

ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ

Государственное образовательное учреждение высшего  
профессионального образования

Ульяновский государственный технический университет

ПРИКЛАДНЫЕ  
ИНТЕЛЛЕКТУАЛЬНЫЕ  
СИСТЕМЫ, ОСНОВАННЫЕ  
НА МЯГКИХ ВЫЧИСЛЕНИЯХ

Под редакцией Н. Г. Ярушкиной

Ульяновск 2004

УДК 004.8  
ББК 32.813  
П75

Рецензенты: д-р физ.-мат. наук, профессор А. А. Бутов;  
кафедра математической кибернетики и информатики  
Ульяновского государственного университета

Авторы:

М. С. Азов, Ю. Ю. Бушмелев, А. А. Лебедев, А. С. Макеев, И. В. Се-  
мушин, М. С. Сунопля, М. А. Федорова, А. Б. Шамшев, Т. Р. Юнусов,  
Н. Г. Ярушкина, А. А. Стицко.

П75     **Прикладные** интеллектуальные системы, основанные на мягких вы-  
числениях/под ред. Н. Г. Ярушкиной.—Ульяновск: УлГТУ, 2004 – 139 с.

ISBN 5-89146-700-0

Рассматривается прикладной потенциал научного направления «мягкие вычисления». Описаны интеллектуальные системы моделирования. Показаны возможности мягких вычислений в современных информационных технологиях.

Книга будет полезна для специалистов по интеллектуальным информационным технологиям, а также в преподавании дисциплины «Интеллектуальные информационные системы» для специальности 352400 «Прикладная информатика (в экономике)».

УДК 004.8  
ББК 32.813

ISBN 5-89146-700-0

© Оформление. УлГТУ, 2004  
© Колл. авторов, 2004

## ОГЛАВЛЕНИЕ

ПРЕДИСЛОВИЕ .....	7
ВВЕДЕНИЕ.....	9
Глава 1. ФУНДАМЕНТАЛЬНЫЕ ЗАДАЧИ ВЫЧИСЛИТЕЛЬНОГО ИНТЕЛЛЕКТА .....	13
1.1. Фундаментальные и технологические причины конвергенции нейроинформатики и искусственного интеллекта .....	13
1.1.1. Нейроинформатика .....	14
1.1.2. Искусственный интеллект.....	15
1.2. Перспективы и задачи вычислительного интеллекта .....	16
1.2.1. Моделирование восприятия и умозаключений. Интеграция нейронных сетей и систем логического вывода – нечеткие нейронные сети.....	17
1.2.2. Моделирование сознания и неосознанного .....	17
1.2.3. Интеграция нейронных сетей и вероятностных вычислений .....	18
1.3. Синергетический компьютер.....	18
1.4. Аттракторные нейронные сети.....	19
1.5. Искусственный, естественный и нейроинтеллект.....	20
1.5.1. Баланс правополушарного и левополушарного .....	20
1.6. Искусственная эволюция интеллектуальных систем.....	21
Заключение .....	22
Библиографический список .....	22
Глава 2. МЯГКИЕ ВЫЧИСЛЕНИЯ В МОДЕЛИРОВАНИИ СИСТЕМ .....	23
2.1. Использование генетических алгоритмов для параметрической идентификации моделей.....	23
2.1.1. Введение.....	23
2.1.2. Постановка задачи .....	24

2.1.3. Общая схема процесса идентификации.....	27
2.1.4. Моделируемые генетические алгоритмы [7, 8, 12, 13]....	30
2.1.5. Некоторые результаты вычислительных экспериментов	33
2.1.6. Заключение .....	38
2.2. Метод взвешенных квадратов невязок для построения функций пригодности .....	39
2.2.1. Введение.....	39
2.2.2. Постановка задачи .....	40
2.2.3. Метод обнаружения на основе ВКН.....	42
2.2.4. Проверка и выбор фильтра обратной связи на основе ВКН .....	47
2.2.5. Некоторые результаты вычислительных экспериментов..	50
2.2.6. Разработка моделирующего программного обеспечения .	52
2.2.7. Заключение .....	55
Приложение. Доказательство теоремы 2.1.....	60
Библиографический список .....	62
Глава 3. МЯГКИЕ ВЫЧИСЛЕНИЯ В ПРОЕКТИРОВАНИИ ВЫЧИСЛИТЕЛЬНЫХ СЕТЕЙ.....	64
3.1. Этапы проектирования вычислительных сетей. Место технологий мягких вычислений на разных этапах проектирования .....	64
3.2. Система проектирования ВС на основе потоковых диаграмм данных .....	65
3.2.1. Общая характеристика проблемы проектирования .....	65
3.2.2. Нечеткое описание трафика вычислительной сети.....	66
3.2.3. Реализация САПР вычислительных сетей на основе потоковых диаграмм .....	69
3.3. Проектирование вычислительных сетей на основе байесовских сетей доверия .....	72
3.3.1. Архитектура САПР ВС.....	72
3.3.2. Динамическая модель элементов САПР ВС .....	74
3.3.3. Генетическая оптимизация вычислительной сети .....	75
3.3.4. Определение и механизм работы байесовских сетей доверия .....	76
3.3.5. Представление информации об интенсивности взаимодействия узлов ВС сетью доверия.....	79
Библиографический список .....	88

