнения программы каналы не содержат данных и готовы к приему информации. АФС-программа завершается, если все функциональные процессы находятся в пассивном состоянии, то есть они либо завершили свое выполнение, либо находятся в состоянии ожидания ввода или вывода информации. Заметим при этом, что выполнение команды *wait*(*time*), хотя и приостанавливает выполнение процесса, не означает, что процесс находится в пассивном состоянии. Завершение всех функциональных процессов свидетельствует об успешном завершении АФС-программы. В противном случае имеет место блокировка.

Алгоритм верификации на основе семантических моделей выглядит следующим образом.

Вначале для программы на исходном языке программирования пишется соответствующая ей программа на языке АФС. Для АФС-программы сначала устанавливается семантическое значение, задавая априорную семантику каждого функционального процесса и канала связи программы и объединяя их операцией композиции в алгебре вычислительных последовательностей. Далее на основе семантического значения программы производится построение системы рекурсивных уравнений, опираясь на которую, осуществляется поиск ошибочных состояний (зацикливаний, блокировок и других).

Определим семантические области, на которые отображаются значения АФС-программ. Предполагаем, что заданными являются следующие семантические области:

1) множества значений

– элементарных команд *ACom* с типичным элементом *A*,

– булевских выражений *BExp* с типичным элементом *B*,

– команд ввода *PICom* с типичным элементом *IN*,

– команд вывода *POCom* с типичным элементом *OUT*;

2) множество взаимодействий *PCom* с типичным элементом γ;

3) множество *Const*, содержащее константы τ (тождественное преобразование), $\emptyset$ (останов), *T* (тождественно-истинное значение) и *F* (тождественно-ложное значение);

4) элементы *BREAK*, *EXIT* и *TIME*, являющиеся значениями команд завершения цикла, завершения процесса и ожидания соответственно.

Определим множество тестов *TEST* с типичным элементом β и множество действий *ACTION* с типичным элементом α следующим образом:

$\beta ::= T \mid F \mid B \mid TIME \mid IN \mid OUT \mid \gamma$

$\alpha ::= \tau \mid \emptyset \mid A \mid IN \mid OUT \mid \gamma \mid TIME \mid BREAK \mid EXIT$

Определим множество *вычислительных последовательностей* (ВП) *CPath* с типичным элементом *cp*:

$cp ::= \alpha \mid \beta \,\hat{}\, cp \mid cp_1 \circ cp_2$

Здесь символы ^ и ∘ – операции последовательной композиции ВП.

Через *SP* с типичным элементом *sp* обозначим множество всех подмножеств *CPath*.

Определим на множестве *SP* операции теоретико-множественного объединения (sp_1+sp_2), последовательной композиции ($sp_1 \circ sp_2$), параллельной композиции ($sp_1 \parallel sp_2$), теоретико-множественного вычитания ($sp_1 \setminus sp_2$) и минимальной фиксированной точки (sp^+).

Зададим семантическую функцию для функциональных процессов $\mathbb{F}: FProc \to SP$, где *FProc* – множество функциональных процессов с типичным элементом *fproc*:

$\mathbb{F}[fproc_1; fproc_2] = \mathbb{F}[fproc_1] \parallel \mathbb{F}[fproc_2]$

$\mathbb{F}[FUN\ i :: c] = \mathbb{C}[c]$, где $\mathbb{C}: Cmd \to SP$ – функция, сопоставляющая командам языка АФС множества ВП:

$\mathbb{C}[a] = \mathrm{A}$

$\mathbb{C}[skip] = \tau$

$\mathbb{C}[exit] = EXIT$

$\mathbb{C}[break] = BREAK$

$\mathbb{C}[wait(time)] = TIME$

$\mathbb{C}[read(i,l)] = IN_{i,l}$

$\mathbb{C}[write(i,k)] = OUT_{i,k}$

$\mathbb{C}[SEQ(c_1; c_2)] = \mathbb{C}[c_1] \circ \mathbb{C}[c_2]$

$\mathbb{C}[ALT(gc_1; gc_2)] = \mathbb{C}[gc_1] + \mathbb{C}[gc_2]$

$\mathbb{C}[tt \to c] = \mathbb{E}[tt]\,\hat{}\,\mathbb{C}[c]$

$\mathbb{C}[ff \to c] = \mathbb{E}[ff]\,\hat{}\,\mathbb{C}[c]$

$\mathbb{C}[b \to c] = \mathbb{E}[b]\,\hat{}\,\mathbb{C}[c]$

$\mathbb{C}[wait(time) \to c] = \mathbb{C}[wait(time)]\,\hat{}\,\mathbb{C}[c]$

$\mathbb{C}[\mathrm{read}(i,l) \to c] = \mathbb{C}[read(i,l)]\,\hat{}\,\mathbb{C}[c]$

$\mathbb{C}[write(i,k) \to c] = \mathbb{C}[write(i,k)]\,\hat{}\,\mathbb{C}[c]$

$\mathbb{C}[LOOP(ALT(gc))] = (\mathbb{C}[ALT(gc)])^+$

Определим семантическую функцию для выражений

$\mathbb{E}: Exp1 \to TEST$

$\mathbb{E}\,[tt] = T$

$\mathbb{E}\,[ff] = F$

$\mathbb{E}\,[b] = B$

Канал связи интерпретируется как циклический процесс, который вначале принимает данные от функциональных процессов, а затем, когда все необходимые данные (в соответствии с логикой входов этого канала) уже приняты, передает их функциональным процессам (в соответствии с логикой выходов этого канала).

Пример верификации драйвера на основе семантических моделей

В системе Linux драйверы являются модулями ядра и пишутся преимущественно на языке С. Однако структура драйвера отличается от обычной программы.

В модулях ядра отсутствует функция *main*(), вместо нее в драйвере необходимо зарегистрировать точки входа и выхода модуля с помощью макросов *module_init*(*name*) и *module_exit*(*name*) соответственно.

Для всех пользовательских программ драйвер выглядит как специальный файл, с которым можно взаимодействовать с помощью системных вызовов *open*, *close*, *read*, *write* и др.

В точке входа драйвера происходит инициализация файловых операций, которые может обрабатывать драйвер:

```
struct file_operations test_fops = {
        .open = test_open,
        .write = test_write,
        .read = test_read,
        .release = test_close,
};
```

Реализация файловых операций лежит на разработчике драйверов, и именно в файловых операциях наиболее часто обнаруживаются ошибки. Типичной ошибкой в драйверах Linux является ошибка синхронизации. Это связано с тем, что к драйверу могут обращаться сразу несколько процессов. В такой ситуации возможно возникновение состояний гонок и взаимоблокировки.

Рассмотрим пример драйвера устройств, в котором имеет место взаимоблокировка. Приведем наиболее значимые части рассматриваемого драйвера, а именно операции чтения и записи:

```
static struct semaphore sem1;
static struct semaphore sem2;

static ssize_t test_read(struct file *filp,
                         char __user *buff,
                         size_t count,
                         loff_t *offp)
{
    while (1) {
        down(&sem1);
        down(&sem2);
        printk("\nIn read function");
        up(&sem2);
        up(&sem1);
    }
    return count;
}

static ssize_t test_write(struct file *filp,
                          const char __user *buff,
                          size_t count,
                          loff_t *offp)
```

```
{
    while (1) {
        down(&sem2);
        down(&sem1);
        printk ("\nIn write function");
        up(&sem1);
        up(&sem2);
    }
    return count;
}
```

В этом примере используются два семафора *sem1* и *sem2*, которые необходимы для ограничения доступа к определенному участку кода. В функции инициализации драйвера счетчикам семафоров присваивается значение 1:

```
sema_init(&sem1, 1);
sema_init(&sem2, 1);
```

Счетчик семафора изменяется с помощью функций *down*(&*semaphore_name*) – уменьшение счетчика и *up*(&*semaphore_name*) – увеличение счетчика. Если счетчик семафора больше 0, значит, семафор находится в свободном состоянии, если он равен 0 – семафор занят.

Пусть данный драйвер использует программа, состоящая из двух потоков, один из которых производит чтение, а другой – запись в драйвер.

```
void * writer(void *arg) {
        int fptr;
        fptr=open("/dev/test", O_RDWR|O_NONBLOCK));
        write(fptr, buf, 4);
        close(fptr);
}
void * reader(void *arg) {
        int fptr;
        fptr=open("/dev/test", O_RDWR| O_NONBLOCK));
        read(fptr, buf, 4);
        close(fptr);
}
int main() {
        pthread_t rd, wr;
        pthread_create(&wr, NULL, writer, NULL);
        pthread_create(&rd, NULL, reader, NULL);
        pthread_join(wr, NULL);
        pthread_join(rd, NULL);
        return 0;
}
```

Здесь функция *pthread_create* отвечает за создание нового потока, функция *pthread_join* ожидает завершения потока и только после этого завершает выполнение основной программы.

При запуске приложения на выполнение программа некоторое время работает, а затем происходит взаимоблокировка. Это объясняется тем, что функция чтения уменьшает счетчик семафора *sem1* до нуля и ждет, пока функция записи увеличит значение счетчика семафора *sem2*. Но функция записи не может этого сделать, так как она ждет увеличения счетчика *sem1*.

В соответствии с [5] АФС-программа для приведенного примера имеет вид

```
NET
    CHAN 1 :: ALL (1) : ALL(1);
    CHAN 2 :: ALL (1): ALL(1);
BEGIN
    FUN 1:: LOOP(ALT(tt→SEQ(write(1,1), write(2,1),
    read(2,1), read(1,1))));
    FUN 2:: LOOP(ALT(tt→SEQ(write(2,1), write(1,1),
    read(1,1), read(2,1))));
END
```

Здесь *FUN* 1 соответствует операции чтения драйвера, а *FUN* 2 – операции записи. Каналы *CHAN* 1 и *CHAN* 2 моделируют семафоры *sem*1 и *sem*2 соответственно.

Определим априорную семантику функциональных процессов и каналов связи рассматриваемой АФС-программы:

$$\mathbb{F}[FUN\,1::LOOP(ALT(tt\rightarrow SEQ(write(1,1), write(2,1), read(2,1), read(1,1))))]=$$
$$=\mathbb{C}[LOOP(ALT(tt\rightarrow SEQ(write(1,1), write(2,1), read(2,1), read(1,1))))] =$$
$$=(\mathbb{C}[ALT(tt\rightarrow SEQ(write(1,1), write(2,1), read(2,1), read(1,1)))])^{+}=$$
$$=(\mathbb{C}[tt\rightarrow(SEQ(write(1,1), write(2,1), read(2,1), read(1,1)))])^{+}=$$
$$=(\mathbb{E}[tt]\hat{}\mathbb{C}[SEQ(write(1,1), write(2,1), read(2,1), read(1,1))])^{+}=$$
$$=(T\hat{}(\mathbb{C}[write(1,1)]^{\circ}\mathbb{C}[write(2,1)]^{\circ}\mathbb{C}[read(2,1)]\circ\mathbb{C}[read(1,1)]))^{+} =$$
$$=(T\hat{}(OUT_{1,1}\circ OUT_{2,1}\circ IN_{2,1}\circ IN_{1,1}))^{+}=(T\hat{}OUT_{1,1}\circ OUT_{2,1}\circ IN_{2,1}{}^{\circ}IN_{1,1})^{+}.$$

Аналогично для второго функционального процесса имеем:

$$\mathbb{F}[FUN\,2::LOOP(ALT(tt\rightarrow SEQ(write(2,1), write(1,1), read(1,1), read(2,1))))]=$$
$$=(T\hat{}\,OUT_{2,1}{}^{\circ}\,OUT_{1,1}{}^{\circ}\,IN_{1,1}{}^{\circ}\,IN_{2,1})^{+}.$$

Построим семантическое значение каналов связи, интерпретируя их как циклический процесс:

$$\mathbb{K}[CHAN\,1::ALL(1):ALL(1)] = (ALL(IN^{1,1})\circ ALL(OUT^{1,1}))^{+} =$$
$$=(IN^{1,1}\circ OUT^{1,1})^{+}$$

Аналогично для второго канала имеем:

$$\mathbb{K}[CHAN\,2::ALL(1):ALL(1)]=(IN^{2,1}\circ OUT^{2,1})^{+}$$

Здесь $\mathbb{F}$ и $\mathbb{K}$ – семантические функции вида $\mathbb{F}$:*Prog*→*SP* и $\mathbb{K}$:*Chan*→*SP*, сопоставляющие, соответственно, функциональным процессам *Prog* и каналам связи *Chan* множества ВП *SP*; *IN* и *OUT* – семантические значения команд ввода и вывода; *T* – константа, обозначающая тождественно-истинное значение; операция $^{+}$ задает минимальную фиксированную точку соответствующего оператора. Отметим, что для наглядности восприятия ВП использованы надстрочные индексы для значений *IN*(*OUT*), характеризующих ввод (вывод) информации в канал связи, тогда как подстрочные индексы используются для значений команд ввода (вывода) информации в функциональный процесс.

Обозначим полученные априорные семантические значения через *P*1, *P*2, *K*1 и *K*2 соответственно, а семантическое значение всей программы – через *P*, тогда *P*=*P*1||*P*2||*K*1||*K*2. Система рекурсивных уравнений, характеризующая *P*, имеет следующий вид:

$P_1=T\,\hat{}\,P_2+T\hat{}P_4$ $\qquad$ $P_5=\gamma_{1,1}\circ P_{13}$

$P_2=T\,\hat{}\,P_5+\gamma_{1,1}\circ P_8$ $\qquad$ $P_8=T\hat{}P_{16}+\gamma_{2,1}\circ P_{17}$

$P_4=T\,\hat{}\,P_9+\gamma_{2,1}\circ P_{12}$ $\qquad$ $P_9=\gamma_{2,1}\circ P_{18}$

P_{12}=T^P_{18}+$\gamma_{1,1}\circ P_{22}$
P_{13}=$\gamma_{2,1}\circ P_{24}$
P_{16}=$\gamma_{2,1}\circ P_{25}$
P_{17}=T^P_{25}+$\gamma_{2,1}\circ P_{29}$
P_{18}=$\gamma_{1,1}\circ P_{32}$
P_{22}=T^P_{32}+$\gamma_{1,1}\circ P_{37}$
P_{24}=∅
P_{25}=$\gamma_{2,1}\circ P_{40}$
P_{29}=T^P_{40}+$\gamma_{1,1}\circ P_1$
P_{32}=$\gamma_{1,1}\circ P_{48}$
P_{37}=T ^ P_{48}+$\gamma_{2,1}\circ P_1$
P_{40}=$\gamma_{1,1}\circ P_4$
P_{48}=$\gamma_{2,1}\circ P_2$

Очевидно, что выполнение потоков может быть заблокировано (уравнение P_{24}). Кроме того, можно определить последовательность действий, приводящую к блокировке. В данном случае такой последовательностью является $\gamma_{1,1} \circ \gamma_{2,1}$, что следует из уравнений P_1, P_2, P_5, P_{13} и P_{24}.

В заключение отметим, что в данной статье предложен метод верификации на основе семантики языков распределенного программирования. Рассмотрен пример верификации драйвера с использованием языка асинхронных функциональных схем.

В дальнейших исследованиях предлагается разработка автоматизированной системы верификации драйверов на основе семантических моделей. Представление семантического значения с помощью системы рекурсивных уравнений позволят дать ответы на вопрос о наличии тупиков и блокировок в программе, о достижимости, о возможности зацикливания программы и других ошибках.

Литература

1. Ball T., Bounimova E., Kumar R., Levin V. SLAM2: Static Driver Verification with Under 4 % False Alarms // Formal Methods in Computer-Aided Design. October 20–23, 2010. Lugano, Switzerland.

2. Post H., Küchlin W. Integration of static analysis for linux device driver verification // The 6th International. Conference on Integrated Formal Methods, July 5–7, 2007. Oxford, UK.

3. Witkowski T., Blanc N., Kroening D., Weissenbacher G. Model checking concurrent linux device drivers // In Proceedings of the twenty-second IEEE/ACM international conference on Automated software engineering. November 5–9, 2007. Atlanta, Georgia, USA.

4. Архитектура Linux Driver Verification / В.С. Мутилин [и др.]: тр. ИСП РАН. М., 2011. Т. 20. С. 163–187.

5. Кораблин Ю.П. Семантика языков распределенного программирования. М.: Изд-во МЭИ, 1996. 102 с.

УДК 004.051

МЕТОД ПЛАНИРОВАНИЯ РАЗМЕЩЕНИЯ ГРУППЫ ВИРТУАЛЬНЫХ МАШИН С ПЕРЕРАСПРЕДЕЛЕНИЕМ РЕСУРСОВ

В.П. Соловьев, к.т.н.; А.О. Удовиченко
(Московский государственный университет путей сообщения, wsolovjov@gmail.com, aspudovichenko@mail.ru)

Предложен метод планирования размещения группы виртуальных машин с перераспределением ресурсов. Рассмотрена задача размещения группы виртуальных машин, представлен разработанный метод и приведены результаты экспериментов, которые подтверждают его эффективность.

Ключевые слова: *информационная система, инфраструктура, виртуализация, виртуальная машина, управление ресурсами.*

В условиях возрастающей зависимости эффективности функционирования современных предприятий от информационных технологий все больше внимания уделяется автоматизации процессов управления *информационными системами* (ИС) и эффективности использования ресурсов. Одной из широко распространенных технологий, направленных на решение подобных задач, является технология *виртуальных машин* (ВМ). Она стала одним из наиболее значимых и быстро развивающихся направлений в корпоративных вычислениях.

По результатам исследований аналитических агентств, во всем мире наблюдается значительный рост рынка ПО виртуализации и ожидается, что в ближайшие годы более половины всей серверной нагрузки будет обрабатываться виртуализированными серверами [1]. Технология ВМ обеспечивает функционирование нескольких логических серверов на одном компьютере с уровнем изоляции (с точки зрения совместимости приложений и информационной безопасности), близким к уровню изоляции отдельных физических компьютеров [2].

Одной из основных трудностей при управлении ИС, построенной на основе технологии ВМ, является необходимость принимать во внимание распределение ресурсов между хостами. Достаточно часто в процессе функционирования ИС складываются ситуации, когда имеется большой объем свободных ресурсов, однако он распределен между хостами так, что его эффективное (в частности, с точки зрения загрузки ресурсов) использование невозможно. Например, объем ресурсов, достаточный для размещения ВМ, распределен

между двумя хостами поровну, что не позволяет использовать его для размещения. На практике поиск хоста с требуемым объемом ресурсов обычно сводится к перебору хостов. Если таковой не найден, считается, что размещение ВМ невозможно. Однако перераспределение ресурсов между хостами может высвободить необходимый объем ресурсов.