Глава 4. МЯГКИЕ ВЫЧИСЛЕНИЯ В ТЕХНОЛОГИИ БАЗ ДАННЫХ .....	90
4.1. Определение, возможности и ограничения нечеткого интеллектуального сервера данных .....	90
4.1.1. Возможности и перспективы нечеткого реляционного интеллектуального сервера данных .....	90
4.1.2. Ограничения существующих нечетких реляционных моделей.....	92
4.1.3. Определение нечеткого реляционного сервера данных ....	93
4.2. Нечеткий интеллектуальный реляционный сервер данных ....	96
4.3. Система анализа нечетких тенденций .....	102
4.3.1. Проблема Data Mining .....	102
4.3.2. Задача обработки нечеткой тенденции.....	103
4.3.3. Метод анализа НТ .....	105
4.3.4. Реализация метода .....	105
4.3.5. Оценка .....	107
Глава 5. МЯГКИЕ ВЫЧИСЛЕНИЯ В СОВРЕМЕННЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЯХ .....	110
5.1. Основные задачи и мягкие технологии в реализации интеллектуального хранилища информационных ресурсов.....	110
5.2. Реализация интеллектуального хранилища .....	111
5.2.1. Публикация ресурсов.....	111
5.2.2. Поиск ресурсов.....	113
5.2.3. Управление хранилищем.....	113
5.3. Функции автоматизированной системы проектирования нечетких контроллеров.....	115
5.4. Реализация системы проектирования НК.....	116
5.4.1. Анализ методов реализации нечетких контроллеров .....	116
5.4.2. Некоторые существующие аппаратные fuzzy процессоры .....	118
5.4.3. Пример проектирования нечеткого контроллера в среде Matlab.....	119
Библиографический список .....	122
Глава 6 МЯГКИЕ ВЫЧИСЛЕНИЯ В ЭКОНОМИЧЕСКИХ И ГУМАНИТАРНЫХ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМАХ.....	124
6.1. Задача анализа терминосистемы нечеткой логики.....	124
6.2. Реализация системы анализа терминосистемы нечеткой логики .....	126

6.2.1. Модель образования терминосистемы нечеткой логики.	126
6.2.2. Синонимичность .....	127
6.2.3. Реализация программы обработки терминосистемы .....	128
6.3. Задача моделирования доходов населения региона.....	136
Библиографический список .....	137
ЗАКЛЮЧЕНИЕ .....	137

## ПРЕДИСЛОВИЕ

В начале 90-х годов К. Циммермен (Аахен, Германия) предложил термин «вычислительный интеллект». Синонимичный «зонтичный» термин предложил в 1994 г. На семинаре в Беркли Л. Заде – «мягкие вычисления», включающий в себя нечеткую логику, нейронные вычисления эволюционные вычисления. Рождение новых гибридных терминов означает начало объективного процесса интеграции различных направлений в интеллектуальном моделировании. В течение первого пятилетия научного направления усилия исследователей были направлены на теоретические результаты, устанавливающие возможности и ограничения новых методов. В последние пять лет успешно развиваются прикладные мягкие интеллектуальные системы. В настоящей монографии рассматривается прикладной потенциал научного направления мягкие вычисления. Описаны интеллектуальные системы моделирования трафика телекоммуникационных и вычислительных сетей на базе нечеткой случайной величины. В составе задач моделирования сетей:

- задача оптимизации распределения потоковых диаграмм бизнес – процессов по структуре вычислительной сети;
- задача проектирования интеллектуальной автоматизированной системы проектирования сетей на основе байесовских сетей доверия;
- задача моделирования нечеткого роутинга в телекоммуникационных сетях, представленных нечетким гиперграфом.

Приведены результаты исследования эффективности генетических алгоритмов в задачах идентификации моделей. Показаны возможности мягких вычислений в современных информационных технологиях:

- интеллектуальном анализе данных (Data Mining)
- построении нечетких реляционных серверов данных Fuzzy Data Base
- разработке нейросетевой системы управления информационным хранилищем (Text Warehouse).

Актуальность нового прикладного вычислительного интеллекта связана с рядом задач развития информационных технологий.

Материал монографии «Прикладные интеллектуальные системы, основанные на мягких вычислениях» сложился в результате более чем десятилетних исследований в области мягких вычислений. Представлен современный уро-

вень аппаратуры нечетких процессоров. Кроме технических областей применения, рассмотрено решение задачи моделирования социально – экономического положения населения региона и формирования терминосистем подязыков науки на примере нечеткой логики. Такой широкий охват возможных применений мягких вычислений стал возможен благодаря работе большого авторского коллектива. Глава 1, п. 3.1, 4.1, 4.2, 5.3, 6.3 написаны Ярушкиной Н. Г., глава 2 – А. Е. Кондратьевым, И. В. Семушиным, М. С. Суноплей, О. А. Фатьяновой, М. А. Федоровой, А. Д. Юрьевым при решающем вкладе И. В. Семушина. Подразделы 3.2, 3.3, 3.4 третьей главы написаны соответственно Юнусовым Т. Р., в главе 5 п. 5.1, 5.2 написаны Бушмелевым Ю. Ю., а п. 5.4 – Лебедевым А. А. Подпункты 6.1 и 6.2 написаны Арзамасцевой И. В. В разработке описанных систем в разное время принимали участие канд. техн. наук Наместников А. М., канд. техн. наук Пирогов В. В., канд. техн. наук Краснов С. В., канд. техн. наук Горбоконек Е. А. и канд. техн. наук Мытарев П. В., которые выполнили кандидатские исследования по тематике монографии и успешно защитили диссертации. Учитывая разнообразные области применения рассматриваемых прикладных интеллектуальных систем, список литературы разделен на литературу, относящуюся к конкретной главе или пункту главы и на общий список литературы, имеющий отношение ко всем главам. Ссылки в книге относятся к тем спискам литературы, которые приводятся в конце глав.