Очевидно, что задача размещения группы ВМ с перераспределением ресурсов ИС может быть представлена как задача об упаковке объектов предопределенной формы в контейнеры предопределенной формы, например, с целью, чтобы упаковать наибольшее количество объектов. Однако такая постановка задачи предполагает остановку ИС или отдельных ее компонентов и не гарантирует, что активные до перераспределения ВМ будут запущены вновь. Это решение обычно сопровождается высокими издержками и в большинстве случаев недопустимо. Следовательно, необходимо разработать такой метод размещения группы ВМ, который обеспечивал бы сохранение активности компонентов ИС.

В данной работе предложен и рассмотрен подобный метод с перераспределением ресурсов PRR (Placement of virtual machines with Resource Redistribution), целью которого является увеличение количества размещенных ВМ и уменьшение длительности процесса размещения, сформулирована задача размещения группы ВМ в общем виде, приведены результаты экспериментов.

Постановка задачи планирования размещения группы ВМ

Как правило, изначально ИС, построенная на основе технологии ВМ, проектируется с учетом запаса ресурсов. Однако в процессе функционирования неизбежны изменения как состояния ресурсов, так и их распределения, вызванные изменением количества хостов, ВМ, их требований к ресурсам и перемещением между хостами. В результате не всегда можно гарантировать наличие требуемого объема ресурсов на каком-либо хосте в будущем.

Представим наиболее распространенные задачи, решение которых осложняется из-за неудачного распределения ресурсов.

- Задача размещения множества новых ВМ. Одним из ее основных показателей является количество размещенных ВМ.
- Задача освобождения одного или группы хостов. Данная задача может преследовать различные цели: плановое обслуживание хостов, снижение затрат на электропитание, повышение коэффициента использования ресурсов и т.п. В каждом из случаев могут использоваться различные показатели, например, количество высвобожденных хостов, загрузка хостов, длительность перемещения группы машин.
- Задача обеспечения высокой готовности и отказоустойчивости. После возникновения отказа требуется быстрый перезапуск одной или группы ВМ на других хостах в кратчайшие сроки. В данном случае основными показателями являются длительность процесса размещения ВМ и их количество.

Остановка ИС или отдельных ее компонентов для перераспределения ресурсов в большинстве случаев недопустима или сопровождается высокими издержками. Поэтому перемещение ВМ между хостами должно выполняться с сохранением их активности. Для этой цели применяется технология горячей миграции [3], которая обеспечивает возможность перемещения ВМ без нарушения ее работы и поддерживается всеми наиболее распространенными платформами виртуализации.

Таким образом, необходима разработка метода планирования размещения группы ВМ с учетом перераспределения ресурсов и сохранения активности ВМ. Как уже было сказано, цели планирования размещения ВМ могут быть различными. Целью настоящей работы является улучшение двух показателей – количество размещенных ВМ и продолжительность их размещения. Следовательно, данная задача является задачей многокритериальной оптимизации. Существует несколько подходов к решению многокритериальных задач [4]. Очевидно, что первый критерий, то есть количество восстановленных ВМ, значительно важнее второго. Поэтому в данной работе выбран метод лексикографического упорядочения критериев, предполагающий, что последовательность, в которой перечислены критерии, определяет их значимость: каждый предшествующий критерий несравненно важнее любого из следующих за ним: $\max_{x \in D}(f_1(x), f_2(x))$, где D – множество допустимых решений; $f_1(x)$ – первый критерий; $f_2(x)$ – второй критерий.

Сначала решается однокритериальная задача: $\max_{x \in D}(f_1(x))$.

Пусть $D1 \subseteq D$ – множество оптимальных решений данной однокритериальной задачи. Если $D1$ – одноэлементное множество, то данное единственное решение и является оптимальным для всей задачи. В противном случае решается следующая однокритериальная задача: $\max_{x \in D1}(f_2(x))$.

Пусть $D2 \subseteq D1$ – множество оптимальных решений второй однокритериальной задачи. Если $D2$ – одноэлементное множество, то данное единственное решение и является оптимальным для всей задачи. В противном случае может оказаться, что оптимальных (при имеющемся лексикографическом упорядочении критериев) решений этой задачи более одного и все они эквивалентны.

Продолжительность размещения группы ВМ определяется как время от начала процесса размещения первой ВМ до завершения размещения последней с учетом их одновременного перемещения. Размещение ВМ может представлять собой последовательность вспомогательных перемещений других ВМ, в таком случае длительность ее размещения будет включать в себя и длительность реализации такой последовательности перемещений.

Решение сформулированной задачи осложняют следующие факторы:

– взаимное влияние параметров размещения ВМ; например, изменение параметров размещения одной ВМ может повлечь за собой снижение показателей оптимальности, если параметры размещения других ВМ не будут изменены;

– наличие множества локальных оптимумов;

– в некоторых случаях, например, при ограниченном объеме ресурсов на хостах, решение задачи может потребовать полного перебора для получения оптимальных результатов.

Поэтому для решения задачи размещения группы ВМ был предложен метод на основе разбиения исходной задачи планирования на более простые подзадачи определения порядка размещения ВМ и планирования размещения каждой из них.

Метод планирования размещения группы ВМ

Авторы предлагают комбинированный метод планирования размещения группы ВМ с перераспределением ресурсов PRR, состоящий из двух уровней оптимизации: глобальной оптимизации на уровне всего множества размещаемых ВМ и локальной оптимизации на уровне планирования размещения одной ВМ. На глобальном уровне выполняется решение многокритериальной задачи оптимизации, определяющей порядок размещения ВМ, с применением жадного алгоритма. На локальном используется метод динамического программирования для определения оптимальной последовательности перераспределения ресурсов с точки зрения времени размещения группы ВМ.

На ***глобальном уровне*** осуществляется поиск наилучшей последовательности размещения ВМ с точки зрения количества, которое будет размещено. На каждом шаге глобального уровня выбирается ВМ из оставшихся с наименьшими требованиями к ресурсам. При размещении ВМ обычно учитываются требования по нескольким типам ресурсов, а значит, задача выбора ВМ является многокритериальной. Для ее решения используется метод линейной свертки векторного критерия в одну функцию – обобщенный (агрегированный) критерий [4]: $V^i_{agr} = \sum_{j=1}^{K} c_j V^i_j$, где V^i_j – требование ВМ i к ресурсу типа j; V^i_{agr} – обобщенная оценка требований ВМ к ресурсам; c_i – некоторые положительные числа, характеризующие относительную важность ресурса типа j.

Для объединения различных критериев в один агрегированный необходимо привести их к общей единице измерения. Для этого перед запуском алгоритма определяются максимальные значения по каждому типу ресурсов среди всех хостов. Затем выбираются такие коэффициенты для этих критериев, что $V^{init}_j = 1000$, где j=1, …, K, в абсолютных безразмерных единицах. Коэффициенты важности приняты одинаковыми и равными 1/K, поскольку ни одному из типов ресурсов не отдается предпочтение при формировании последовательности освобождения хоста.

На ***локальном уровне*** определяются параметры размещения ВМ. Для планирования размещения каждой из ВМ предложен итерационный алгоритм, все итерации которого включают следующие шаги.

1. Формирование решений-кандидатов на данной итерации. Решение-кандидат представляет собой распределение ресурсов, отличающееся от исходного на q перемещений ВМ, где q – номер итерации. На первой итерации распределение ресурсов отличается от исходного на одно перемещение ВМ, на второй – на два последовательных перемещения и т.д.

2. Оценка длительности размещения группы рассмотренных ВМ для каждого решения-кандидата. Длительность размещения группы ВМ w^q_u для решения-кандидата u на итерации q принимается равной сумме длительностей перемещения и размещения ВМ с учетом одновременности данных операций.

3. Выбор наилучшего решения-кандидата по двум условиям:

– реализация данного решения-кандидата не приводит к нарушению требований ВМ к ресурсам: $E^k_j \geq 0$, где E^k_j, k=1, …, K, j=1, ..., C, – объем свободного ресурса типа j на хосте k после реализации решения-кандидата;

– данное решение-кандидат является наилучшим, если $w^q_{\min} = \min_{u \in Z_q}\{w^q_u, w^{q-1}_{\min}\}$, где $w^q_{\min}$ – наименьшая длительность размещения группы ВМ после итерации q; w^q_u – длительность размещения для решения-кандидата u итерации q; Z_q – множество решений-кандидатов итерации q.

Работа алгоритма планирования размещения ВМ завершается, если множество решений-кандидатов пусто или дальнейший поиск не ведет к улучшению решения. В качестве окончательного решения принимается наилучшее из найденных. Работа метода планирования размещения группы завершается после рассмотрения всех ее ВМ.

Модельный эксперимент

Для оценки работы предложенного метода было проведено имитационное моделирование. В качестве его среды использовано разработанное авторами приложение, моделирующее работу виртуального кластера, которое позволяет исследовать работу различных алгоритмов управления ВМ.

Моделирование проводилось для кластера, состоящего из 15 хостов и 80 ВМ. Рамещалась группа из 20 ВМ. В эксперименте принимались во внимание два типа ресурсов – объем памяти и загрузка процессора. Требования ВМ по каждому типу ресурсов задавались случайным образом в соответствии с равномерным распределением в интервале от 5 % до 30 % от базового уровня ресурсов хостов. Начальное количество ВМ на каждом хосте составляло от 4 до 8 и выбиралось в соответствии с законом равномерного распределения. Объем свободных ресурсов на каждом хосте, оставшийся после размещения ВМ, равномерно распределялся между ВМ таким образом, чтобы весь базовый уровень ресурсов хостов был занят. Длительность перемещения и размещения ВМ выбиралась в соответствии с нормальным распределением и математическим ожиданием 90 секунд.

Для тестирования разработанного метода планирования размещения группы ВМ проведено несколько серий опытов. Для каждой серии устанавливался максимальный свободный объем ресурса каждого типа, который может быть добавлен к базовому объему хоста. Данный объем задавался как процент от базового объема ресурса хоста, представляющий собой произведение номера серии опытов на 10. В каждом опыте серии к базовому объему ресурсов каждого хоста прибавлялся дополнительный объем свободного ресурса, он выбирался в пределах от 0 до заданного максимального объема для данной серии в соответствии с законом равномерного распределения.

Для сравнения проведена оценка количества размещенных ВМ при использовании алгоритма без учета возможности перераспределения ресурсов. Выбор хоста для размещения ВМ выполнялся на основе метода *наилучшего из подходящих* (Best Fit, BF) [5], обеспечивающего выбор хоста с объемом свободных ресурсов, более всего соответствующего требованиям ВМ.

На графиках рисунка 1 показано количество размещенных машин для методов PRR и BF.

Результаты эксперимента (рис. 1) показывают, что метод PRR при малом объеме свободного ресурса (максимальная величина дополнительного ресурса соответствует 5–15 %) позволяет выполнить размещение ВМ, в то время как второй метод не позволяет этого сделать. Кроме того, метод PRR обеспечивает размещение всей группы ВМ, начиная со 120 %, а метод BF – только со 170. Метод PRR, учитывающий распределение ресурсов

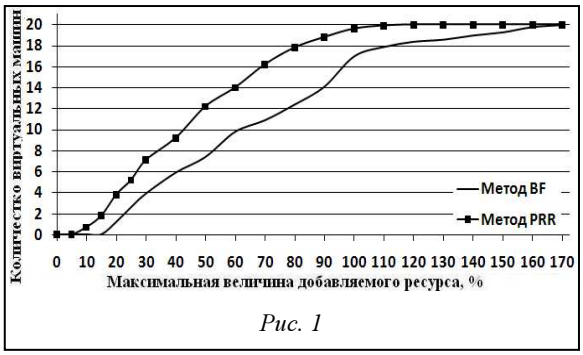


Рис. 1

между хостами, в среднем позволяет повысить количество размещенных ВМ на 50 % по сравнению с методом, не учитывающим такую возможность.

На рисунке 2 приведен график длительности размещения группы ВМ для метода PRR.

Рис. 2

На его основании можно сделать вывод, что на длительность размещения группы сильное влияние оказывает распределение ресурсов, об этом свидетельствует большое отклонение минимальных и максимальных значений от среднего.

В заключение можно отметить следующее. В работе предложен метод планирования размещения группы ВМ с перераспределением ресурсов. Работа метода основана на сведении задачи планирования размещения группы ВМ к решению более простых задач, таких как формирование последовательности размещения ВМ и планирование размещения отдельной ВМ.

Все сказанное позволяет сделать вывод о том, что распределение ресурсов между хостами в виртуальной инфраструктуре оказывает сильное влияние на работу ИС, а наличие достаточного объема свободных ресурсов не всегда может гарантировать возможность размещения ВМ.

Из результатов экспериментов видно, что разработанный метод позволяет разместить в среднем на 50 % больше ВМ при ограниченном объеме ресурсов, чем при использовании метода без учета распределения ресурсов.

Другим преимуществом предложенного метода является планирование размещения группы ВМ с учетом одновременного размещения/перемещения нескольких ВМ, что значительно сокращает длительность размещения всей группы.

Литература

1. Орлов С. Виртуализация «от и до» // Журнал сетевых решений / LAN. 2010. № 2.
2. Джонс Т. Обзор методов виртуализации, архитектур и реализаций // IBM developerWorks. 2007. URL: http://www.ibm.com/developerworks/ru/library/l-linuxvirt/index.html (дата обращения: 12.06.2011).
3. Clark C. [et. al.]. Live Migration of Virtual Machines // 2nd Symposium on Networked Systems Design and Implementation. 2005. Vol. 2, pp. 273–286.
4. Коган Д.И. Задачи и методы конечномерной оптимизации: Ч. 3. Динамическое программирование и дискретная многокритериальная оптимизация. Н. Новгород: Изд-во НГУ, 2004. 157 с.
5. Гэри М., Джонсон Д. Вычислительные машины и труднорешаемые задачи. М.: Мир, 1982. 439 с.

УДК 681.3

АВТОМАТИЧЕСКОЕ ПОСТРОЕНИЕ И ОБОБЩЕНИЕ СХЕМНЫХ РЕШЕНИЙ ПРИ ПРОЕКТИРОВАНИИ СИСТЕМ УПРАВЛЕНИЯ

Н.Н. Филатова, д.т.н.; А.Г. Требухин
(Тверской государственный технический университет, nfilatova99@mail.ru)

Рассмотрен программный комплекс, расширяющий возможности САПР систем управления путем автоматической генерации объемов понятий о заданном классе функциональных схем автоматизации и автоматически унифицирующий и расширяющий свой опыт.

Ключевые слова: *схема системы управления, дерево схемотехнических решений, приближенное множество, решающее правило.*

Анализ средств САПР систем промышленной автоматики показывает, что они применяются в основном на стадии технического проектирования при разработке принципиальных электрических схем и трансляции полученных описаний в монтажные схемы и заказные спецификации на материалы и оборудование. Так, например, в САПР «СА» (разработка Российского федерального ядерного центра, г. Саров) для проектирования схем используются обширная БД элементов и специализированный редактор схем. Система «САПР-АЛЬФА» (разработка фирмы «САПР-АЛЬФА», г. Москва) поддерживает функции сбора, контроля и передачи данных между компонентами проекта, ведения БД, автоматизированного редактирования схем, использования типовых решений. Система отличается высоким значением показателя «цена/качество» [1]. Достоинствами САПР E3.series (CIM-Team, Германия, холдинг Zuken) являются ее многофункциональность и развитый пользовательский интерфейс, а недостатком – необходимость создавать функциональные схемы вручную. Только система EPLAN (Software&Service, Германия) позволяет в полуавтоматическом режиме генерировать функциональные схемы, используя типовые технические решения, а также осуществлять предварительную оценку стоимости реализации всей системы автоматизации или оценку стоимости ее отдельных контуров.

Проведенное сравнение функциональных возможностей САПР систем автоматизации показывает, что для повышения качества предварительной проработки технических решений на начальных этапах разработки технического обеспечения АСУ ТП необходимо расширить их возможности двумя типами средств: с помощью автоматической генерации вариантов функциональных и других взаимосвязанных типов схем систем автоматизации и обобщения схемотехнических решений, полученных в процессе функционирования САПР.

Генераторы схемотехнических решений позволяют расширить множество вариантов схем автоматизации, которые проектировщик может проанализировать. Инженер освобождается от полумеханических процедур перебора вариантов структурных решений, сосредоточив свое внимание только на их оценках. Это создает условия для проработки значительного числа вариантов технической реализации структурных схем (n>>2) и способствует получению более точных и обоснованных технических решений.

В настоящее время опыт САПР реализуется в виде различных БД с архивом результатов (схем автоматизации). Однако эффективность САПР возрастет, если реализовать процедуры автоматического формирования обобщений путем анализа созданных схемотехнических решений. Подобный механизм позволит говорить о создании САПР, расширяющей свою БЗ на основе собственного опыта.

В работе [2] авторами представлен алгоритм автоматической генерации множества описаний функциональных схем на основе заданной структурной схемы и правил построения *деревьев схемотехнических решений* (ДСР). Любой вершине ДСР соответствует блок *технических средств автоматизации* (ТСА). Корневой вершиной дерева

является регулятор (рис. 1а). Число ДСР не меньше числа регуляторов, включенных в структурную схему. Для любой пары связанных вершин ДСР (родительской и дочерней) выполняется условие идентичности функций преобразования и согласованности типов и диапазонов сигналов соответствующих им блоков. Каждая ветвь дерева представляет собой фрагмент измерительной, исполнительной, корректирующей или интерфейсной цепи. На основе анализа каждого ДСР строятся шаблоны – технические реализации некоторых структурных схем или их фрагментов (рис. 1б).

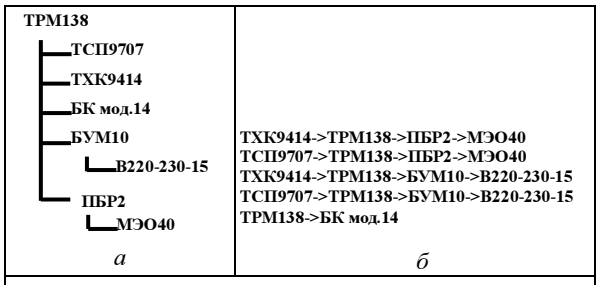


Рис. 1. Пример а) дерева схемотехнических решений на регуляторе ТРМ138; б) шаблонных схем с регулятором ТРМ138

Алгоритм автоматической генерации вариантов функциональных схем имеет следующие особенности.

1. При компоновке из шаблонов функциональных схем с большим количеством измерительных и исполнительных цепей корректность каждого генерируемого варианта определяется только наличием свободных входов (выходов) используемого регулятора, а условия согласованности диапазонов и типов сигналов используемых элементов не проверяются.

2. Для построения одноконтурных и сложных многоконтурных систем при условии идентичности номенклатуры используемых ТСА набор шаблонов содержит одни и те же схемы. Обращения к БД ТСА осуществляются на этапе генерации шаблонных схем, имеющих простую одноканальную структуру. Временная сложность компоновки шаблонов пренебрежимо мала по сравнению с их построением. Следовательно, можно считать, что описанный выше алгоритм практически инвариантен к сложности проектируемой схемы.

Алгоритм генерации ДСР позволяет организовать полный перебор на множестве блоков ТСА, доступных в БД ТСА. Но в этом случае мощность получаемого множества технических решений (вариантов функциональных схем) полностью зависит от размеров БД и может достигать нескольких десятков или сотен схем.

Первым шагом к автоматическому анализу построенного множества решений будет создание методики оценки альтернатив схемотехнических решений на основе теории приближенных множеств [3].

С учетом того, что шаблон функциональной схемы представляет собой описание реализации одного канала регулирования, а число шаблонов намного меньше числа альтернатив функциональных схем, для анализа и обобщения целесообразно использовать именно шаблоны.

Множество шаблонов (X), сформированных при рассмотрении одного типа структурной схемы, анализируется экспертом. Каждый шаблон расширяется оценками (A), характеризующими качество проектных решений. На множестве A выделяется так называемый решающий атрибут d, который в частном случае может принимать значения $d1$:= «Использовать шаблон для функциональной схемы» или $d2$:= «Не использовать шаблон для функциональной схемы».

Пара вида $S=(X, A)$ образует информационную систему, а ее расширение вида $S=(X, A\cup\{d\})$ – решающую систему. На множестве шаблонов X можно построить отношение неразличимости $IND(A)$: если x_i, $x_j \in IND(A)$, то x_i и x_j неразличимы по значениям атрибутов A.

Приближенное множество X образует пара $< \underline{A}X, \overline{A}X >$. Нижнее приближение $\underline{A}X$ множества X является объединением классов эквивалентности отношения неразличимости, все объекты которых входят в X. Верхнее приближение $\overline{A}X$ является объединением классов эквивалентности, в которых хотя бы часть объектов относится к X. Множество $BN_A = \overline{A}X - \underline{A}X$ называется граничной областью X и состоит из объектов, которые нельзя уверенно отнести к нему [4].

На основе анализа S строятся правила вида [3]:

$$\underline{A}X \to X, \quad (1)$$

$$U \setminus \overline{A}X \to \neg X, \quad (2)$$

$$BN_A \to \text{возможно } X. \quad (3)$$

Используя (1–3), можно получать непротиворечивые решающие правила, по которым тот или иной объект или принимается к дальнейшему рассмотрению, или отклоняется.

Анализ выделенных шаблонов выполняется в два этапа.

1. Любому шаблону ставится в соответствие число от 0 до 1 – его вес. Весовым значением может являться отличное от нуля минимальное нечеткое значение одной из оценок (A), характеризующих качество проектных решений, указанных в правилах, по которым шаблон принят к дальнейшему рассмотрению. Для этого предварительно создаются функции принадлежности критериев оценивания схем (A): «желаемая погрешность датчика», «наиболее предпочтительная стоимость регулятора» и др.

2. Из набора шаблонов (X) для дальнейшей генерации функциональных схем исключаются те, весовые значения которых меньше заданного.

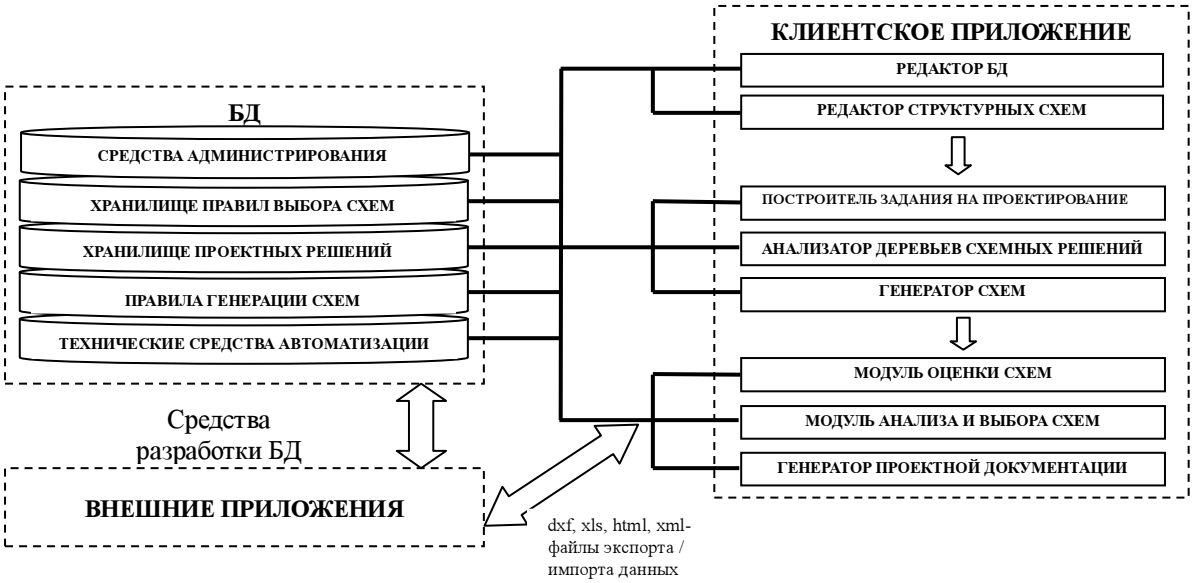


Рис. 2. Структура программного комплекса «Controlic's»

Программный комплекс «Controlic's»

Рассмотренные алгоритмы реализованы в программном комплексе для проектирования систем автоматизации технологических процессов «Controlic's», разрабатываемом в Тверском государственном техническом университете (рис. 2).

Программный комплекс «Controlic's» работает с ПСАПР – БД «Технические средства автоматизации» (Свид. о госрегистр. № 2007620224) [5]. В качестве СУБД проекта используется объектно-реляционная система PostgreSQL 8.2.

Для работы с ПК «Controlic's» необходимо сформировать задание на проектирование, включающее структурную схему системы автоматизации и ограничения на область поиска блоков ТСА. Структурную схему выбирают из набора или загружают из файла, которую можно создать с помощью редактора схем ПСАПР [5].

В «Controlic's» полностью исключена необходимость подбора блоков ТСА пользователем с привязкой к конкретному звену структурной схемы. Выбор блоков для технической реализации измерительных и исполнительных цепей структурной схемы осуществляется автоматически с учетом введенных в задание ограничений.

Предусмотрено несколько уровней детализации требований к исходным данным:

– определять только тип контролируемых величин (температуры, давления и др.) и способ их контроля и регулирования (световая сигнализация, ШИМ-регулирование);

– задавать класс используемых датчиков, регуляторов и исполнительных устройств (термометр сопротивления, датчик дифференциального давления, специализированный регулятор температуры, исполнительный механизм постоянной скорости и др.);

– задавать конкретные элементы ТСА (термометр сопротивления медный 9620, ТРМ151).

Из БД в задание на проектирование будут выбраны все датчики, исполнительные устройства и регуляторы, удовлетворяющие заданным ограничениям. Все остальные элементы (нормирующие преобразователи, пусковые устройства и др.) в процессе генерации подбираются автоматически. На основе анализа этих блоков ТСА формируется набор шаблонов для каждой базовой структуры. Далее «Controlic's» может работать в трех режимах.

Генерация на основе шаблонов полного набора моделей функциональных схем. Например, необходимо разработать схему системы автоматизации, включающую контур управления задвижкой при помощи *механизма электрического однооборотного* (МЭО) и контур сигнализации соотношения температур объекта в двух точках контроля. Следует рассмотреть варианты реализации схемы на микропроцессорных контроллерах ремиконт БК14, Термодат-12 и ТРМ151. В задание на проектирование включены термометр сопротивления 9620, МЭО40, сигнальная лампа накаливания. Для структурной схемы (рис. 3) «Con-

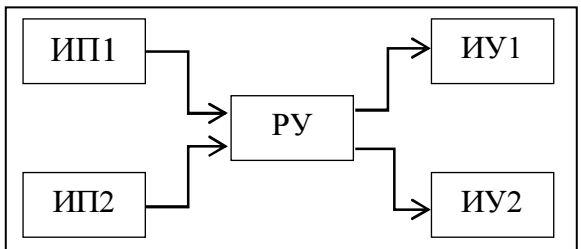


Рис. 3. Структурная схема системы

Термометр сопротивления медный 9620 **(ИП1)**	->	ТРМ151Щ1КК00 **(РУ)**
Термометр сопротивления медный 9620 **(ИП2)**	->	ТРМ151Щ1КК00 **(РУ)**
ТРМ151Щ1КК00 **(РУ)**	->	Пускатель бесконтактный реверсивный
ТРМ151Щ1КК00 **(РУ)**	->	Блок усилителей мощности БУМ10
Пускатель бесконтактный реверсивный	->	Механизм электрический однооборотный МЭО40 **(ИУ1)**
Блок усилителей мощности БУМ10	->	Сигнальная лампа накаливания **(ИУ2)**

Рис. 4. Модель одного варианта функциональной схемы

trolic's» генерирует пять вариантов функциональных схем (рис. 4).

При необходимости функциональные схемы можно сохранить в табличном виде в форматах xml, html, xls. Предусмотрена возможность экспорта данных в AutoCAD в dxf-формате.

Обобщение схемных решений систем автоматизации – анализ шаблонов. Шаблоны оцениваются по количественным критериям: «срок службы», «погрешность датчика», «обобщенный показатель технологичности», «ремонтопригодность», «стоимость канала регулятора». Множество (A) можно расширить, добавив качественные показатели: наличие механизма автонастройки регулятора схемы, резервирование каналов, возможность безударного переключения управления цепями нагрузки и др.

Например, на рисунке 5 представлена обучающая выборка шаблонов схем регулирования температуры.

В результате ее анализа найдены нижнее и верхнее приближения множества X шаблонов и сформированы решающие правила:

***IND*:** $\{X2, X4\}$, $\{X9\}$, $\{X5, X7\}$, $\{X1, X3\}$, $\{X10\}$, $\{X6, X8\}$,

$\underline{AX}$**:** $X9 \mid X1 \mid X3 \mid X10 \mid$, $\overline{AX}$: $X9 \mid X1 \mid X3 \mid X10 \mid X6 \mid X8 \mid$, ***BNa*:** $X6 \mid X8 \mid$.

По шаблонам $\{X1, X3\}$ решение принимается, если {Погрешность датчика меньше 4 %}, а также {Стоимость канала регулятора больше 4 536 руб.}. По шаблонам $X9$ и $X10$ решение принимается, если {Стоимость канала регулятора от 2 485 до 4 536 руб.}.

Все правила сохраняются в БД.

Анализ схем рабочего множества. В этом случае для заданной структурной схемы строятся варианты функциональных схем, но для их генерации из набора шаблонов исключаются те, которые не принимаются ни по одному правилу или с весом, меньше заданного (по умолчанию 0,1). Например, из девяти шаблонов, приведенных на рисунке 6, по решающим правилам принимаются только три.

■ X01 ТСМ9620—>БУС10_50_150—>РН1/5—>БК1 мод.14—>БУМ10—>В 220-230-15
□ X02 ТХК 9414—>БУТ10 ХК—>РН1/5—>БК1 мод.14—>БУМ10—>В 220-230-15
■ X03 ТСМ9620—>БУС10_50_150—>КБС3—>БК1 мод.14—>БУМ10—>В 220-230-15
□ X04 ТХК 9414—>БУТ10 ХК—>КБС3—>БК1 мод.14—>БУМ10—>В 220-230-15
□ X05 ТХК 9414—>модель 1УВ/2Р/1Т/485/64k—>В 220-230-15
■ X06 ТСМ9620—>модель 1УВ/2Р/1Т/485/64k—>В 220-230-15
□ X07 ТХК 9414—>модель 1УВ/2Р/1Т/485/64k—>БУМ10—>В 220-230-15
□ X08 ТСМ9620—>модель 1УВ/2Р/1Т/485/64k—>БУМ10—>В 220-230-15
■ X09 ТХК 9414—>ТРМ151КК00—>БУМ10—>В 220-230-15
■ X10 ТСМ9620—>ТРМ151КК00—>БУМ10—>В 220-230-15

Рис. 5. Решающая система для генерации правил

X01 w=0,102 ТСП9707—>БУС10 _50_200—>РН1/5—>БК1 мод.14—>БУМ10—>В 220-230-15
X02 ИСКЛ ТХК 9414—>БУТ10 ХК—>РН1/5—>БК1 мод.14—>БУМ10—>В 220-230-15
X03 w=0,102 ТСП9707—>БУС10 _50_200—>КБС3—>БК1 мод.14—>БУМ10—>В 220-230-15
X04 ИСКЛ ТХК 9414—>БУТ10 ХК—>КБС3—>БК1 мод.14—>БУМ10—>В 220-230-15
X05 ИСКЛ ТХК 9414—>модель 1УВ/2Р/1Т/485/64k—>В 220-230-15
X06 ИСКЛ ТСП9707—>модель 1УВ/2Р/1Т/485/64k—>В 220-230-15
X07 ИСКЛ ТХК 9414—>модель 1УВ/2Р/1Т/485/64k—>БУМ10—>В 220-230-15
X08 ИСКЛ ТСП9707—>модель 1УВ/2Р/1Т/485/64k—>БУМ10—>В 220-230-15
X09 w=0,622 ТХК 9414—>ТРМ151КК00—>БУМ10—>В 220-230-15

Рис. 6. Исключение наименее перспективных шаблонов при генерации функциональных схем

Предложенный вариант программного комплекса расширяет возможности САПР систем управления и способствует повышению степени проработки вариантов технической реализации систем автоматизации на этапах, предшествующих созданию принципиальных электрических схем.

Литература

1. Одегов П. Программный комплекс «САПР-АЛЬФА»: краткий обзор // САПР и графика. 2009. № 2. С. 52–55.

2. Филатова Н.Н., Требухин А.Г., Ахремчик О.Л. Автоматическая генерация деревьев схемотехнических решений // AIS-IT'11: тр. Междунар. конгр. по интеллект. сист. и информац. технологиям. М.: Физматлит, 2011. Т. 2. С. 122–130.

3. Достоверный и правдоподобный вывод в интеллектуальных системах / В.Н. Вагин [и др.]; [под ред. В.Н. Вагина, Д.А. Поспелова]. М.: Физматлит, 2008. 712 с.

4. Pawlak Z. Rough Sets // International Journal of Information and Computer Science, 1982. № 11 (5), pp. 341–356.

5. Ахремчик О.Л. Программная система для анализа технических решений при проектировании // Программные продукты и системы. 2009. № 1. С. 29–31.

УДК 519.876.5

МНОГОПОДХОДНОЕ ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ ДЛЯ ОПТИМИЗАЦИИ СОСТАВА СПЕЦИАЛИСТОВ КОНСАЛТИНГОВОЙ КОМПАНИИ

С.Г. Митрошин; В.В. Пикулин, к.т.н.
(Пензенская государственная технологическая академия, pvv@pgta.ru)

Рассматривается применение многоподходного имитационного моделирования для повышения эффективности деятельности консалтинговой компании путем изменения состава специалистов. Приводятся пример имитационной модели, разработанной в среде AnyLogic, и результаты ее использования.

Ключевые слова: *консалтинг, планирование персонала, многоподходное имитационное моделирование.*

При управлении компаниями, ориентированными на оказание консультационных услуг, ключевой задачей является построение оптимальных организационных структур, так как основной источник доходов и затрат – это специалисты компании. Объектом исследования является деятельность фирмы, занимающейся проектированием, разработкой, внедрением и сопровождением автоматизированных систем, предоставлением консалтинговых услуг. Компания заинтересована в наличии оптимального количества специалистов, приносящих доход своим участием в различных проектах. В то же время ей невыгодно содержать сотрудников, не задействованных в проектах, так как это приводит к убыткам (выплата должностных окладов, амортизация используемого оборудования, накладные и транспортные расходы).

Поскольку деятельность компании связана с обслуживанием клиентов на основе поступающих заявок, то есть потока дискретных событий, для решения задач планирования штата актуально применение средств компьютерного имитационного моделирования. Для решения простых, некритичных задач клиенты заинтересованы в привлечении специалистов со средним и низким профессиональным уровнем с целью сокращения затрат на обслуживание. К решению срочных задач или задач, имеющих повышенные требования к качеству, привлекаются специалисты высокого уровня. Требуемый профессиональный уровень исполнителя определяется свойствами поступающей заявки.

В обобщенной постановке задача формулируется следующим образом: компании необходимо обеспечить обслуживание поступающих заявок с максимизацией прибыли $P(T)$ в течение заданного интервала времени T.

Введем обозначения переменных и показателей, характеризующих рассматриваемый процесс.

Q – множество обслуживаемых (сопровождаемых) компанией проектов (систем, производственных объектов).

J – множество производственных задач по обслуживаемым проектам.

J_i – i-е подмножество однотипных задач; тип производственной задачи определяется ее назначением и содержанием, соответствующим определенной профессии: программист, инженер-механик, инженер-электрик и т.п.; для решения задач каждого типа должны привлекаться специалисты соответствующей профессии; можно считать, что подмножества J_i не пересекаются, то есть справедливы выражения $J=\bigcup_{i=1}^{n} J_i$; $\bigcap_{i=1}^{n} J_i=\varnothing$; если рассматриваются однотипные производственные задачи, то n=1.

R_{ij} – уровень сложности j-й производственной задачи i-го типа; однотипные задачи могут группироваться по уровню сложности, например, если j-я и k-я задачи i-го типа имеют один и тот же уровень сложности ($R_{ij}=R_{ik}$), то они относятся к одной группе.

V – множество профессий сотрудников, выполняющих производственные функции; специалист i-й профессии обеспечивает решение множества $J(V_i)$ типов производственных задач, при этом должно выполняться условие $J\subseteq\bigcup_{i=1}^{m} J(V_i)$.

В частном случае $|V|$=1.

K_i – множество уровней квалификации (должностей или специализаций) сотрудников i-й профессии.

K_{ij} – j-й уровень квалификации i-й профессии; уровнем профессиональной квалификации определяются также производительность труда и качество выполняемых работ (чем выше уровень квалификации, тем выше производительность труда и качество выполняемых работ).