Ульяновск  
ноябрь 2004

Н. Г. Ярушкина



## **ВВЕДЕНИЕ**

Рассмотрим также некоторые новые применения гибридных интеллектуальных систем, характеризующие их прикладной потенциал, в том числе для традиционных задач искусственного интеллекта. Выберем направления, сочетающие распознавание и логическую обработку. На наш взгляд место для успешных применений гибридных технологий может быть найдено в направлении интеллектуализации информационных систем: баз данных и корпоративных хранилищ ресурсов.

### **Интеллектуализация баз данных**

Технология баз данных является важнейшей из прикладных технологий информационных систем, так как на ней строятся экономические информационные системы любого класса. Потребность в интеллектуализации баз данных связана с несколькими причинами.

Базы данных состоят из десятков тысяч рабочих таблиц, индексов, ограничений, доменов, представлений. Простого словаря, регистрирующего объекты, недостаточно для управления данными. Обработка транзакций в корпоративных системах (более 1000 автоматизированных рабочих мест) требует встраивания в сервер данных логики событий.

Состоявшаяся в свое время в искусственном интеллекте широкая дискуссия о том, чем данные отличаются от знаний, остановилась на мнении, что «знания, это данные и их интерпретация», то есть знания – это еще и метазнания. Ограничения целостности, характерные для серверов данных, это всегда знания о законах проблемной области. Простейшими примерами являются сведения о том, что среди сотрудников предприятия нет лиц моложе 16 лет, что годовой доход продавца включает кроме оклада комиссионные и т.д. В современных системах полное описание ограничений целостности совпадает с внутрикорпоративными стандартами организации бизнеса. Показательно, что крупные производители систем баз данных (Oracle Inc.) включают в состав программного продукта систему описания и интерпретации бизнес-правил в форме продукций ЕСЛИ – ТО и систему логического вывода действий. Особенностью таких систем знаний, представляющих собой метауровень над данными, явля-

ется реализованная семантика и прагматика. Любое действие, которое выводится из бизнес-правил, реализуется в системе как транзакция обработки данных.

## **Обработка транзакций**

Особенно интересным является использование интеллектуальных методов при обработке транзакций, так как модуль транзакций – это центральная компонента любого сервера данных. При многопользовательском доступе сервер данных отслеживает огромное количество событий: запуск и окончание транзакций, блокировки ресурсов, откаты транзакций, ожидания и нарушение прав доступа. Сервер данных реализует прикладную логику событий, которая описывается явно в системе приоритетов, режимов и прав доступа, то есть представляет собой слой знаний.

## **Новые возможностные типы данных**

Взаимодействие баз данных с системами обработки знаний расширяет базовые типы данных реляционной модели данных. Если система знаний представлена системой нечетких продукций, то такая система становится необходимой системой хранения и обработки нечетких атрибутов. В настоящее время построена нечеткая реляционная алгебра, выполняющая операции соединения, селекции, проекции, объединения и пересечения. Нечеткая реляционная алгебра составила теоретическую основу для реализации системы представления, хранения и обработки нечетких данных Fuzzy Data Manager, которая основывается на существующей промышленной базе данных Oracle 8i. Нечеткий реляционный сервер данных – это репозиторий, который предназначен для хранения функций принадлежности, лингвистических меток.

## **Интеллектуальная обработка кубов данных**

В настоящее время прогрессивными организациями осознается, что базы данных, накопленные ими за длительный период работы, содержат основные законы их бизнеса, которые необходимо осознать. К сожалению, традиционная технология баз данных предписывает хранить в актуальной форме только оперативные данные, а данные прошлых периодов хранятся в компактном, но недоступном для непосредственного анализа архиве. В результате развиваются модели темпоральных баз данных или кубов данных. Кубом данных называют реляционную таблицу, имеющую третью координату – время. Появляются новые подмножества SQL-языка, содержащие предикаты времени. На наш взгляд подмножество темпорального языка должно включать логику времени, разработанную в искусственном интеллекте (ИИ). Это первый «условный» уровень интеллектуализации кубов данных. Второй уровень составят сами системы анализа данных и извлечения закономерностей.

## **Data Mining**

Анализ данных (Data Mining) – это научное направление, изучающее методы извлечения законов проблемной области из баз данных. Такие «законы», разумеется, не имеют строгой аналитической формы, а в лучшем случае аппроксимируются некоторым набором правил ЕСЛИ-ТО. Следовательно, Data Miner имеет на входе базу данных (обучающую выборку), а на выходе базу правил (базу знаний). Но именно такие входы и выходы имеет нечеткая нейронная сеть, которая в таком контексте становится средством класса Data Miner.

## **Интеллектуализация Intranet и Internet-технологий**

Корпоративные хранилища информационных ресурсов строятся на основе технологии менеджеров файлов и внутреннего WWW-сервиса. Под ресурсом понимают совокупность файлов, объединенных общей семантикой. Хранение и использование таких ресурсов требует построения индексов хранения, которые не просто обеспечивают быстрый поиск ресурса, а становятся метаописанием ресурсов, т.е. знаниями.

## **Интеллектуальная поисковая машина ресурсов**

Чтобы обеспечить хранение и поиск ресурсов необходимо формировать индексы (каталоги) в соответствии с систематическими рубриками. Рубрики представляют собой классы объектов проблемной области, причем трудно вручную составить исходную достаточно полную классификацию ресурсов. Следовательно, необходимо решить задачу кластеризации – порождение системы нечетких классов для ресурсов. Затем, каждый новый ресурс необходимо относить к нечетким классам. Очевидно, что такую задачу можно решить только на основе аннотаций текстов, которые должны присутствовать в каждом ресурсе. Текст можно отнести к нечеткому классу на основе встречаемости ключевых слов. Для решения задачи кластеризации можно применить нейронную сеть Кохонена, обучаемую по алгоритму победителя, задача классификации ресурсов решается за счет нечеткой нейронной сети, например, класса ANFIS.

## **Моделирование вычислительных и телекоммуникационных сетей на основе бизнес – диаграмм**

В условиях проектирования вычислительной сети, исходные технические данные включают прогноз трафика. Прогноз можно сделать на основе измерений в старой сети, если речь идет о модернизации или на основе мнения эксперта, выраженного в лингвистической форме. Современные методики информатизации организаций предписывают реинжиниринг бизнес-процессов, то

есть описание бизнес-процессов предшествует проектированию вычислительной сети. Из описаний бизнес-процессов можно извлечь данные и рассчитать прогнозный трафик. Основу расчета составляют диаграммы потоков данных. Но расчет представляет собой вывод на основе вероятностных рассуждений, осуществленных, например, с помощью байесовских сетей (сетей доверия). Таким образом, и байесовская сеть может рассматриваться как средство извлечения и обработки знаний.

## **Глава 1. ФУНДАМЕНТАЛЬНЫЕ ЗАДАЧИ ВЫЧИСЛИТЕЛЬНОГО ИНТЕЛЛЕКТА**

### **1.1. Фундаментальные и технологические причины конвергенции нейроинформатики и искусственного интеллекта**

В современной нейроинформатике соотношение фундаментальных и технологических задач складывается в пользу технологических. Отметим, что фундаментальные задачи: моделирование ощущений, восприятий, распознавание образов, обучение и запоминание паттернов являются основными задачами когнитивной науки. Технологические задачи связаны с повышением эффективности использования нейронных сетей в решении задач кластеризации, классификации образов и сцен, аппроксимации функций для процессов технической, экономической, биологической природы. Достижения нейроинформатики в решении технологических задач привели к созданию целого спектра разнообразных нейронных сетей: сетей с прямым распространением сигнала, рекуррентных сетей, радиально базисных сетей. Созданы развитые архитектуры гибридных систем: нечетких нейронных сетей, нечетких нейронных сетей с генетической настройкой параметров.

**Ситуация в нейроинформатике требует вернуться к фундаментальным когнитивным задачам и выяснить, позволяют ли новые классы нейронных и гибридных систем решать новые когнитивные задачи.**

Объектом анализа будут служить нечеткие нейронные сети с генетической оптимизацией параметров, так как они обладают комплексной структурой и сложными алгоритмами обучения, интегрирующими структурные части различных нейронных сетей. Следовательно, именно гибридные системы, как можно предположить, должны обладать сильным когнитивным потенциалом. Далее будем называть именно их гибридными системами. Гибридные системы включают в себя слои радиально базисных нейронов, логических нейронов, традиционных пороговых суммирующих нейронов. Алгоритм их обучения обычно комбинирует соревновательное обучение (по алгоритму победителя), генетическую оптимизацию параметров и классический метод обратного распространения ошибки [1].

### 1.1.1. Нейроинформатика

#### 1.1.1.1. История

Исследования восприятия на искусственных моделях нейронных сетей (НС) успешно развивались с 40-х годов прошлого века (У. Мак Каллок и У. Питтс 1943, Д. Хеббс 1949). Алгоритм обучения линейного однослойного перцептрона, предложенный Ф. Розенблатом, показал, что нейроподобные вычисления (*brain-style computation*) эффективны для задач восприятия и распознавания. В 1982 г. Дж. Хопфилд изложил математические основы динамики нейронных сетей с обратными связями. В 1984 г. Т. Кохонен предложил алгоритм обучения «без учителя» для самоорганизующихся сетей. В 1986 г. Д. Румельхарт вывел алгоритм обратного распространения ошибки для нелинейных многослойных сетей. Успехи в области нейронных сетей, нечетких систем и генетических алгоритмов привели к формированию нового направления, так называемого вычислительного интеллекта.

#### 1.1.1.2. Состояние

Эффективность аппарата НС определяется аппроксимирующими возможностями. В 1989 г. японским ученым Л. Фунахаши была доказана теорема об аппроксимации. Теорема «о функциональном универсальном аппроксиматоре» формулируется следующим образом:

Пусть  $Y(t)$  - множество непрерывных монотонных возрастающих функций;  $K$  - компактное множество  $R^N$ , а функция  $Y(t): K \rightarrow R$  дает на выходе конкретное число; тогда для любой функции  $F$  существуют массивы чисел  $W_i$ ,  $W_{ij}$ , которые аппроксимируют функцию  $F$ :

$$\tilde{F} = \sum_i^M W_i Y_i \left( \sum_{j=1}^N W_{ij} X_j \right), \text{ так что } \left\| F - \tilde{F} \right\| = \sup \left\| F - \tilde{F} \right\| < E, \forall E > 0, X \in K.$$

#### 1.1.1.3. Перспективы и задачи

Перспективы фундаментальных исследований нейронных сетей связаны с развитием двух существенных идей, высказанных в последнее десятилетие развития нейроинформатики:

- развитие квантового компьютера, которое можно считать «линией Хопфилда в нейроинформатике», превратившего «нейронные сети в раздел теоретической физики»;
- развитие осциляторных сетей, основанных на последних исследованиях мозга;
- развитие гибридных систем, в частности нечетких нейронных сетей, вероятностных НС.