Для каждой профессии $v\in V$ установлено множество уровней квалификации специалистов K_v, $|K_v|\geq 1$, и каждому $k\in K_v$ соответствует определенное количество сотрудников N_k (плановое или фактическое), что формально можно записать следующим образом: $\forall\ v\in V\ \exists\ f{:}v\rightarrow K_v\,|\,\forall\ k\in K_v\ \exists\ N_k\,|\ k\leftrightarrow N_k\ \&\ N_k\in[0, 1, 2, \ldots]$.

J_{ij} – подмножество производственных задач i-го типа j-го уровня сложности, которые выполнит специалист K_{ij}-й квалификации; между K_{ij} и J_{ij} может быть установлено взаимно-однозначное ($K_{ij}\leftrightarrow J_{ij}$) или многозначное ($K_{ij}\rightarrow\{J_{i,1}, \ldots, J_{i,j-1}, J_{i,j}\}$) соответствие, если специалист i-й профессии j-го

уровня квалификации может выполнять работы i-го типа от первого до j-го уровня сложности включительно.

$\lambda_{q,ij}$ – интенсивность поступления по q-му проекту заявок i-го типа j-го уровня сложности ($R_{q,ij}$).

$v_{q,ij}$ – объем работ (в часах) по заявке i-го типа j-го уровня сложности q-го проекта ($R_{q,ij}$).

$U_{s,ij}(T)$ – объем работ (в часах), который может быть выполнен специалистом i-й профессии j-го уровня квалификации в течение заданного интервала времени T.

C_{ij} – цена услуг специалиста i-й профессии j-го уровня квалификации, привлеченного для выполнения работ (руб./ч); если для s-го специалиста устанавливаются индивидуальные расценки на работы, то для обозначения этого факта следует использовать дополнительный индекс, тогда следует обозначать цену как $C_{s,ij}$ (руб./ч). Правила назначения цены: 1) взаимно-однозначное соответствие цены услуги и уровня сложности производственной задачи ($C_{ij}\leftrightarrow R_{ij}$); 2) если для выполнения работы i-го типа j-го уровня сложности привлекается специалист более высокой квалификации, чем требуется, то цена услуги устанавливается в соответствии с квалификацией привлекаемого специалиста ($C_{s,i,j+1}>C_{ij}$).

Для использования формальных правил назначения цены каждая заявка должна иметь соответствующий классификационный признак $z\in Z$, например:

z=1 – заявка должна обслуживаться только на основе взаимно-однозначного соответствия уровня сложности производственной задачи уровню квалификации специалиста ($K_{ij}\leftrightarrow J_{ij}$); дополнительно может быть установлено условие ограничения продолжительности обслуживания заявки ($t_{\text{обсл.},q,ij}\le t_{\text{доп.},q,ij}$),

z=2 – заявка может обслуживаться специалистом более высокой квалификации; при этом стоимость услуги соответствует квалификации привлекаемого специалиста ($C_{s,i,j+1}>C_{ij}$),

z=3 – то же, что при z=2, только дополнительно требуется минимизация продолжительности обслуживания (то есть заказчику требуется минимизация продолжительности обслуживания даже по завышенной цене); этот признак может соответствовать внеочередному обслуживанию за дополнительную плату.

Значение прибыли на k-й итерации моделируемого процесса оценивается как $P_k(T)=D_k(T)-R_k(T)$, где $D_k(T)$, $R_k(T)$ – доходы и расходы за период времени T соответственно:

$$D_k(T)=\sum_{q(k)}\sum_{i(k)}\sum_{j(k)} l_{q,ij,k}v_{q,ij,k}C_{q,ij},$$

$l_{q,ij,k}$ – количество заявок i-го типа j-го уровня сложности, выполненных по q-му проекту на k-й итерации;

$$R_k(T)=\sum_{i(k)}\sum_{j(k)} g_{ij,k}H_{ij},$$

$g_{ij,k}$ – количество сотрудников i-й профессии j-го уровня квалификации, обслуживавших заявки на k-й итерации; H_{ij} – норматив расходов на одного сотрудника i-й профессии j-го уровня квалификации; может использоваться более детальная калькуляция расходов при наличии соответствующей математической модели.

Случайными величинами в данной модели являются интенсивность поступления заявок ($\lambda_{q,ij}$), уровень сложности ($R_{q,ij}$), объем работ по заявке ($v_{q,ij}$), объем работ, который может быть выполнен специалистом в течение заданного интервала времени ($U_s(T)$ ч).

Оптимизируемым параметром, варьируемым при выполнении итераций в целях поиска наибольшего значения целевой функции $P(T)$, является количество специалистов i-й профессии и j-го квалификационного уровня $g_{ij}(T)$, привлекаемых на заданное время T().

Значение целевой функции процесса $P(T)$ можно определить путем варьирования значений оптимизируемых параметров при выполнении итераций (циклов моделирования).

Количество заявок j-го типа за время T зависит от количества сопровождаемых компанией проектов и интенсивности поступления заявок на обслуживание по каждому проекту: $k_j(T)=\sum_{q_j} M_q\cdot T_q$, где q_j – количество проектов j-го типа; M_q – интенсивность поступления заявок по q-му проекту; T_q – продолжительность сопровождения q-го проекта.

Значение целевой функции должно определяться при следующих ограничениях:

$$\sum_{q(k)}\sum_{i(k)}\sum_{j(k)} l_{q,ij,k}v_{q,ij,k}\le\sum_{i(k)}\sum_{j(k)} g_{ij,k}U_{ij}(T).$$

В данной работе рассматривается модель оптимизации указанной деятельности компании для частного случая при n=1, $|V|$=1, K_i=3, $J_i=\bigcup_{j=1}^{m(i)} J_{ij}$; $\bigcap_{j=1}^{m(i)} J_{ij}=\varnothing$, $K_{ij}\leftrightarrow J_{ij}$, $C_{ij}\leftrightarrow R_{ij}$, $|Q|\le 50$. Оптимизируемым параметром является количество специалистов каждого из трех уровней квалификации.

Для решения поставленной задачи рационально применение методики *многоподходного имитационного моделирования* (МИМ) [1], позволяющей объединить в одной модели различные методологии моделирования (в данном случае – дискретно-событийное и агентное).

Авторами разработаны имитационная модель определенной выше деятельности, механизмы управления ее параметрами, средства сбора статистики. Средой для разработки модели выбрана система AnyLogic компании XJ Technologies, позволяющая применять МИМ с описанием различных частей гетерогенных систем и с использованием различных подходов к моделированию [2].

Разработанная модель состоит из следующих элементов:

Main – основной класс, определяющий взаимодействие объектов;

Client – класс, описывающий поведение клиентов компании (класс – агент);

Firm – класс, определяющий деятельность компании, обработку заказов;

Specialist – класс, описывающий поведение специалистов (класс – агент);

Request – класс обрабатываемой заявки (Java-класс);

Simulation – эксперимент, позволяющий выполнить имитационное моделирование при заданных параметрах;

Optimization – оптимизационный эксперимент, позволяющий вычислить оптимальные значения заданных параметров.

Процесс обслуживания заявок описывается с помощью дискретно-событийного моделирования с использованием объектов библиотеки Enterprise Library (рис. 1):

– заявки класса Request, поступающие в объект Firm, направляются на вход процесса обслуживания (enter) и далее в очередь (queue) на распределение по специалистам;

– при выходе из очереди выполняются задержка заявки (setSpecDelay) и назначение ответственного специалиста (setSpecialist); время задержки заявки распределено согласно треугольному закону

$$f(x)=\begin{cases}\dfrac{2(x-\min)}{(\max-\min)(\mathrm{mod}-\min)}, & \min\le x\le \mathrm{mod},\\ \dfrac{2(\max-x)}{(\max-\min)(\max-\mathrm{mod})}, & \mathrm{mod}<x\le\max,\end{cases} \quad (1)$$

где min=0,1; max=1; mod=0,2;

– функция setSpecialist определяет наименее загруженного компетентного специалиста по работе с типом системы, указанным в заявке, и выполняет проверку возможности передачи заявки специалисту (readyToSend); при отсутствии свободных специалистов требуемой компетенции заявка отмечается как упущенная и передается объекту sinkLost, который уничтожает ее и оповещает клиента о том, что заявка не обработана;

– если заявке был назначен ответственный за ее выполнение специалист, она пересылается (exitToSpec) соответствующему экземпляру объекта Specialists; после обработки заявка поступает на вход процесса enterFinished и передается объекту sinkFinished, который уничтожает ее и оповещает владельца заявки о том, что она обработана;

– заявки, не обработанные специалистом по каким-либо причинам (например, по истечении установленного периода обслуживания), возвращаются в процесс на вход enterLost и передаются объекту sinkLost, который их уничтожает и оповещает клиента о том, что заявка не обработана;

– если время ожидания в очереди queue превышает заданные нормативы, заявки также попадают к объекту sinkLost.

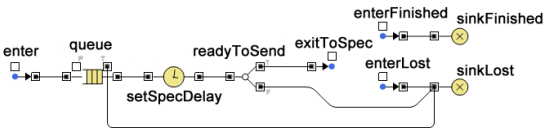


Рис. 1. Процесс обработки заявки: диаграмма класса Firm

Класс Specialist является агентом, что позволяет моделировать произвольное децентрализованное поведение отдельных экземпляров объектов и их взаимодействие. Процесс обработки заявки специалистом достаточно детерминирован и моделируется с помощью инструментария дискретно-событийного моделирования (рис. 2):

– заявки Request поступают на входы процесса (enter, enterFromSpec) и передаются объекту seize, который выделяет необходимое количество ресурсов из пула (resource) и передает заявки в очередь на обслуживание (inWork); количество ресурсов в пуле (resource) моделирует возможности специалиста заниматься параллельной обработкой нескольких задач; количество ресурсов K_R определяется дискретным равномерным распределением $p(x)=\left\{\dfrac{1}{b-a+1}\right.$; в данном эксперименте принято $K_R\in[1;\ 3]$, a=1, b=3;

– если в пуле (resource) не хватает свободных ресурсов, заявки остаются в очереди объекта inWork и ожидают высвобождения ресурсов;

– из очереди на обслуживание (inWork) заявки передаются объекту delay, который моделирует их обработку; время обработки заявок зависит от объема работ по ним (параметр value) и коэффициента задержки, который распределен согласно треугольному закону (1) (min=0,5; max=1,5; mod=1);

– после обработки заявки (delay) ресурс (release) высвобождается и заявка возвращается в родительский процесс (exitFinished);

– если время ожидания заявки в очереди (seize, inWork) превышает установленные нормативы, она отмечается как упущенная и покидает процесс через выход exitLost.

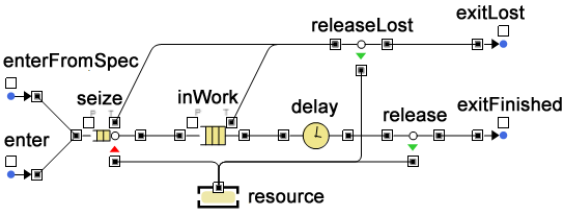


Рис. 2. Процесс обработки заявки: диаграмма класса Specialist

В случае, если специалист полностью загружен (нет свободных ресурсов в пуле resource) и очередь задач на обработку превышает критическое значение (параметр CriticalQueueSize), агент переходит в состояние «перегружен» и пытается передать задачи из очереди свободным агентам. Если свободный агент имеет необходимую компетенцию (параметр competence) и желание принять задачу (вероятность приема задачи равна 0,8), то она покидает очередь перегруженного агента и встает в очередь на обслуживание свободным агентом. Таким образом обеспечивается равномерное распределение задач между специалистами.

Для определения оптимальных значений параметров модели использовался оптимизатор OptQuest, встроенный в систему AnyLogic. Комбинируя эвристики, нейронные сети и математическую оптимизацию, OptQuest позволяет при заданных параметрах (табл. 1) находить оптимальные значения переменных модели ($g_{ij}(T)$ – количество специалистов j-й квалификации), соответствующие максимуму целевой функции $P(T)$.

Таблица 1

Исходные параметры модели

Параметр	Описание	Значение
j	Количество различных квалификаций специалистов	3
q	Количество обслуживаемых проектов	48
T	Моделируемый период времени, рабочих часов	200

OptQuest использует результаты поиска для самообучения, что позволяет выполнять интеллектуальный поиск следующего набора альтернатив. Если альтернатива в пространстве поиска не соответствует определенным пользователем ограничениям, она автоматически исключается и исследуются другие варианты, которые с большей вероятностью удовлетворяют требованиям.

После завершения процесса оптимизации процедура OptQuest в виде графика «итерация–прибыль» (рис. 3) отображает варианты решения оптимизационной задачи и выделяет лучший из них. По оси абсцисс отложена последовательность шагов процесса поиска решения с запуском соответствующей имитационной модели. На рисунке текущие результаты изображены в виде точек, а процесс их улучшения – сплошной линией.

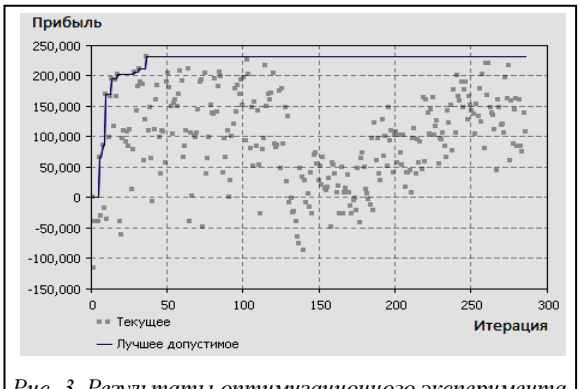


Рис. 3. Результаты оптимизационного эксперимента

За основу стратегии выбора альтернатив взяты увеличение количества специалистов квалификации «1» (S_1) и снижение количества специалистов квалификации «3» (S_3). Ввиду отсутствия роста значений целевой функции системой был выбран альтернативный подход к подбору оптимизируемых параметров – уменьшение S_1 и увеличение S_3. Это не дало положительных результатов, а лишь привело к снижению значений целевой функции (итерации N_k=100, …, 150), так как специалисты квалификации «3» являются наименее востребованными. Система вновь сменила стратегию подбора параметров альтернатив (итерации N_k=150–290, увеличение S_1 и S_2 при минимальном S_3). На графике наблюдается уверенный рост значений целевой функции, дисперсия достаточно мала – система определила критические параметры (S_1 и S_2 при минимальном S_3) и начала их варьирование в узком диапазоне. Однако найденный ранее максимум показателя «прибыль» ($P(T)$=231320) не был превышен. Ввиду отсутствия предпосылок роста и начала снижения (итерации N_k=272–290) значений целевой функции поиск альтернатив был прекращен.

Результаты выполнения оптимизационного эксперимента позволяют сделать вывод, что при заданных параметрах (табл. 1) оптимальными являются значения, приведенные в таблице 2.

Таблица 2

Результаты моделирования

Переменная	Оптимальное значение
S_1 – количество специалистов квалификации «1»	11
S_2 – количество специалистов квалификации «2»	6
S_3 – количество специалистов квалификации «3»	1
$P(T)$ – получаемая прибыль, руб.	231 320

Применение разработанной модели дает возможность компании наиболее рационально выстраивать свои бизнес-процессы и использовать существующие ресурсы, оперативно реагировать на любые изменения в динамично развивающейся среде, повышать эффективность деятельности консалтинговой компании путем изменения состава специалистов.

Литература

1. Попков Т.В. Многоподходное моделирование: практика использования // Имитационное моделирование. Теория и практика: сб. докл. IV Всеросс. науч-практич. конф. СПб: ОАО «ЦТСС», 2009. Т. 1. С. 62–67.

2. Борщев А.В. От системной динамики и традиционного ИМ – к практическим агентным моделям: причины, технология, инструменты. URL: http://www.gpss.ru/paper/borshevarc.pdf (дата обращения: 29.03.2011).

УДК 517.977

ПАРАЛЛЕЛЬНЫЙ ПРОГРАММНЫЙ КОМПЛЕКС РЕШЕНИЯ НЕГОЛОНОМНЫХ ЗАДАЧ УПРАВЛЕНИЯ

(Работа выполнена в рамках ФЦП «Исследования и разработки по приоритетным направлениям развития научно-технологического комплекса России на 2007–2013 годы», ГК № 07.514.11.4033)

А.П. Маштаков
(Институт программных систем им. А.К. Айламазяна РАН, г. Переславль-Залесский, alexey.mashtakov@gmail.com)

Описан метод управления нелинейными системами с линейным управлением на основе нильпотентной аппроксимации. Представлен алгоритм приближенного решения конструктивной задачи управления пятимерными системами такого вида в классах кусочно-постоянных и оптимальных управлений для аппроксимирующей системы. Приведенный алгоритм был положен в основу параллельного программного комплекса MotionPlanning235, предназначенного для решения поставленной задачи.

Ключевые слова: *нильпотентная аппроксимация, оптимальное управление, параллельные алгоритмы и программы.*

Перевод механической системы в требуемую конфигурацию является одной из важнейших задач робототехники и инженерии: это вывод спутника на орбиту, управление мобильным роботом, вращение твердого тела в руке робота-манипулятора и др. В математике подобная задача формализуется как конструктивная задача управления.

Имеется динамика механической системы, описываемая системой дифференциальных уравнений с функциональными параметрами (управления). Требуется найти управляющие воздействия из заданного класса функций, при применении которых система переходит из начального в заданное целевое состояние.

В работе рассматривается двухточечная задача управления следующего вида:

$$\dot{q} = u_1(t)\, X_1(q) \; + \; u_2(t)\, X_2(q), \qquad (1)$$

$$q(0) \; = \; q_0, q(T) \; = \; q_1, \qquad (2)$$

где пространство состояний $q \in Q$ – это связное пятимерное гладкое многообразие, $\dim(Q)=5$; управление принимает значения на двухмерной плоскости, $(u_1,\ u_2) \in \Re^2$; гладкие векторные поля $X_1(q)$, $X_2(q)$ удовлетворяют условию полного ранга [1] на многообразии Q. Требуется найти управление $u(t)=(u_1(t),\ u_2(t))$, переводящее систему (1) за время $T>0$ из начального состояния q_0 в терминальное q_1 с любой ранее заданной точностью $\varepsilon>0$, то есть в такое состояние $\tilde{q}_1$, что $dist(q_1, \tilde{q}_1) < \varepsilon$, где $dist$ – расстояние на многообразии Q (например, если $Q=\Re^5$, то $dist(q_1, \tilde{q}_1) = \sqrt{\sum_{i=1}^{5} (q_1^i - \tilde{q}_1^i)^2}$).