Развитие квантового и осцилляторного компьютера важно с точки зрения развития идеи аттрактивной нейронной сети или НС детерминированного хаоса, так как предлагают конкретные параметры порядка: энергия и частота осцилляций. Минимум энергии или синхронизации частоты НС – это аттрактор НС, состояние которой таким образом распознает паттерн.

Разработка гибридов НС и хаотических вычислений нуждается в конструктивном уточнении.

## 1.1.2. Искусственный интеллект

### 1.1.2.1. История

В истории искусственного интеллекта (ИИ) можно выделить термины - предшественники: *кибернетика, информатика, вычислительная техника*; термины - попутчики: *бионика, психоника, распознавание образов, нейроинформатика* и термины - потомки: *инженерия знаний, вычислительный интеллект, мягкие вычисления*.

В объем современной информатики входят: *Теория алгоритмов; Логические модели; Базы данных; Искусственный интеллект* (представление знаний, вывод на знаниях, обучение, экспертные системы); *Бионика; Распознавание образов и обработка зрительных сцен; Теория роботов; Инженерия математического обеспечения; Теория компьютеров и вычислительных сетей; Компьютерная лингвистика; Численные и символьные вычисления; Системы человеко-машинного взаимодействия; Нейроматематика и нейросистемы; Использование компьютеров в управлении* [4].

**Почему научные направления, моделирующие интеллектуальную деятельность человека с разных точек зрения, такие как нейроматематика, теория роботов, распознавание образов, бионика, искусственный интеллект, входят в информатику как отдельные направления, отличающиеся языком, методами и кадровым потенциалом?**

Разделение исследователей произошло из-за разных ответов на вопросы: ***что такое интеллект и как можно построить разумную машину?***

Часть исследователей отвечала на вопрос таким образом: *интеллект* – это способность человека к формально-логической обработке символьной информации. А разумная машина должна эффективно перерабатывать символы (Минский, Маккарти, Робинсон, Кольмероз). Таким образом, важен знак, символ, слово, понятие. Со временем именно данное направление стали называть *искусственный интеллект*

Часть исследователей отвечали на вопрос так: «*интеллект* – это способность человеческого мозга к целенаправленной деятельности, сложная система рефлексов, инстинктов, логической обработки информации. Думающая машина может быть только подобна естественному интеллекту» (Розенблатт, Маккаллок, Хеббс). Таким образом, важен сигнал, ощущение, восприятие. Сторонники

этого направления работают в направлениях: *нейроинформатика, распознавание образов, бионика*.

К. Циммермен (Аахен, Германия) предложил в начале 90-х годов термин «*вычислительный интеллект*». Синонимичный «зонтичный» термин предложил в 1994 г. на семинаре в Беркли Л. Заде – «*мягкие вычисления*», включающий нечеткую логику, нейронные вычисления и эволюционные вычисления. Рождение новых гибридных терминов означает начало объективного процесса интеграции различных направлений в интеллектуальном моделировании.

#### *1.1.2.2. Состояние*

К настоящему времени предмет искусственного интеллекта остался по прежнему неопределенным, что не мешает конкретным исследованиям быть полезными. Во всяком случае предмет искусственного интеллекта настолько же определен, как и у философии, химии, физики, между которыми в современном состоянии невозможно провести границу.

За 40 лет развития искусственного интеллекта основные результаты были получены не в решении фундаментальных проблем, а в решении отдельных научно-технических задач, хотя влияние ИИ на развитие науки и техники очень велико. Перечислим только основные достижения ИИ.

- создание систем, основанных на знаниях (*knowledge based system*), решение задач представления, обработки и извлечения знаний, разработка многочисленных экспертных систем;
- разработка естественно-языковых интерфейсов и развитие машинного перевода. Хотя ни одна из этих задач и не решена окончательно, но продвижение в указанных направлениях существенное;
- развитие математической логики. От метода резолюций Робинсона до развития нетрадиционных логик (немонотонных, правдоподобного вывода и т.д.).

Значительное влияние ИИ оказал на развитие индустрии программирования. В частности язык ЛИСП был предложен Маккарти для решения задач именно ИИ, а сейчас используется более широко. Парадигма объектно-ориентированного программирования сложилась в ходе проведения работ по ИИ.

### **1.2. Перспективы и задачи вычислительного интеллекта**

Анализ показывает сближение представителей разных направлений ИИ: традиционная обработка знаний и естественные языки; нейронные сети; распознавание и синтез речи; многоагентные системы. Сближение связано не только с естественными циклами развития любой науки, но и с тенденцией глубинной интеграции. На наш взгляд, важнейшая тенденция в развитии ИИ ближайшего



будущего – это *интеграция* автономных достижений, причем глубинная интеграция, осуществляемая на основе создания гибридных систем.

Трудно сказать под каким именем будет происходить такая широкая гибридизация, возможна регенерация термина *гибридные системы* или развитие термина *синергетический интеллект*.

**В любом случае впереди - гибридизация достижений инженерии знаний, распознавания образов, нейроинформатики, роботики, которая должна обеспечить синергизм сигнала, числа, слова и понятия.**

В самом общем виде такую тенденцию можно выразить как *переход от задачи создания думающей машины к задаче создания интеллектуального существа*.