Системы вида (1) характеризуются тем, что размерность пространства управлений меньше размерности пространства состояний $\dim(\Re^2)=2<\dim(Q)=5$, однако любые точки пространства состояний могут соединяться траекторией системы. В теории управления такие системы называются вполне неголономными. Нелинейная система (1), линейно зависящая от управлений, количество которых меньше размерности пространства состояний, характеризуется тем, что возможность передвижения по различным направлениям неодинакова. Величина смещения в направлении полей $X_1(q)$, $X_2(q)$ за малое время t есть $O(t)$, в направлении коммутатора $X_3(q)=[X_1,\ X_2](q)$ есть $O(t^2)$, в направлении коммутаторов более высокого порядка $X_4(q)=[X_1,\ X_3](q)$ и $X_5(q)=[X_2,\ X_3](q)$ есть $O(t^3)$. В силу этой анизотропии пространства состояний задача управления для таких систем весьма нетривиальна.

В последнее время конструктивная задача управления активно изучается в нелинейной теории управления. Удовлетворительное решение она имеет лишь для некоторых специальных классов систем [2]. Однако пятимерные системы общего положения с двумя управлениями имеют вектор роста (2, 3, 5), потому эти результаты неприменимы к таким системам.

В данной работе представлен способ отыскания приближенного решения задачи (1)–(2), основанный на построении нильпотентной аппроксимации. Идея метода в том, что исходная нелинейная система заменяется приближенной нильпотентной системой, для которой точно решается задача управления. Затем найденные управления подставляются в исходную систему. Если состояние, достигнутое после применения найденного управления, отличается от желаемого в пределах допустимой погрешности, задача считается решенной, иначе процедура повторяется. Управления, точно решающие нильпотентную систему, дают приближенное решение исходной задачи управления в малой окрестности целевой точки. Метод нильпотентной аппроксимации применим к задачам управления общего вида; существенно только умение решать задачу управления для нильпотентной аппроксимации. Этот метод предложен в [3].

В данной работе рассматривается задача управления нелинейными пятимерными системами общего вида, линейно зависящими от двухмерного управления. Конкретными примерами систем

такого вида, которые в робототехнике имеют самостоятельное значение из-за своего прикладного значения, являются система, моделирующая качение шара по плоскости без прокручивания и проскальзывания, и система, моделирующая движение машины с двумя прицепами по плоскости.

Алгоритм поиска приближенного управления на основе вычисления траекторий аппроксимирующей системы и его реализация программным комплексом (ПК) MotionPlanning235. Итерационный алгоритм приближенного решения задачи (1)–(2) основан на локальном приближении исходной нелинейной системы нильпотентной канонической системой (3), для которой задача управления решается точно на каждой итерации.

Итак, имеется исходная система в начальном состоянии q_0. Требуется найти управление, переводящее систему в конечное состояние q_1 с предварительно заданной точностью ε. Поиск приближенного решения осуществляется следующим образом.

1. В окрестности конечной точки q_1 строится аппроксимирующая система, для которой точно решается задача управления.

2. Найденные управления подставляются в исходную систему. Если достигнутое состояние не попадает в ε-окрестность конечного состояния, значит, нужная точность не достигнута. Алгоритм повторяется снова для исходной системы, где в качестве начального состояния выбирается состояние, достигнутое на предыдущей итерации алгоритма, иначе вычисления прекращаются.

В среде Mathematica 8 (*http://www.wolfram.com/mathematica/*) был реализован параллельный ПК **MotionPlanning235** (см. рис. 1) решения задачи (1)–(2). Он является дополнительным пакетом для системы Wolfram Mathematica (MotionPlanning235.m) и состоит из следующих основных модулей:

– **NilpotentApproximation235** строит нильпотентную аппроксимацию NA (см. (3)) системы (1) и возвращает замену переменных τ=A°Φ, в которых NA имеет канонический вид;

– **CPControl235** решает задачу (1)–(2) в классе кусочно-постоянных управлений;

– **OptimalControl235** решает задачу (1)–(2) в классе оптимальных управлений.

Модуль NilpotentApproximation235 построения нильпотентной аппроксимации. Локальное приближение управляемой системы другой, более простой, широко используется в теории управления. Обычно в качестве локальной аппроксимации используется линеаризация управляемой системы. Однако для линейных по управлению систем вида (1) линеаризация дает слишком грубое приближение. Так как размерность управления меньше размерности состояния, линеаризация не может быть вполне управляемой. Естественной заменой линейной аппроксимации в этом случае является нильпотентная аппроксимация – наиболее простая система, сохраняющая структуру управляемости исходной системы (в частности, сохраняется такой важный инвариант, как вектор роста).

В данной работе использован алгоритм вычисления нильпотентной аппроксимации для линейных по управлению систем, предложенный в [2]. Этот алгоритм был конкретизирован для систем вида (1), а именно, вычислена замена переменных для перехода в привилегированные координаты $A{:}q\to z$, а векторные поля нильпотентной аппроксимации $\hat{X}_1(z)$, $\hat{X}_2(z)$ в привилегированных координатах z_i, i=1, …, 5, были выражены через векторные поля исходной системы $X_1(z)$, $X_2(z)$ и их производные.

Кроме того, алгоритм Беллаиша дополнен следующим образом: выполняется переход в систему координат y (замена переменных $\Phi{:}z\to y$), в которой нильпотентная аппроксимация имеет канонический вид

$$\begin{cases} \dot{y}_1 = u_1, \\ \dot{y}_2 = u_2, \\ \dot{y}_3 = \dfrac{1}{2}\left(y_1u_2 - y_2u_1\right), \\ \dot{y}_4 = \dfrac{1}{2}\left(y_1^{\,2} + y_2^{\,2}\right)u_2, \\ \dot{y}_5 = -\dfrac{1}{2}\left(y_1^{\,2} + y_2^{\,2}\right)u_1, \end{cases} \qquad (3)$$

а граничные условия представлены как

$$y(0)=y^0,\ y(T)=(0,\ 0,\ 0,\ 0,\ 0). \qquad (4)$$

Модуль CPControl235 решения задачи в классе кусочно-постоянных управлений. В ПК **MotionPlanning235** реализован модуль **CPControl235** решения задачи управления нильпотентной канонической системой (3) с граничными условиями (4) в классе кусочно-постоянных функций. При этом время можно перепараметризовать так, что T=1. Требуется найти управления $(u_1(t),\ u_2(t))$ в классе кусочно-постоянных функций на отрезке $t\in[0,\ 1]$, переводящие систему из произвольного начального состояния $y^0=\Re^5$ в начало координат. Доказано следующее утверждение.

Предложение. Для решения задачи управления (3)–(4) достаточно управлений с тремя точками переключения:

$$u_{1,2} = \begin{cases} \alpha_{1,2}, & t\in[0,\ 0,25), \\ \beta_{1,2}, & t\in[0,25,\ 0,5), \\ \gamma_{1,2}, & t\in[0,5,\ 0,75], \\ \delta_{1,2}, & t\in(0,75,\ 1]. \end{cases}$$

Коэффициенты α, β, γ, и δ определяются из системы пяти алгебраических уравнений с восемью неизвестными. Получается трехпараметрическое семейство решений, причем для любого начального состояния $y^0=\Re^5$ существует способ зафиксировать свободные параметры так, чтобы

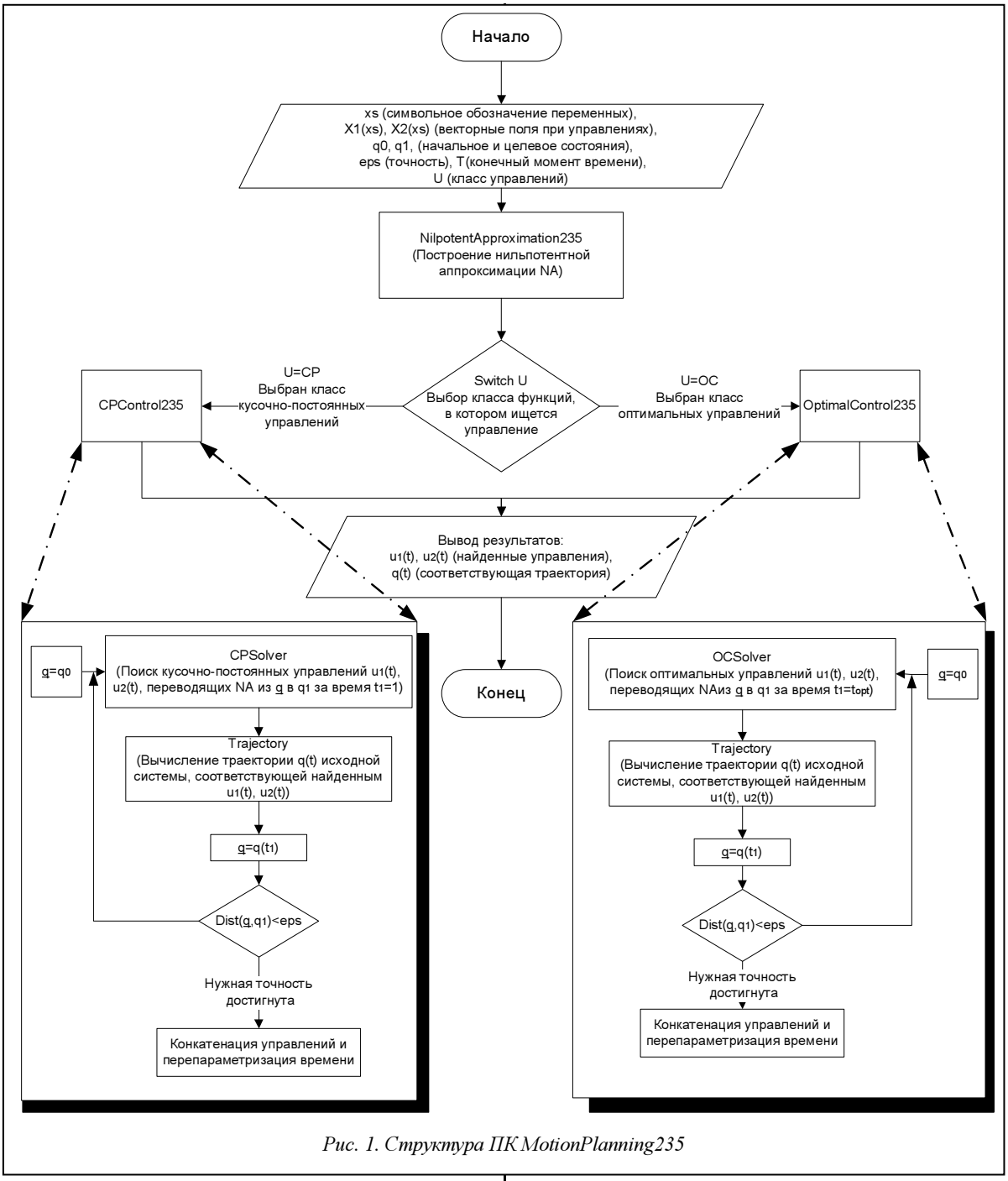


Рис. 1. Структура ПК MotionPlanning235

получалось решение без особенностей. В модуле **CPControl235** фиксация свободных параметров осуществляется так, чтобы в соответствующей траектории не было больших отклонений, а именно, используется критерий $\max|u_i| \to \min$.

Модуль OptimalControl235 решения задачи в классе оптимальных управлений. Для управляемой системы (3) с граничными условиями (4) рассматривается задача оптимального управления с естественным интегральным критерием (функционалом субримановой длины)

$$\int_0^T \sqrt{u_1^2 + u_2^2}\, dt \to \min. \qquad (5)$$

Эта задача известна в теории управления как обобщенная задача Дидоны, а в субримановой геометрии – как субриманова задача с вектором роста (2, 3, 5). Она имеет богатую историю и была детально теоретически изучена в [4]. Основным результатом теоретических исследований явилось описание структуры экспоненциального отображения и первых времен Максвелла вдоль экстремальных траекторий. На основе этого задача оптимального управления (3)–(5) была сведена к задаче решения пяти алгебраических уравнений в неэлементарных функциях с пятью неизвестными. Явно решить эту систему практически невозмож-

но из-за сложности получившихся формул, поэтому предложено использовать численные методы.

Приведем некоторые результаты из статьи [4], необходимые для дальнейшего изложения. Семейство экстремальных траекторий в задаче оптимального управления параметризуется экспоненциальным отображением exp, переводящим пару (вектор сопряженных переменных, время) в соответствующую точку экстремальной траектории. Прообраз L и образ M экспоненциального отображения exp: $L \to M$ разбиваются поверхностями разреза на четыре области ($L = \bigcup_{i=1}^{4} L_i$ в прообразе и $M = \bigcup_{i=1}^{4} M_i$ в образе), таких, что L_i переходит в M_i и экспоненциальное отображение является диффеоморфизмом на этих областях. В свою очередь, каждая область L_i разбивается на непересекающиеся множества C_1 и C_2, в которых экспоненциальное отображение задается разными формулами. Задача построения оптимального синтеза состоит в обращении экспоненциального отображения. При этом разбиение L на L_i и M на M_i известно, а разбиение на C_i неизвестно. В задаче (3)–(5) была найдена двухмерная группа симметрий, состоящая из вращений и растяжений. Факторизация задачи по этой группе уменьшает размерность задачи с пяти до трех. Получается система из трех уравнений с тремя неизвестными:

$$\begin{cases} P(u,v,k) = P_1, \\ Q(u,v,k) = Q_1, \\ R(u,v,k) = R_1. \end{cases} \qquad (6)$$

Итак, для построения оптимального синтеза в задаче (3)–(5) требуется решить систему (6) относительно u, v, k при заданной правой части P_1, Q_1, R_1. Заметим, что функции $P(u, v, k)$, $Q(u, v, k)$ и $R(u, v, k)$ выражены в эллиптических функциях Якоби и эллиптических интегралах первого и второго рода и имеют сложную аналитическую запись. Известно, что при любой правой части (за исключением особых множеств меньшей размерности) система (6) имеет единственный корень.

Для решения поставленной задачи был разработан модуль **OptimalControl235**. Схема его работы представлена на рисунке 2. Пользователь легко может организовать параллельное вычисление корня на нескольких ядрах процессора. Для этого требуется указать системе Mathematica 8 (функция недоступна в более ранних версиях) количество ядер и изменить при необходимости функцию **SolveHen** (выбрать, какие алгоритмы решения системы – AlgorithmA, AlgorithmB, AlgorithmC, AlgorithmD – будут выполняться одновременно и сколько именно). Кроме стандартного метода Ньютона (AlgorithmA) и метода хорд (AlgorithmB), пользователю предлагаются гибридные методы (AlgorithmC, AlgorithmD). Проведенное обширное тестирование показало, что в большинстве случа-

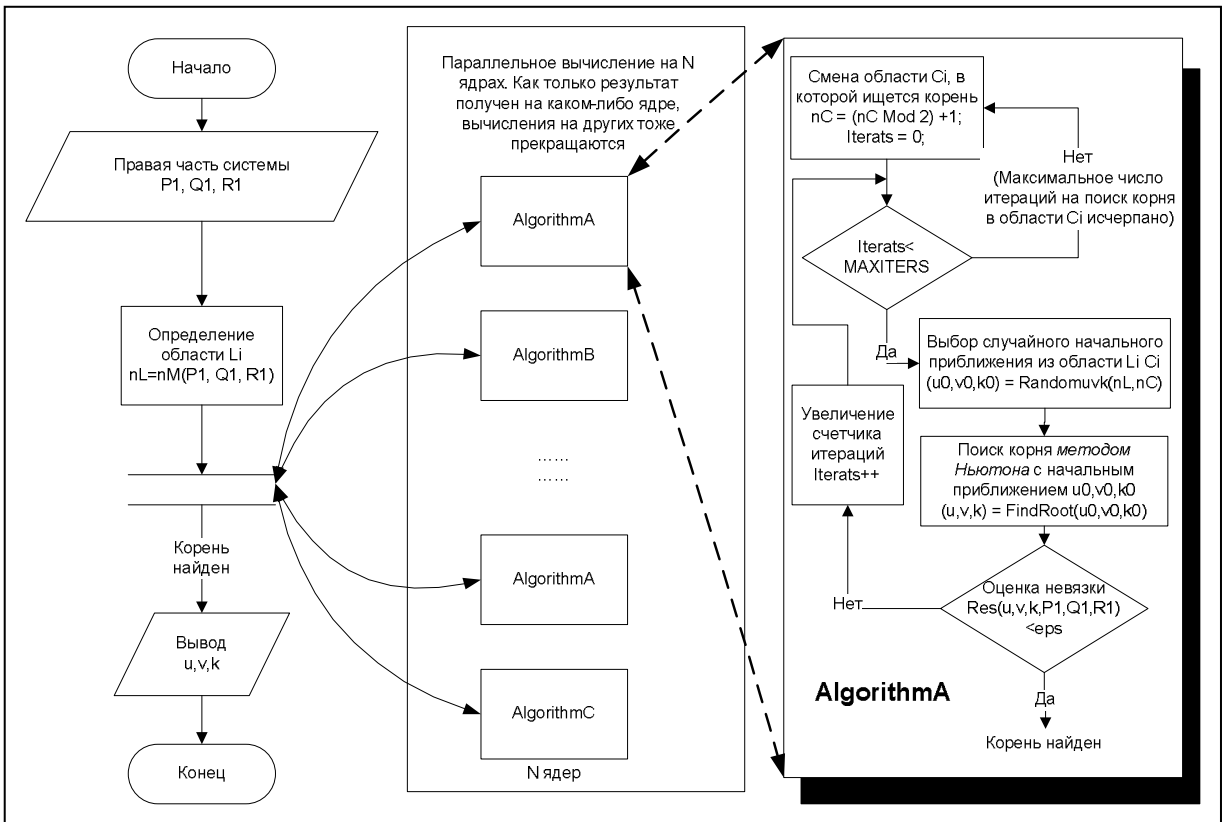


Рис. 2. Схема работы модуля Optimalcontrol235

ев первым результат выдает AlgorithmA или AlgorithmB, хотя и бывают ситуации, когда гибридные методы справляются с задачей быстрее. Поэтому по умолчанию предлагается делить доступные ядра поровну между AlgorithmA и AlgorithmB. Отметим, что одновременный запуск одного и того же алгоритма на разных ядрах имеет смысл, так как во всех алгоритмах используется генератор случайных начальных приближений. Так как основное время вычисления занимает выбор удачного начального приближения, использование нескольких ядер дает существенное ускорение.