#### 1.2.1. Моделирование восприятия и умозаключений. Интеграция нейронных сетей и систем логического вывода – нечеткие нейронные сети

Исследование известных архитектур гибридных систем и алгоритмов их обучения позволяет, по нашему мнению, считать их моделями, адекватно отражающими соотношение восприятия и логического умозаключения при когнитивной деятельности человека. При реальном решении задачи человек комбинирует процессы мышления (компьютерный аналог – логический вывод), воспоминания (сознательное «внутреннее» возбуждение образа), восприятия окружающих предметов (распознавание) и, возможно, движения (управление телом). Обученная нечеткая нейронная сеть и хранит паттерны, и выполняет логические операции, так как содержит И-ИЛИ-нейроны, и переключается от распознавания на логический вывод, так как обычно включает в себя несколько отдельно обученных сетей с разными функциями.

Таким образом, несмотря на то, что нечеткие нейронные сети созданы как нейро-нечеткие контроллеры для управления процессами, они могут служить когнитивной моделью для изучения взаимодействия процессов восприятия и мышления при когнитивной деятельности.

#### 1.2.2. Моделирование сознания и неосознанного

Особенностью гибридных систем является их принципиальная интерпретируемость, то есть как и всякая система логического вывода, гибридная система объясняет свой результат с помощью обратного просмотра протокола применяемых вербализованных правил. Любая нечеткая нейронная сеть работает как система нечеткого логического вывода, но строится не с помощью инженерии знаний, а с помощью обучения по образцам. В результате матрица весов отражает силу связи входных и выходных переменных. Результатом обучения нечеткой нейронной сети служит не только матрица весов, но совокупность правил и оценок их достоверности. Следовательно, любая гибридная система является как минимум двухуровневой, включающей систему и метасистему,

отражающую систему первого уровня. Исходя из сказанного, можно предложить нечеткие нейронные сети в качестве когнитивной модели соотношения сознательного (логического) и бессознательного (аналогового, вычислительного) процессов в решении интеллектуальной задачи.

Таким образом, в нечеткой нейронной сети возможно сочетать манипулирование образами, заданными количественными параметрами, с преобразованием символов (слов). Такая возможность заложена в самой базовой конструкции теории нечетких множеств. Каждое нечеткое множество связывает слово, имя с порядковой или метрической шкалой с помощью функции принадлежности, т. е. количественно моделирует новое качество – смысл.

### 1.2.3. Интеграция нейронных сетей и вероятностных вычислений

Интеграция НС и вероятностных вычислений привела к созданию и широкому использованию радиальных базисных сетей, сетей регрессии, вероятностных сетей, байесовских сетей. Первой особенностью таких сетей является параллельное формирование не только решения, но и вероятностной оценки такого решения. Например, сети регрессии, которые используют в основном для классификации, не только соотносят с классом входной вектор, но и формируют вероятность такого классифицирования.

Таким образом, вероятностные нейронные сети сочетают собственно распознавание образа с формированием его оценки и могут служить для моделирования восприятия.

Второй особенностью вероятностных НС служит сложная послойная структура, так как разные слои содержат разные типы нейронов. Такая структура приводит и к сложным многошаговым алгоритмам обучения, которые часто включают в себя соревновательное обучение по Кохонену, обучение на основе штрафов, по алгоритму обратного распространения ошибки. Такие алгоритмы обучения далеки от минимизации функции ошибки с помощью метода градиентного спуска, они комбинируют кластеризацию образцов, поиск зависимостей, то есть моделируют обобщение, порождение понятий в ходе обучения.

## 1.3. Синергетический компьютер

Особенность человеческого восприятия – это способность понимать искаженную информацию (ассоциативная память). Глядя на образ, мы выделяем примитивы, которые порождают параметр порядка. В результате процесса самоорганизации и при подчинении параметру порядка человеческий мозг воссоздает целый образ и распознает объект. На основе теории синергетики (нелинейной динамики) создан синергетический компьютер в форме алгоритма. Синергетический компьютер – это программа распознавания человеческих лиц, основанная на принципах самоорганизации. Такой компьютер проверен на искажениях, амбивалентных изображениях.

Нейрокомпьютер и синергетический компьютер – конкуренты в организации параллельных процессов. Г. Хакен в [2] называет синергетический компьютер альтернативой нейронному пути познания когнитивных возможностей. Характеристики синергетического компьютера следующие:

- Основная гипотеза: психическая деятельность мозга протекает в соответствии с основными принципами самоорганизации.
- Объект исследований – зрительное восприятие.

**Синергетический компьютер – модель самоорганизации – мост между нейронным микромиром и макроскопическим восприятием.**

#### **1.4. Аттракторные нейронные сети**

Детерминированный хаос – стереотип поведения многих синергетических систем, например, игровые автоматы часто используют запланированный хаос. Как хаос связан с синергетикой? Синергетическая система может управлять не одним, а сразу несколькими параметрами порядка. Параметры сотрудничают, конкурируют, один из них может доминировать, но смена доминанты может быть хаотической. Следовательно, хаотическими являются те процессы, которые при малейшем изменении условий полностью изменяются.

Хаос обладает двумя особенностями:

- чувствительность к исходным условиям;
- самоподобие.

Аттрактор – недостижимая точка притяжения состояния системы.

Почему аттракторные НС имеют перспективы в решении интеллектуальных задач ?