Дополнительные функции ПК Motionlanning235. Помимо основных модулей **Nilpotent-Approximation235**, **CPControl235** и **OptimalControl235,** решающих задачу (1)–(2), пользователю предоставляются некоторые дополнительные инструменты для представления результатов и слежения за процессом вычислений. Речь идет о следующих функциях:

1. **TrajectoryNatPar[X1, X2, {u1,u2,T}, q0, xs]** возвращает траекторию $q(t)$ системы (1), где $t\in[0, T]$, соответствующую управлениям $\{u1, u2\}$, выходящую из точки $q0$.

2. **PlotTrajectoryNatPar[trajectory, q0, q1, T]** строит разным цветом графики пяти компонент заданной траектории trajectory(t) за время $t\in[0, T]$, а также отмечает на нем заданные состояния $q0$ и $q1$. Эта функция может использоваться для визуальной оценки достигнутого состояния и найденных траекторий. Пользователь может включить режим работы ПК, в котором графики найденных траекторий будут выводиться на экран на каждой итерации.

3. **PlotTrajectoryOtklNatPar[trajectory, q0, q1, T]** строит разным цветом графики отклонений пяти компонент заданной траектории trajectory(t) за время $t\in[0, T]$ от состояния $q1$.

4. **AnimateCar[trajectory, T, q0, q1, N, {{xmin, xmax}, {ymin, ymax}}]** строит анимацию движения машины с двумя прицепами при движении по траектории trajectory(t) за время $t\in[0, T]$ из состояния $q0$. Область, по которой движется машина с прицепами, ограничивается прямоугольником $\{\{x\min, x\max\},\{y\min, y\max\}\}$. В результате на жестком диске создается последовательность из N кадров. Последовательность кадров – это упорядоченный набор изображений .png, который впоследствии легко может быть собран в видеофайл стандартными утилитами. Размер изображения является параметром, доступным пользователю (по умолчанию 1024×768 пикселей). Помимо положения машины и прицепов, на каждый кадр пунктиром наносится состояние $q1$. Функция предназначена как для самостоятельного использования (для моделирования системы «машина с двумя прицепами» и ее исследования), так и для проверки решений, найденных с помощью ПК **MotionPlanning235**.

5. **AnimateBall[trajectory, T, q0, q1, N]** визуализирует качение без прокручиваний и проскальзываний сферы по траектории trajectory(t) (аналогично функции **AnimateCar).**

Пример использования ПК MotionPlanning235. Продемонстрируем работу ПК MotionPlaning235 на задаче о перемещении по плоскости машины с двумя прицепами. Состояние системы описывается пятью координатами $(x, y, \theta, \varphi_1, \varphi_2)\in Q=\Re^2\times S^3$. Здесь $(x, y)\in\Re^2$ – координаты центра машины на плоскости; θ – угол, задающий ориентацию машины на плоскости; φ_1 – угол, задающий положение первого прицепа относительно машины; φ_2 – угол, задающий положение второго прицепа относительно первого прицепа. Динамика системы имеет вид

$$\begin{cases} \dot{x} = \cos(\theta)u_1, \\ \dot{y} = \sin(\theta)u_2, \\ \dot{\theta} = u_2, \\ \dot{\varphi}_1 = -\sin(\varphi_1)u_1 + (-1-\cos(\varphi_1))u_2, \\ \dot{\varphi}_2 = (\sin(\varphi_1-\varphi_2)+\sin(\varphi_1))u_1 + \\ \qquad +(\cos(\varphi_1-\varphi_2)+\cos(\varphi_1))u_2. \end{cases} \quad (7)$$

Зададим начальное состояние $q_0=(0, 0, \pi/4, \pi/4, -\pi/4)$ и целевое $q_1=(-0{,}252, -0{,}339, 1{,}085, 0{,}514, -1{,}281)$, в которое система должна перейти с точностью $\varepsilon=10^{-3}$.

На рисунках 3 и 4 приведены результаты работы модулей **CPControl235** и **OptimalControl235**. В случае применения кусочно-постоянных управлений алгоритму потребовалось шесть итераций для достижения требуемой точности, а для опти-

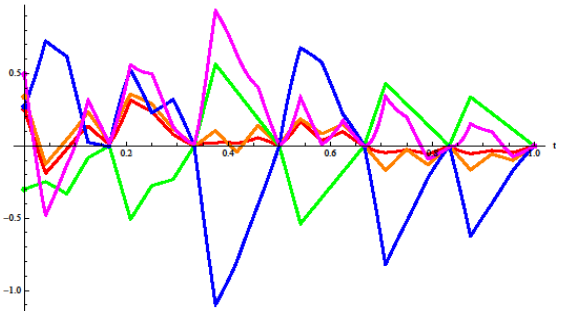


Рис. 3. Результат работы CPControl235

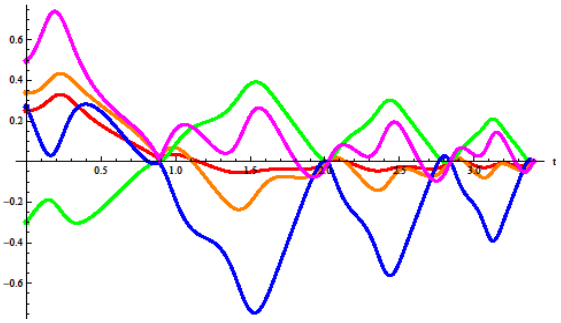


Рис 4. Результат работы OptimalControl235

мальных управлений – четыре. При использовании кусочно-постоянных управлений система совершает больший маневр (траектория имеет большую амплитуду отклонений от целевого состояния).

В заключение следует отметить, что рассматриваемый в работе метод приближенного решения двухточечной задачи управления (1)–(2) был реализован в параллельном ПК **MotionPlaning235.** Комплекс испытан на двух прикладных задачах (задача о качении шара по плоскости и задача управления машиной с двумя прицепами). В случаях, когда граничные условия не были слишком далеки, комплекс успешно решал поставленную задачу управления. При далеких граничных условиях алгоритм не сходился, что соответствует теоретическому обоснованию метода (нильпотентная аппроксимация является локальным приближением исходной системы). В будущем планируется расширить функционал ПК для решения глобальной задачи управления путем ее сведения к последовательности локальных задач. В настоящее время ПК **MotionPlaning235** является удобным и надежным средством решения локальной задачи (1)–(2).

Литература

1. Аграчев А.А., Сачков Ю.Л. Геометрическая теория управления. М.: Физматлит, 2005. 391 с.

2. Bellaiche A. The tangent space in sub-Riemannian geometry // Sub-Riemannian Geometry, A. Bellaiche and J.J. Risler, Eds. Basel, Swizerland: Birkh auser, 1996, pp. 1–78.

3. Laferriere G., Sussmann H.J. A differential geometric approach to motion planning // Nonholonomic Motion Planning, Zexiang Li and J.F. Canny Eds. Basel, Swizerland: Kluwer, 1992.

4. Сачков Ю.Л. Полное описание стратов Максвелла в обобщенной задаче Дидоны // Мат. сб. 2006. Т. 197. № 6. С. 111–160.

5. Laumond J.P. Lecture Notes in Control and Information Science. Springer. 1998. № 229. 343 с.

6. Murray R.M., Sastry S.S. Steering controllable systems, Proc. 29th IEEE Conf. Dec. and Control. Honolulu, Hawaii, 1990, pp. 408–412.

УДК 004.942

МОДЕЛИРОВАНИЕ РАСТВОРЕНИЯ ТВЕРДЫХ ТЕЛ С ПОМОЩЬЮ КЛЕТОЧНЫХ АВТОМАТОВ

(Работа выполнена при поддержке Министерства образования и науки РФ, ГК № 16.552.11.7046)

Н.В. Меньшутина, д.т.н.; С.И. Иванов; Д.Д. Шипилова
(Российский химико-технологический университет им. Д.И. Менделеева, г. Москва, patephon2009@yandex.ru)

Статья посвящена моделированию растворения твердых тел с помощью вероятностных клеточных автоматов. Приведено описание математической модели растворения и рассмотрен алгоритм работы клеточного автомата, реализующего модель. Проведено сравнение экспериментальных (кинетика растворения таблетки аскорбиновой кислоты) и расчетных данных.

Ключевые слова: *клеточные автоматы, растворение, моделирование, параллельные вычисления, фармацевтическая промышленность, многокомпонентные смеси, диффузия.*

Высокие темпы развития пищевой, фармацевтической и косметической промышленности в мире обусловливают необходимость решения разнообразных задач моделирования процессов в этих предметных областях. Одним из важных является процесс растворения твердых тел в различных растворителях при разных условиях.

При растворении твердых тел стоит учитывать ряд факторов, которые могут повлиять на ход процесса растворения, например: твердое тело может состоять из нескольких компонентов, имеющих разную растворимость; оно может быть покрыто оболочкой, затрудняющей растворение, а также иметь несимметричную форму; в состав твердого тела могут входить вещества, способные увеличивать растворимость других входящих в него веществ.

На текущий момент уже существуют разнообразные математические модели растворения твердых тел на основе дифференциальных уравнений (Weibull, Hixon–Crowell, Korsemeyer–Peppas и т.д.).

Данная статья посвящена описанию модели растворения твердых тел на основе вероятностного клеточного автомата. При работе над моделью авторы попытались учесть все факторы, перечисленные выше.

Описание модели клеточного автомата

В основе модели лежит вероятностный клеточный автомат [1]. Клетки автомата образуют прямоугольное поле размером $N \times N$ клеток. Модель имеет следующие допущения:

– система представляется в виде поля, состоящего из квадратных клеток;

– каждая клетка имеет четыре соседние клетки;

– поле может быть замкнутым или открытым;

– клеточный автомат является синхронным, то есть считается, что все изменения за одну итерацию происходят в одно и то же время;

– клетка описывается тремя характеристиками – типом вещества, его количеством, находящимся в данной клетке, и агрегатным состоянием («жидкость» или «твердое вещество»);

– в каждой клетке может находиться только одно вещество;

– состояние клетки зависит только от соседних клеток;

– при снижении количества вещества в клетке до уровня, соответствующего концентрации насыщенного раствора этого вещества, состояние клетки переходит в «жидкость»;

– при увеличении количества вещества в клетке до уровня, превышающего концентрацию насыщенного раствора этого вещества, состояние клетки переходит в «твердое вещество»;

– диффузия может происходить только между клетками, имеющими состояние «жидкость».

Клеточный автомат работает итеративно. В начале итерации рассчитываются новые значения клеток, которые принимают значения «раствор вещества» и «твердое вещество». Начиная с клетки с индексом (0, 0), последовательно происходит проверка значения, которое содержится в клетке.

Если значение от 0 до $C_{насыщ.}$, значит, клетка имеет состояние «раствор вещества». Для клеток такого типа действуют следующие правила.

Каждая соседняя клетка с состоянием «растворитель» уменьшает значение в клетке на $C_p^{\,1}$ (зависящее от коэффициента диффузии, линейного размера клетки и времени одной итерации) за одну итерацию, при этом клетка с состоянием «растворитель» переходит в состояние «раствор вещества» со значением $C_p^{\,1}$.

Каждая соседняя клетка с состоянием «твердое вещество» увеличивает значение в клетке на $C_m^{\,1}$ (зависящее от коэффициента растворения, линейного размера клетки и времени одной итерации) за одну итерацию, при этом значение в клетке с состоянием «твердое вещество» уменьшается на $C_m^{\,1}$. При достижении значения, большего $C_{насыщ.}$, состояние клетки принимается как «твердое вещество».

Если значение в клетке от $C_{насыщ.}$ до $C_{\max}$, значит, клетка имеет состояние «твердое вещество». Для клеток такого типа действует следующее правило.

Каждая соседняя клетка с состоянием «растворитель» уменьшает значение в клетке на $C_p^{\,2}$ (зависящее от коэффициента растворения, линейного размера клетки и времени одной итерации) за одну итерацию, при этом клетка с состоянием «растворитель» переходит в состояние «раствор вещества» со значением $C_p^{\,2}$. При достижении значения, меньшего или равного $C_{насыщ.}$, состояние клетки принимается как «раствор вещества».

Таким образом, суммарные значения по каждому из веществ в клетках с состоянием «твердое вещество» и «раствор вещества» в процессе расчета не изменяются. Коэффициенты $C_{насыщ.}$, $C_m^{\,1}$, $C_p^{\,1}$, $C_p^{\,2}$ определяются исходя из физико-химических величин (коэффициенты диффузии и растворения) и параметров клеточного автомата (линейный размер клетки, время одной итерации). Растворение в данной клеточно-автоматной модели по физическому смыслу схоже с дифференциальным уравнением растворения твердого тела: $\frac{dM}{d\tau} = -kF\left(C_{насыщ.} - C\right)$, где M – изменение массы твердого вещества; k – коэффициент растворения; F – поверхность растворения; $C_{насыщ.}$ – концентрация насыщенного раствора.

После окончания расчета значений для клеток с состоянием «твердое вещество» и «раствор вещества» начинается расчет диффузии. Происходит случайный выбор p процентов пар клеток «раствор вещества»–«растворитель», и в каждой паре значения клеток меняются местами. Таким образом, этап расчета диффузии позволяет предотвратить ситуацию, когда вокруг клеток «твердого вещества» образуется пограничная зона с клетками «раствора вещества».

После расчета диффузии начинается следующая итерация. Количество итераций определяет пользователь в начальных настройках. На рисунке 1, где изображена часть поля клеточного автомата, показано изменение состояния автомата через одну итерацию. Как видно из рисунка, клетки с координатами (2, 3) и (3, 3) имели состояние «твердое вещество» (*Т*) и перешли в состояние «раствор вещества» (*Н*) (изменение произошло при расчете растворения), а клетки (0, 1) и (0, 2) поменяли свои состояния (расчет диффузии).

Как было показано в допущениях к модели, для

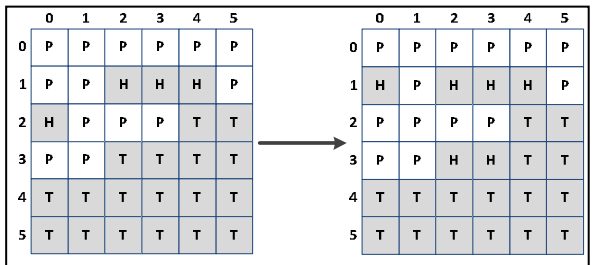


Рис. 1. Изменение состояния клеток автомата

данного клеточного автомата предусмотрена работа в режимах с открытыми и замкнутыми границами. При работе в режиме открытых границ в конце каждой итерации границы клеточного автомата (те клетки, которые имеют координату X или координату Y, равную 0 или N) принимают значение «растворитель». Это позволяет моделировать систему растворения твердого тела в объеме жид-

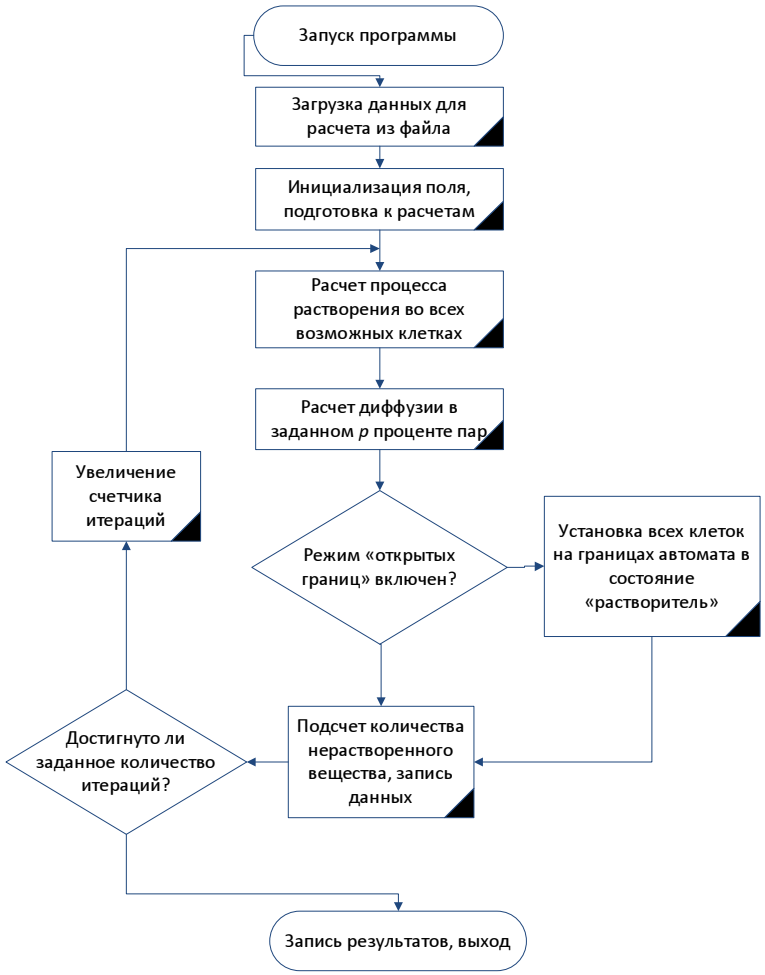


Рис. 2. Блок-схема алгоритма работы клеточного автомата

кости, многократно превышающем объем тела. При работе автомата в режиме замкнутых границ соседней клеткой с клеткой границы считается клетка на противоположной границе поля автомата, то есть для клеток (A, 0) и (0, B) соседними являются (A, N-1) и (N-1, B) соответственно.

На рисунке 2 приведена блок-схема алгоритма работы программы, реализующей описанный выше клеточный автомат.

Результаты расчета

После разработки клеточного автомата были выполнены расчеты с различными коэффициентами $C_{насыщ.}$, C_m^1, C_p^1, C_p^2, p для сравнения с экспериментальными данными.

В качестве экспериментальных данных взяты значения концентраций в различное время при растворении таблетки аскорбиновой кислоты. В двух различных экспериментах были выбраны разные обороты лопастной мешалки. На рисунке 3 показаны результаты расчета программы в графическом виде. На фоне растворителя черным цветом отображены ячейки, содержащие нерастворенную аскорбиновую кислоту, серым – раствор аскорбиновой кислоты.

После расчетов было проведено сравнение расчетных и экспериментальных данных. На рисунке 4 приведены расчетные графики и экспериментальные точки. Как видно из результатов, клеточно-автоматная модель хорошо описывает растворение твердого тела в растворителе. Отклонение расчетных данных от экспериментальных составляет не более 5 %.

Программная реализация

Клеточный автомат реализован на языке программирования C# 4.0 на платформе Microsoft .NET Framework 4.0. В программе использованы алгоритмы распараллеливания расчетов Microsoft Parallel [2], что позволяет оптимально использовать всю суммарную вычислительную мощность центральных процессоров компьютера, на котором производится расчет. Кроме реализации клеточного автомата, была разработана программа, позволяющая пользователям задавать параметры с помощью графического интерфейса. На данный момент максимальный размер линейного поля клеточного автомата составляет 1 500 клеток при расчете на компьютере с объемом оперативной памяти 4 GB.

Дальнейшие работы. Как уже отмечалось, существуют такие проблемы, как многокомпонентность, различные покрытия твердого тела и др. Данная модель позволяет учитывать все вышеперечисленные факторы путем введения новых состояний клеток и, соответственно, новых правил для этих состояний. В ходе дальнейших исследований предполагается провести ряд экспериментов и представить результаты моделирования растворения для различных твердых тел. Кроме того, планируется реализация данного клеточного автомата на языке программирования C/C++ с использованием технологии nVidia CUDA, что позволит

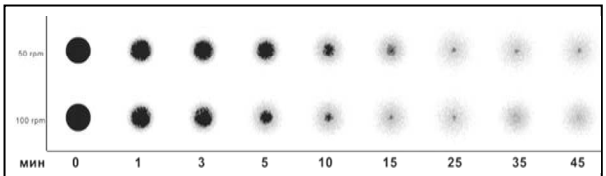


Рис. 3. Графические результаты расчета программы

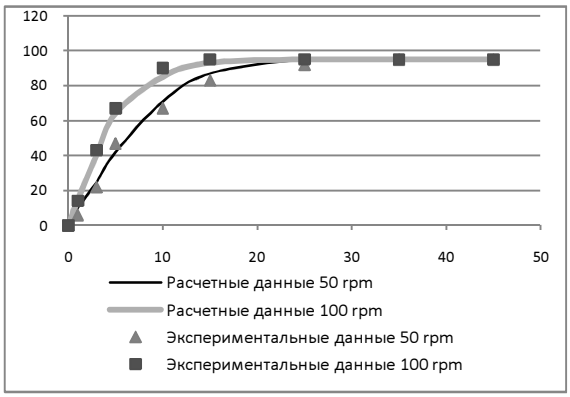


Рис. 4. Сравнение расчетных и экспериментальных данных

проводить параллельные вычисления с использованием ядер графического адаптера компьютера. Применение технологии nVidia CUDA значительно увеличит скорость расчета и даст возможность проводить расчеты полей линейным размером более 10 000 клеток. Применение данной технологии распараллеливания позволит перейти от двухмерной модели к трехмерной, что дает возможность моделировать растворение тел с любой формой.

В заключение отметим, что в данной статье представлены клеточно-автоматная модель растворения твердых тел на примере растворения таблетки аскорбиновой кислоты и ее программная реализация. Отмечено, что процесс растворения твердого тела зависит от многих параметров, в частности, от состава твердого тела, наличия покрытия, среды растворения, температуры среды и других. Рассмотренная клеточно-автоматная модель позволяет учесть при расчете все данные параметры, но с увеличением их количества скорость расчета по представленной модели уменьшается.

Литература

1. Тоффоли Т., Марголус Н. Машины клеточных автоматов; [пер. с англ.]. М.: Мир, 1991. 280 с.

2. URL: http://msdn.microsoft.com/en-us/library/dd460693.aspx (дата обращения: 17.09.2011).

УДК 371.693.4

ОБОСНОВАНИЕ ПРИМЕНЕНИЯ ЭЛЕКТРОННЫХ ТРЕНАЖЕРОВ ДЛЯ ОБУЧЕНИЯ ОПЕРАТОРОВ

Ю.И. Арепин, д.т.н.
(НИИ «Центрпрограммсистем», г. Тверь, arep@cps.tver.ru)

Рассматривается подход к обоснованию применения различных средств обучения для операторов сложных технических систем.

Ключевые слова: *сложная техническая система, оператор, автоматизированное рабочее место, обучающие системы.*

В современных *сложных технических системах* (СТС) в настоящее время ряд функций по управлению технологическими процессами, контролю и слежению за ними осуществляют операторы, которых можно отнести к операторам сенсомоторного вида деятельности, заключающейся в самом общем виде в просмотре (контроле, слежении за показаниями) средств отображения информации (сенсорная составляющая деятельности), принятии определенных решений (в соответствии с алгоритмом работы) и воздействии на органы управления на рабочем месте (моторная составляющая деятельности) для реализации функций управления (контроля или слежения).

Подготовка операторов такого типа обычно сопряжена с необходимостью первоначального профессионального отбора для выявления психофизиологических исходных данных каждого из них и последующего обучения будущей специальности с учетом квалификационных требований к ней. Профессиональному отбору посвящено много публикаций и методических материалов, однако каждая специальность оператора предъявляет свои требования к кандидатам на обучение. Не останавливаясь на процессе профессионального отбора, рассмотрим этап обучения уже отобранных операторов.

Будущие операторы СТС последовательно проходят теоретическое обучение, обучение на тренажерах и на реальном рабочем месте.

Формализация научной задачи выглядит следующим образом:

$$W_{\text{обуч.}}\,(\overline{V}\,/\,\overline{A}\,,\,C_{\text{обуч.}},\,T_{\text{обуч.}}) \xrightarrow[V\in\Omega]{} \max$$

при $C_{\text{обуч.}} \le C_{\text{выд.}}$, $T_{\text{обуч.}} \le T_{\text{выд.}}$, где $W_{\text{обуч.}}$ – эффективность обучения оператора; $\overline{V} = \{n_1, n_2, \ldots, n_k\}$ – варианты обучения оператора; $\overline{A} = \{a_1, a_2, \ldots, a_e\}$ – сложность алгоритмов работы оператора, зависящая от назначения СТС и распределения функций между оператором и средствами автоматизации СТС; $C_{\text{обуч.}}$ – необходимые затраты на обучение оператора для конкретной СТС; $C_{\text{выд.}}$ – выделен-

ные ресурсы для обучения; $T_{обуч.}$ – время обучения оператора для конкретной СТС; $T_{выд.}$ – располагаемое время на обучение.

Под эффективностью (качеством) обучения оператора здесь понимается степень соответствия ожидаемых (необходимых) и реально полученных результатов обучения заданным требованиям в конкретной СТС.

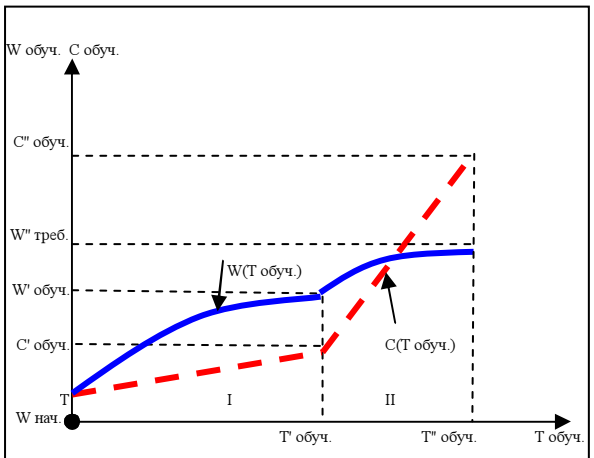
Современные обучающие системы в настоящее время реализуются на ПЭВМ, обеспечивающих имитацию информационной модели для оператора на реальном рабочем месте.

Моторная составляющая деятельности оператора может быть реализована на сенсорных панелях (они имитируют внешний вид и функции органов управления системой) или на электромеханическом имитаторе рабочего места оператора, оснащенном реальными органами управления. Последний вариант в современных условиях встречается редко из-за сложности конструктивной реализации такого имитатора, а также его значительно более высокой стоимости. Основными факторами, определяющими качество обучения операторов СТС, как известно, являются время и методы обучения, а также профпригодность оператора. Графически процесс обучения представлен на рисунке, где $W'_{обуч.}$ – уровень качественных характеристик обучения, который можно обеспечить на тренажерах-имитаторах (ПЭВМ) за время $T'_{обуч.}$; $W'_{треб.}$ – уровень качественных характеристик обучения, который достигается на реальном АРМ оператора на образце СТС за время $T''_{обуч.}$; $W_{нач.}$ – уровень начального (теоретического) обучения оператора, достигаемый в процессе теоретической подготовки оператора СТС будущей профессии (теория работы СТС, алгоритмы работы оператора на АРМ и т.д.).

Обучение на первом этапе (после начальной теоретической подготовки) осуществляется на тренажере-имитаторе, и затраты складываются из стоимости труда преподавателя, амортизационных отчислений при эксплуатации тренажера и текущих расходов (на электроэнергию, аренду помещения). Как показывает практика, эти затраты относительно невелики. В то же время возможно обучение операторов на реальном образце СТС, однако стоимость такого обучения на порядок выше, так как она складывается из стоимости эксплуатации образца СТС, стоимости труда обучающего и текущих расходов. При этом стоимость эксплуатации образца значительно больше стоимости амортизации тренажеров, так как учитывает расход технического ресурса образца СТС.

Приведем анализ процессов обучения операторов. Он включает математическую формулировку задачи обучения и построения математической модели процесса обучения.

В результате измерения (оценки) качества обучения операторов получаем отличающиеся друг от



Изменение во времени уровня обученности операторов $W(T_{обуч.})$ и стоимости обучения $C(T_{обуч.})$ для комбинации обучения на тренажерах-имитаторах (I) и реальных АРМ СТС (II)

друга значения измеряемых величин (показателей качества обучения). Как известно, результат может быть назван случайным (стохастическим), как и соответствующие величины измерений (оценок).

Таким образом, рассматриваем $W(T_{обуч.})$ как функцию времени. Математическое описание процесса обучения оператора – это описание случайного процесса, при этом переменные процессы дискретны и могут принимать только отдельные значения в некотором интервале.

Распределение накопленной вероятности для дискретной величины $W(T_{обуч.})$ обозначим символом $P(\omega,t) = P\{W(T_{обуч.}) < \omega_i\} = \sum_{i=1}^{\kappa} P(x_i, t)$, где ω – некоторое число; i – число занятий с оператором.

При этом $W(T_{обуч.}) < \omega_i$ означает, что все значения случайной величины $W(T_{обуч.})$ меньше детерминированной величины ω_i.

Выбор варианта обучения наиболее целесообразно осуществлять на основе использования известного показателя «эффективность–стоимость» [Ю.И. Арепин и др.] или «качество обучения–стоимость» путем расчета $P(\omega, t)/C_{обуч.}$ для разных вариантов обучения или их сочетания. Этот же показатель можно использовать при комбинации вариантов обучения для обоснования времени обучения с помощью современных тренажеров-имитаторов на ПЭВМ и реальных АРМ на СТС.

Таким образом, обоснование средств и способов обучения операторов СТС должно включать количественную оценку эффективности применения тех или иных средств или их разумного сочетания.

Литература

Военная экономика: управление, планирование, военно-экономическая безопасность / Ю.И. Арепин [и др.]; [под ред. А.С. Сумина, Ю.И. Арепина]. М., 1995. 183 с.

SUMMARY

INPUT LANGUAGE OF THE OBJECT-ORIENTED KNOWLEDGE BASE OF GRID-SYSTEM

***Oparin G.A.**, Ph.D.; **Feoktistov A.G.**, Ph.D.; **Vartanyan E.K.** (Institute of System Dynamics and Control Theory SB RAS, Irkutsk, oparin@icc.ru, agf@icc.ru, e.vartanyan@mail.ru)*

Abstract. In article the description language of the knowledge about experimental Grid-system is considered. The description of such knowledge is based on object-oriented model of the data.

Keywords: input language, object-oriented knowledge base, Grid-system.

GridNNN INFORMATION SYSTEM

Kryukov A.P.**, Ph.D.; **Shamardin L.V.**, Ph.D.; **Patrikeev D.O.
(Scobeltsyn Institute of Nuclear Physics, Lomonosov Moscow State University, kryukov@theory.sinp.msu.ru, shamardin@theory.sinp.msu.ru, patrikeev@theory.sinp.msu.ru)

Abstract. Russian National Nanotechnology Network grid infrastructure (GridNNN) is developed to provide access to supercomputer resources to scientists, engineers and students working in the area of nanotechnology sciences. It should enable unified transparent and secure access to computational resources connected to the NNN. One of the distinguishing key features of GridNNN is its design based on REST architectural style for grid services. This work describes the GridNNN Information System based on RESTful web services, the structure of the published information and the access security model.

Keywords: high performance computing, distributed computing, grid, GridNNN, REST, RESTful web services, information system, accounting system.

PARALLEL COMPUTING TOOLKIT FOR SOLVING BOOLEAN EQUATIONS ON MULTI-CORE PROCESSORS

***Oparin G.A.**, Ph.D.; **Bogdanova V.G.**, Ph.D.*
(Institute of System Dynamics and Control Theory SB RAS, Irkutsk, oparin@icc.ru, bvg@icc.ru)

Abstract. Enterprise principles, base functions and structural components of toolkit for automation of parallel computing of boolean equations on multi-core processors is considered. In particular, the technology of automation of boolean modeling are considered.

Keywords: parallel computing, boolean modeling, boolean equations solving.

SOFTWARE SYSTEM FOR RESEARCH OF STABILITY OF NONLINEAR DYNAMICAL SYSTEMS

***Kuznetsov A.Yu.** (Tver State University, ferus.tigris@gmail.com)*

Abstract. The complex of programs for research stability of autonomic nonlinear dynamic systems without Lapunov's functions using is represented. The structure of complex and main features of complex realization on PC is presented. Structures of algorithms for few calculating modules of complex are represented.

Keywords: dynamic system, nonlinear, stability, bifurcation, distributed systems.

SOFTWARE SYSTEM FOR SUPPORT DECISIONS FOR OPERATIONAL PLANNING

***Kolesnikov A.V.**, Ph.D. (Immanuel Kant Baltic Federal University, Kaliningrad, avkolesnikov@yandex.ru);*
***Soldatov S.A.** (LLC «GIMAS+», Kaliningrad, soldatov@l-on.ru)*

Abstract. This paper describes a software system for support decisions for operational planning. Shows the structure of the system, its elements are described. The results of experiments on the example of machine-building enterprise – LLC plant «Kaliningradgazavtomatika» is presented.

Keywords: decision support system, operational planning, machine-building enterprise.

MULTIAGENT APPROACH IN THE SYSTEMS OF INFORMATION SUPPORT OF MANAGEMENT DECISIONS

***Kozminykh N.M.; Golovanov A.A.**, Ph.D.*
(Vyatka State University, Kirov, kzmnhn@yahoo.com, kzmnhn@gmail.com)

Abstract. The approach to creation of application of information support of management decisions is given in the article, the basic of which is represented by the modulus of multiagent system. The general model of multiagent system and the general of the model of it's agents are determined, the main agent types which take part in the decision are distinguished.

Keywords: agent, the agent model, multiagent approach, information support of management decision.

PLANNING PROBLEMS SOLVING IN COALITION MODEL

Zraenko A.S. *(Federal State Unitary Enterprise «Uralgeoinform», Ekaterinburg, zraenko@yandex.ru);*
Fedotov V.P.*, Ph.D. (Mechanical Engineering Research Institute, UB RAS, Ekaterinburg, fedotov@imach.uran.ru);*
Aksenov K.A.*, Ph.D. (Boris Yeltsin Ural State Technical University, Radio Engineering Institute, Ekaterinburg, wiper99@mail.ru)*

Abstract. In the present work are considered a method of scheduling theory and its application to solving scheduling problem in coalition model of the multi-agent resources transformation process. With the use of practical planning task was developed an algorithm of work implementation plans compilation.

Keywords: agent, multi-agent systems, scheduling theory, resources transformation process, the method of random search.

SELECTIONAL PREFERENCES FOR SYNTACTIC ANALYSIS

Malkovskiy M.G.*, Ph.D.;* ***Arefyev N.V.*** *(Lomonosov Moscow State University, malk@cs.msu.su)*

Abstract. In this article, we present a kind of computer dictionary which stores different types of selectional preferences. The use of this dictionary enabled us to improve a quality of syntactic analysis.

Keywords: syntactic analysis, computer dictionary, selectional preferences.

PARALLEL AUTOMATIC TEXT CATEGORIZATION SYSTEM

Kotelnikov E.V.*, Ph.D.;* ***Peskisheva T.A.*** *(Vyatka State University of Humanities, Kotelnikov.EV@gmail.com, peskisheva.t@mail.ru)*

Abstract. The paper describes structure and algorithm of parallel automatic text categorization system based on Support Vector Machines (SVM). Units, implementing main stages of system are discussed, particular attention is paid to classifier training process. The results of experiments based on text collection Reuters-21578 proving the effectiveness of the introduced parallel system are given.

Keywords: text categorization, support vector machines, SVM, parallel system, automatic text processing.

SOFTWARE IMPLEMENTATION IN MAPLE THE RESEARCHES OF STABILITY OF DEFORMABLE SYSTEMS BY UNIVERSAL METHOD

Chusova E.V. *(Tver State University, lioness@pop3.ru)*

Abstract. The main idea of the article is to present to the readers the program for researching the stability of the decision of nonlinear system. The developed program allows to make function of Hamilton, to calculate Jacobian for the basic and interfaced systems, to find points of positions of balance of the decision, to deduce the linear equation of the second order for the interfaced function, and also to calculate expressions of curvature of the decision.

Keywords: stability, universal method, program realization, flowchart.

METHODS OF THE ASSESSMENT OF MIS REAL PROTECTION LEVELS

Mukminov V.A.*, Ph.D.;* ***Khutsishvili V.M.****, Ph.D.;* ***Lobuzko A.V.*** *(The Forth Central Scientific Research Institute of the Ministry of Defence of Russia, mva131@mail.ru, xbm@mail.ru)*

Abstract. Methods of the assessment of MIS real protection levels in the condition of modeling computer attacks were considered. The list of assessment tools, including the simulation of computer attacks, was presented. The complex use of the new forms of tests as well as testing of MIS software programs on the basis of natural and simulated modeling was offered. The application of the new methods of the assessment of real protection levels will allow raising the efficiency of information safety services during MIS maintenance.

Keywords: management information systems, vulnerability protection, assessment of the protection levels, information protection tools, tools of computer attack detection.

INTEGRITY CONTROL OF INPUT DATA DURING AUTOMATED SOFTWARE ANALYSIS

Polyanichko M.A. *(St. Petersburg State Transport University, mark.polyanichko@gmail.com)*

Abstract. In this paper a possibility of applying an additional integrity check during execution of an automated software analysis is studied. An estimation of time losses is calculated, according to different data volumes.