Из детерминированного и микроскопического хаоса возникает порядок. Предъявленный сенсорам образ, организует НС и позволяет извлечь нужный паттерн. Например, рассмотрим применение аттракторной нейронной сети в анализе временных рядов. Необходима реконструкция аттракторов из временного ряда. В некоторый момент времени  $t_1$  определено значение  $x$ , второе значение определяется в момент, смещенный относительно первого на  $T$ . Таким образом, обрабатываются все точки ряда, строится траектория на плоскости  $X(t_1)$ ,  $X(t_1+T)$ . По траектории можно проследить расположение аттрактора, т. е. выполнить экстраполяцию. В случае хаотических аттракторов для его реконструкции требуется, по меньшей мере, трехмерная система координат (а то и большей размерности). Соответствующие новые координаты получают при этом смещением временной оси не только  $T$ , но и  $2T$  и т.д. Проблема выбора смещения  $T$  не решена и определяется эмпирически.

## 1.5. Искусственный, естественный и нейроинтеллект

Естественный интеллект сформировался в ходе многомиллиардной эволюции. Искусственный интеллект создается в научном сообществе, довольно нескоординированном, последние 40 лет. Если учесть, что человечество не знает ответа на вопрос, как работает человеческий мозг, то силы явно не равны. Не существует общепризнанного определения естественного интеллекта. Разброс мнений колеблется от «нечто, что измеряется в интеллектуальных тестах», до «способности решать задачи с неизвестным алгоритмом решения».

Для настоящего момента развития ИИ характерно внимание к эволюционному процессу развития жизни, который и создал естественный интеллект. Создаются системы, моделирующие эволюцию, но не живых организмов, а машин. Однако такой способ создания искусственного разума потребует геологических эпох, несмотря на быстрое действие компьютеров.

От примитивной *лабиринтной модели мышления* ИИ быстро перешел к моделированию свойств естественного интеллекта, проявляемых в ходе *фило* и *этногенеза* интеллекта. Филогенез естественного интеллекта представляет собой сложное взаимодействие развития движений и пространственных моделей окружающего мира. *Телевосприятие* и *теледействие* потребовали развития *моторики*, а органы движения живого представляли собой пространственную модель мира. Осуществление все более сложных движений привело к развитию систем управления, то есть различных оценочных функций, *безусловных и условных рефлексов*. Усложнение движений и поведения в целом привело к формированию сложных врожденных программ движения и поведения - *инстинктов*. Наука в современном состоянии не может ответить на вопрос, когда и как появилась *память*, но ее появление позволило копить и использовать прижизненный опыт. Рано появившаяся *асимметрия мозга* привела к разделению функций, к формированию оценки (осознания) интеллектуальной деятельности и в конце концов к формированию психики и сознания. Данные *онтогенеза* (Пиаже) объясняют явление *интроспекции*, которое заключается в том, что при освоении какой-то деятельности ребенок помогает себе движениями, которые все менее активно используются при дальнейшем росте. Например, решая лабиринтную задачу, ребенок водит пальцем, а взрослый человек «следит глазами». Приведенные данные говорят о связи движения, поведения, жизни с появлением интеллекта. Иначе говоря, чтобы машины получили интеллект в ходе эволюции, они должны начать жить.

### 1.5.1. Баланс правополушарного и левополушарного

В литературе встречаются схемы гибридизации нейроинформатики и ИИ, построенные по следующему принципу: правое полушарие – нейрокомпьютер; левое полушарие – основанная на знаниях система, а вопрос лишь в их взаимодействии или балансе право- и лево-полушарности.

В реальном поведении человека невозможно разделить восприятие и логическую обработку, поэтому более успешной представляется схема глубинной интеграции. Примерами таких глубинных успешных гибридов может служить создание *нечетких нейронных сетей*, объединяющих методы обучения нейронных сетей, управление неопределенностью нечеткой логики и логический вывод дедуктивных систем. Причем гибридизация является именно глубинной, так как одна нейронная сеть содержит и логические (дискретные) И-ИЛИ-нейроны, и непрерывные традиционные пороговые суммирующие нейроны. Таким образом, происходит создание структуры, объединяющей обработку сигнала и оценку такой обработки, в том числе логическую [7, 8, 9, 10, 11, 12, 13].

### **1.6. Искусственная эволюция интеллектуальных систем**

Генетические алгоритмы различных видов в основном используются в нейронных сетях и гибридных системах для оптимизации параметров функций принадлежности входных и выходных переменных. Но появляются работы, в которых генетический алгоритм выполняет структурный синтез нейронной сети, в частности, определяет тип сети, количество скрытых слоев и нейронов, выбирает функции оптимальности, то есть с помощью компьютерного моделирования можно наблюдать разработку (развитие) нейронной сети (когнитивной структуры) в ходе искусственно организованной эволюции.

Для всех классов нейронных сетей, в том числе гибридных систем, связь системы с окружением устанавливается с помощью обучающей выборки. Чем более полна и адекватна обучающая выборка, тем выше аппроксимирующая или классифицирующая способность нейронной сети после обучения. При использовании, кроме классического метода обратного распространения сигнала, еще и генетического алгоритма обучения, гибридные системы получают еще один инструмент адаптации – функцию оптимальности генетического алгоритма, *fitness-function*.

Таким образом, нечеткие нейронные сети с генетическим алгоритмом оптимизации структуры можно рассматривать как перспективные модели компьютерного моделирования эволюции интеллекта. Эволюция интеллекта по современным представлениям может быть охарактеризована формулой:

**Моторика + Сенсорика + Безусловные рефлексy  
+ условные рефлексy + инстинкты + мышление = интеллект.**

Взаимодействие всех названных в формуле систем подразумевает сочетание восприятия и логики. Вместе с тем реализация решения интеллектуальных задач в ходе непрерывной эволюции интеллекта создала сложные структуры (мозг), огромное число элементов которых представляют собой взаимодей-

ствующую (синергетическую) систему. Гибридные системы позволяют изучать проявление таких когнитивных параметров порядка, как обучающие паттерны, в самоорганизующихся структурах.