Keywords: automated analysis, software, testing, fault, vulnerability, fuzzing, verification.

CLUSTERING OF ATOMIC KNOWLEDGE USING METHODS OF LINEAR OPTIMIZATION

Galeev A.Kh.

(Samara State University in Architecture and Civil Engineering, galeev@netcracker.com)

Abstract. The work deals with methods to form compact clusters of factual knowledge from unstructured information stored in the Internet. Some own mathematical models are offered, which allow to solve the problem using the methods of integer linear programming.

Keywords: clustering, knowledge, information, linear programming.

SEMANTIC SEGMENTATION OF LASER SCANNING DATA

Shapovalov R.V.; Velizhev A.B.; Barinova O.V.; Konushin A.S.

(Lomonosov Moscow State University, (shapovalov, avelizhev, obarinova, ktosh) @graphics.cs.msu.su)

Abstract. We address the segmentation problem for 3D point clouds retrieved by laser scanning of outdoor scenes. Associative Markov Networks (AMN) are used in the state-of-the-art methods for approaching the problem. An AMN does not allow expressing some natural interactions between objects such as «roof is likely to be above the ground». We use the general form of Markov random fields. It leads to significant performance improvement. We show how to perform inference and tune model's parameters. Oversegmentation is used to subsample a scan in order to improve efficiency and simplify cloud structure.

Keywords: computer vision, recognition, semantic segmentation, LIDAR, laser scanning, Markov random fields.

SEGMENTATION OF REGIONS OF INTEREST BASED ON ISOLINES CLASSIFICATION

Senyukova O.V.; Galanin V.E.

(Lomonosov Moscow State University, olsen222@yandex.ru)

Abstract. We propose a new approach to segmentation of regions of interest on images. The method can be especially useful in medical image processing. Application of the developed algorithm to the problem of segmentation of brain lesions on magnetic resonance image (MRI) is shown. The algorithm was tested on real data and demonstrated superiority over existing methods.

Keywords: computer vision, machine learning, image segmentation, semantic segmentation, medical image processing, isolines, contouring algorithm, segmentation of brain lesions.

DERIVING CONICAL BEARING TOOL FOR HANDLING MULTIAXIS

***Budnik A.I.; Kats E.I.**, Ph.D.*

(Boris Yeltsin Ural Federal University, Ekaterinburg, budnikalexandr@mail.ru)

Abstract. The paper considers the problem of constructing the curve of sliding on the surface of the conical tool at a time. An algorithm for constructing the curve of sliding on the surface of the conical tool. Modeling of such a curve is necessary, first of all, to model the surface (volume) swept out tool in the multiaxis milling.

Keywords: multiaxis processing, geometric modeling, curve slip, grazing points, conical tool.

CURVE BEARING ON THE SURFACE OF A TOOL FOR DIFFERENT KIND TREATMENT MULTIAXIS

***Budnik A.I.** (Boris Yeltsin Ural Federal University, Ekaterinburg, budnikalexandr@mail.ru)*

Abstract. The paper considers the problem of constructing the curve of sliding on the surface of various types ofinstrument at a certain time. The possible types of forms such a curve. In constructing the curve of the slipcan expect a more accurate representation of the machined surface in the simulation of the surface (volume) swept out by the tool in the multi-axis milling.

Keywords: multiaxis processing, geometric modelling, curve slip, grazing points, swept surface.

THE KNOWLEDGE BASE OF INFORMATION-CONTROL SYSTEM OF DRYING INSTALLATION

***Artemova S.V.**, Ph.D.; **Gribkov A.N.**, Ph.D.*

(Tambov State Technical University, GribkovAlexey@yandex.ru)

Abstract. Questions of working out of the frame knowledge base of information-control system of drying installation are considered.

Keywords: information-control system, frame base of knowledge.

CONTROL SYSTEM OF INNOVATIVE PRODUCTION SYSTEM

Matveykin V.G.**, Ph.D.; **Dmitrievskiy B.S.**, Ph.D.; **Panchenko I.S.
(Tambov State Technical University, irina-pnk@mail.ru)

Abstract. The task of control and modules of the control system of innovative production system are described. The structure of software package and it's program implementation are suggested.

Keywords: software package, innovative production system, status functioning graph of innovative production system.

ANALYSING THE STRESS-STRAIN STATE IN HETEROGENEOUS STRUCTURES

***Kandoba I.N.**, Ph.D. (Institute of Mathematics and Mechanics of Ural Branch of RAS, Ekaterinburg, kandoba@imm.uran.ru); **Spevak L.F.**, Ph.D. (Institute of Engineering Science of Ural Branch of RAS, Ekaterinburg, lfs@imach.uran.ru); **Tariko O.S.** (Ural State Medical Academy, Ekaterinburg)*

Abstract. Methods are presented to analyse the stress-strain state of a heterogeneous structure consisting of a finite number of elastic homogeneous regions and subjected to specified loading. The stresses and strains are calculated on the basis of the boundary element method. The realization of the boundary element method employs analytical integration formulae. The algorithms developed are realized in a specialized software package intended for solving a wide range of applied problems.

Keywords: elasticity, boundary element method, heterogeneous structure, contact boundary, optimal shape.

INTELLECTUAL SYSTEM OF CONTROL AND MONITORING OF THE GAS BOILER-HOUSE

Belousov O.A.**, Ph.D.; **Ivanov S.V.
(Tambov State Technical University, jiour@mail333.com, udarnikk@mail.ru)

Abstract. The problem of intellectual control and monitoring by thermal devices is considered, the algorithm of control and monitoring of a gas boiler-house in real time on the basis of a neural network and the fuzzy logic is developed.

Keywords: fuzzy logic, neural networks, automatic regulate, an information technology, control of a boiler, a gas boiler-house, thermal process, a control system.

SYSTEMS TECHNOLOGICAL PROCESSES CONTROL IN MULTIPRODUCT MANUFACTURES

Burdo G.B.**, Ph.D.; **Semenov N.A.**, Ph.D.; **Isaev A.A.
(Tver State Technical University, gbtms@yandex.ru)

Abstract. The paper describes the results of research function, structure and models automated control system with the elements of artificial intelligence for technological processes in single-part and smale-scale production.

Keywords: automated design system for technological processes, system analysis, artificial intelligence.

CONSTRUCTION OF CORPORATE NETWORK OF METALLURGICAL ENTERPRISE

***Lysetskyi Yu.M.**, Ph.D. («S&T Ukraine», Kiev, Ukraine, Iurii.Lisetskyi@snt.ua)*

Abstract. Presented the construction of a corporate backbone multiservice network of metallurgical enterprise. Formulated a set of requirements for modern corporate backbones, the corresponding infrastructure and functionality of the systems, given the sequence of problems solved in the course of their integration. Experience of realization of project.

Keywords: management of resources, multyservice network, territorial distributed systems, geterogeneous structure, universalization and standartization.

THE INNOVATIVE APPROACH TO PROGRAM-TOOL MEANS STRUCTURE DEFINITION FOR BUSINESS PROCESSES AT THE INDUSTRIAL ENTERPRISE

***Fedunets N.I.**, Ph.D.; **Goncharenko A.N.** (Moscow State Mining University, gan@ngs.ru)*

Abstract. In article is presented the innovative approach to creation of uniform program decisions structure for increase efficiency functioning at the industrial enterprise. Working out of definition optimum program-tool means structure model has allowed to receive a gain of technical-economic, organizational and financial indicators of functioning at the industrial enterprise.

Keywords: information technology, efficiency of business processes, the industrial enterprise, the innovative approach.

MODELING OF WAREHOUSING: DEVELOPMENT AND INTEGRATION IN ORLANDO TOOLS

Basharina O.Yu. *(Irkutsk State University, basharinaolga@mail.ru);*
Gorsky S.A.*, Ph.D. (Institute of Systems Dynamics and Control Theory of Siberian Branch of RAS, Irkutsk, gorsky@icc.ru)*

Abstract. This article is dedicated to the questions of modeling of the modern logistic warehouses on base of multivariant experiments in the parallel computing system. The package of modeling is developed in toolkit Orlando Tools. The object-oriented control system of the knowledge base of Orlando Tools provides complexation by data for all packages of applied programs developed on base of this toolkit.

Keywords: warehouse logistics, modeling, parallel computing.

THE CONSTRUCTION PRINCIPLE OF THE COMPUTER-AIDED DESIGN SYSTEM OF HYDRAULIC MACHINES AND APPARATUS

Krutikov V.S.*, Ph.D.;* ***Likhoded K.A.****;* ***Kopitsa V.V.***
(South Russian State Technical University (Novocherkassk Polytechnic Institute), klikh@mail.ru, vadimnpi@mail.ru)

Abstract. Principles of construction of computer-aided design system, which provide complex solution of all problems creation of new machines and apparatus.

Keywords: hydraulic machines and apparatus, computer-aided design system, complex solution of all problems.

UNIVERSAL SOFTWARE SYSTEM FOR MONITORING OF TECHNOLOGICAL PROCESSES

Kalabin A.L.*, Ph.D.;* ***Kozlov A.V.*** *(Tver State Technical University, alex.ka.86@gmail.com);*
Pakshver E.A.*, Ph.D. (LLC «Tenaks», Moskow)*

Abstract. In the article authors describe the architecture of the software system for monitoring of various technological processes. Software system includes three modules: module for creating schemas of different processes, module for storing and editing data, module that contain different tools for processing and analyzing data. The software was tested using data from the process of polyacrylonitrile production. In the article researchers give some examples of the analysis of the dependences between parameters and offer some advises according to the analysis.

Keywords: control, monitoring, manufacturing process, software architecture.

MODEL OF FORMATION OF ADAPTIVE LEARNING ENVIRONMENT AND EVALUATION OF ITS EFFECTIVENESS

Zaydullina S.G.*;* ***Migranov N.G.****, Ph.D.*
(M. Akmulla Bashkir State Pedagogical University, Ufa, sv_sa@mail.ru, ufangm@yahoo.co.uk)

Abstract. This paper considers a model of adaptation of presentation of material in the system of an electronic support in the learning process. We propose a system for creating interactive courses based on Internet technologies, taking into account the psycho-physiological characteristics of trainees.

Keywords: development tools software, model, electronic adaptive learning systems.

TECHNOLOGY OF INDICATORS MODELLING OF SCIENTIFIC ACTIVITY AT CREATION INFORMATION ANALYTICAL SYSTEM OF HIGH SCHOOL

Maletskiy R.V.*;* ***Pikulin V.V.****, Ph.D. (Penza State Technological Academy, pvv@pgta.ru)*

Abstract. Defining a questions of modeling of indexes of scientific activity of high school, the mathematical and structural models being used for the solution of a given task and also the architecture of program system and its implementation are under discussion in the article.

Keywords: scientific activity, human resources, automation, information-analytical system.

PROCESSING TECHNIQUES IN COMPUTER PERFOMANCE EVALUATION OF THE EDUCATIONAL PROCESS

Dolgov A.I.*, Ph.D.;* ***Martynenko A.F.****;* ***Presnukhin V.V.****, Ph.D.*
(Rostov Military Institute of Rocket Troops, dolgov-ai@yandex.ru)

Abstract. In article processing methods in computer coefficient techniques of initial (entrance) and intermediate indicators for reception of target indicators of a rating estimation of educational process (estimated

object) with the account of standard logic conditions are considered.

Keywords: computer coefficient a technique, educational process, changeable weight factor, the adder, the multiplier.

DISTRIBUTED INFORMATION SYSTEM FOR LITERATURE RESEARCH

***Maran M.M.**, Ph.D.; **Lvin Maung So** (Moscow Power Engineering Institute (Technical University), mm@appmat.ru, lwinmgsoe@gmail.com)*

Abstract. This paper deals with design principles and implementation of distributed information systems for information resources search. The implementation is made in accordance with the principles of service-oriented architecture on Windows Communication Foundation, Microsoft SQL Server 2008 and Visual Studio 2010 platforms.

Keywords: information system, information resources search, SOA, WCF, ADONET MVC.

DEVELOPING OF MANDATORY SECURITY POLICY MECHANISM IN CIM BASED ON ORACLE ECM 11

***Proskuryakov M.A.**; **Palyukh B.V.**, Ph.D.; **Melnikova V.V.** (Tver State Technical University); **Kotov S.L.**, Ph.D. (Chief Test Certification Center Software Computing, Tver, gic@tvcom.ru, info@ooogic.ru)*

Abstract. The problem of Oracle ECM 11g mandatory access control subsystem correspondence to the requirements of management directive «Computer Aids» is considered in the article. The main idea of the article is mandatory access policy mechanism implementation in Oracle ECM 11g.

Keywords: Oracle ECM 11g, mandatory Bell-LaPadula model, attestation, content server, computer integrated manufacturing, certification tests.

SOFTWARE SIMULATION MODEL WALTER GREY'S MACHINA SPECULATRIX

Kolos P.A.**; **Volkova N.S.

(Bogdan Khmelnitsky Cherkassy National University, dr_peter@i.ua, manunya@i.ua)

Abstract. The article deals with the features of modeling of the living beings behavior for the creation of technical systems. There are described the cybernetic biomorphic robotic systems machina speculatrix made by Walter Gray, also there are investigated the features of the creation and using of simulation models of such systems in this article.

Keywords: biomimetic, cyber turtles, robotic systems, biomorphic robots, software simulation models.

VERILOG-A COMPACT MODEL OF GRAPHENE FIELD-EFFECT TRANSISTOR

Tselykovskiy A.A.**; **Danilov I.A.

*(Scientific Research Institute of System Analysis, RAS, atsel@niisi.msk.ru, danilov@niisi.msk.ru); **Zebrev G.I.**, Ph.D. (National Research Nuclear University «MEPhI», gizebrev@mephi.ru)*

Abstract. The paper presents a compact model of the graphene field-effect transistor and its Verilog-A implementation. The potential for using the model in the industry CAD is demonstrated by modeling of analog circuits based on graphene transistors.

Keywords: graphene field-effect transistor, GFET, compact model, modeling, current-voltage characteristic, Verilog-A, CAD, full-wave rectification, frequency multiplication, phase shift keying, ambipolar electronics.

THE REVIEW OF SYSTEMS OF GATHERING AND PREPROCESSING THE INFORMATION APPLIED IN STATIONS OF GEOLOGY-TECHNOLOGICAL RESEARCHES

Kalyonov S.A.

(Tyumen State Oil and Gas University, Institute of Cybernetics, Computer Science and Communication, polinochcka86@rambler.ru)

Abstract. In article systems of gathering the information applied at stations of geology-technological researches SIRIUS and «Geosphere» for support of drilling of chinks are considered extended, for today. Technical characteristics, reliability and convenience of operation of systems are compared.

Keywords: SIRIUS, «Razrez-2», «Geosphere», geology-technological researches, the device of gathering of the information.

DEVELOPMENT TOOLS OF VERIFICATION FOR DEVICE DRIVERS BASED ON SEMANTIC MODELS

Korablin Yu.P.**, Ph.D.; **Pavlov E.G.
(Russian State Social University, Moscow, y.p.k@mail.ru, lucenticus@gmail.com)

Abstract. In this article the questions of driver's proving are investigated. It is proposed the method of drivers verification, based on the processing semantics approach.

Keywords: verification, device drivers, process semantics, equational characterization, algebraic semantics.

VIRTUAL MACHINE PLACEMENT METHOD WITH RESOURCE REDISTRIBUTION

Solovyev V.P.**, Ph.D.; **Udovichenko A.O.
(Moscow State University of Railway Engineering, wsolovjov@gmail.com, aspudovichenko@mail.ru)

Abstract. A method for placement of virtual machines with resource redistribution is proposed in the paper. A problem of virtual machine placement is presented as well as the proposed method. Results of experiments are provided proving the efficiency of the proposed method.

Keywords: information system, infrastructure, virtualization, virtual machine, resource management.

AUTOMATIC CONSTRUCTION AND GENERALIZATION OF CIRCUIT DECISIONS AT DESIGNING OF CONTROL SYSTEMS

***Filatova N.N.**, Ph.D.; **Trebukhin A.G.** (Tver State Technical University, nfilatova99@mail.ru)*

Abstract. New program complex for designing of control systems. The program automatically generates set of function charts, makes the analysis of schemes from system archive and creates new rules of schemes construction.

Keywords: automation system scheme, decision-tree scheme model, rough set, decision rule.

MULTIMETHOD SIMULATION FOR OPTIMIZATION OF THE ORGANIZATION STRUCTURE IN CONSULTING COMPANY

***Mitroshin S.G.; Pikulin V.V.**, Ph.D. (Penza State Technological Academy, pvv@pgta.ru)*

Abstract. Questions of application of the multimethod simulation modeling for increase of efficiency of activity of the consulting company by change of composition of experts are considered. There are an example of a simulation model developed in the AnyLogic modeling tool, and the results of its use.

Keywords: consulting, multimethod simulation modeling, staff planning.

PARALLEL SOFTWARE PACKAGE FOR NONHOLONOMIC CONTROL PROBLEMS

***Mashtakov A.P.** (Aylamazyan Program Systems Institute of RAS, Pereslavl-Zalessky, alexey.mashtakov@gmail.com)*

Abstract. A motion planning problem for nonlinear five-dimensional systems is considered. Parallel software package MotionPlanning235 was developed to solve this problem in class of piecewise constant and optimal controls. Nilpotent approximation is used to obtain an approximate solution with a necessary precision.

Keywords: nilpotent approximation, optimal control, parallel algorithms and programs.

CELLULAR AUTOMATA MODELING OF DISSOLUTIOM PROCESES

Menshutina N.V.**, Ph.D.; **Ivanov S.I.**; **Shipilova D.D.
(Mendeleyev University of Chemical Technology of Russia, patephon2009@yandex.ru)

Abstract. This paper is dedicated to mathematical modeling of solid-stage bodies dissolution on the ground of probabilistic cellular automata. In this paper the description of mathematical model of dissolution is represented, the operation algorithm of cellular automata running this model was examined. There were given a comparison of experimental (kinetics of ascorbic asid dissolution) and calculation data.

Keywords: cellular automata, dissolution, modeling, production sector, multicomponent mixture, diffusion.

A SUBSTANTIATION OF APPLICATION OF ELECTRONIC SIMULATORS FOR OPERATORS' EDUCATION

***Arepin Yu.I.**, Ph.D. (Research Institute «Centrprogrammsystem», Tver, arep@cps.tver.ru)*

Abstract. An approach to substantiation of application of various training tools for complex technical systems operators is observed.

Keywords: complex technical system, operator, automated workstation, training system.

С О Д Е Р Ж А Н И Е