### **Заключение**

Перспективой развития гибридных систем считают разработку синергетического компьютера, построенного как взаимодействующая система. В то же время можно отметить работы, считающие синергетику перспективной моделью мозга. Следовательно, синергетический компьютер, как результат развития нейроинформатики и синергетическая модель мозга, как направление в когнитивной науке имеют возможность развиваться взаимодействуя, благодаря общему вектору развития.

### **Библиографический список**

1. Ярушкина Н. Г. Основы теории нечетких и гибридных систем. М.: Финансы и статистика, 2004.- 320 с. Ил.
2. Хакен Г. Тайны природы. Синергетика: учение о взаимодействии.- Москва-Ижевск: Институт компьютерных исследований, 2003, 320 с.
3. Хакен Г., Хакен-Крелль Г. Тайны восприятия. Синергетика как ключ к мозгу. Москва: Институт компьютерных исследований, 2002, 320 с.
4. Поспелов Д. А. Становление информатики в России// Новости искусственного интеллекта, 1999, № 1
5. Гаврилова Т.А., Хорошевский В.Ф. Базы знаний интеллектуальных систем. СПб: Питер, 2000.- 384 с.
6. Эндрю А. Реальная жизнь и искусственный интеллект// Новости искусственного интеллекта, 2000, № 1-2.
7. Ярушкина Н. Г. Гибридные системы, основанные на мягких вычислениях. определение. архитектура. теоретические возможности и опыт практического использования // Программные продукты и системы, № 3, 2002, с. 19-22
8. Ярушкина Н. Г., Наместников А. М. Эффективность генетических алгоритмов для задач автоматизированного проектирования // Известия РАН. Теория и системы управления, № 2, 2002, с. 127-134
9. Ярушкина Н. Г. Вычислительный интеллект: синергизм слова и числа // Информационные технологии и вычислительные системы, № 4, 2002
10. Ярушкина Н. Г. Нечеткие и гибридные системы: обзор итогов и тенденций развития // Новости искусственного интеллекта, № 1, 2004
11. Ярушкина Н.Г. Нечеткие нейронные сети с генетической настройкой. Лекция научной школы конференции «Нейроинформатика-2004», М.: МИФИ, 2003

## Глава 2. МЯГКИЕ ВЫЧИСЛЕНИЯ В МОДЕЛИРОВАНИИ СИСТЕМ

### 2.1. Использование генетических алгоритмов для параметрической идентификации моделей

#### 2.1.1. Введение

Возрастающий интерес к искусственным системам, способным поддерживать высокое качество функционирования в условиях неопределенности и непредсказуемости, со всей очевидностью показал необходимость заимствовать механизмы адаптации из биологии, эволюции и генетики для улучшения поведения синтезируемой системы. Справедливость такого подхода подтверждается тем фактом, что стандартные модели математического программирования, которые по своей природе также являются искусственными, не всегда гарантируют хорошие результаты: быструю реакцию, кратковременный переходный процесс и высокую точность в установившемся режиме ценой низких вычислительных затрат.

Многие нестандартные подходы, такие как нечеткие технологии, нейронные сети и эволюционные вычисления к настоящему времени оформились в единое направление под названием «Интеллектуальные методы», или «Вычислительный интеллект» (ВИ). Последние три десятилетия отмечены возрастающим интересом к этой области. Начиная с семидесятых годов прошлого столетия, методы ВИ с успехом применяются к большому числу проблем. Так, совместное использование технологии экспертных систем и нечеткой логики позволило сократить размер базы данных в десятки раз [1]. Еще один подход к оптимизации и обучению интеллектуальных систем оказался тесно связан с разработкой смешанных технологий нечетких нейронных структур [2, 3]. Современные программно-инструментальные средства специального назначения позволяют не только имитировать системы во всех деталях на стадии разработки, но также оценивать эффективность принятых проектных решений, основанных на той или иной интеллектуальной технологии. Например, пакет прикладных программ *WinFACT* (Windows Fuzzy and Control

Tools) предоставляет средства перехода от классического проектирования к нейронно-сетевой версии нечеткого управления [4].

Вместе с тем среди специалистов в области прикладной и вычислительной математики сохраняется некоторое недоверие к новым, нестандартным методам оптимизации систем. Этот факт стимулирует сравнительное изучение стандартных и генетических подходов. Генетические методы представляют особый интерес благодаря их способности решать сложную задачу оптимизации, используя простую и относительно недорогую модель.

В данной работе проблема оптимизации дискретного фильтра рассматривается в применении к следующим системам [5]. Экспериментально сравниваются два подхода к идентификации оптимального фильтра в составе стохастической системы управления, работающей в условиях неопределенности:

- 1) метод вспомогательного функционала качества, основанный на классических численных методах оптимизации [6] и
- 2) нечисленная оптимизация фильтра, использующая генетические алгоритмы [7, 8].

Работа построена следующим образом. Пункт 2.1.2 описывает проблему, включая данную систему, фильтр Калмана и адаптивный фильтр. Общая схема процесса идентификации фильтра Калмана дана в пункте 2.1.3. Краткое описание генетических алгоритмов приведено в пункте 2.1.4. Пункт 2.1.5 содержит некоторые результаты вычислительных экспериментов. Пункт 2.1.6 завершает работу.

### 2.1.2. Постановка задачи

Для построения и изучения адаптивной следящей системы выразим задачу в терминах расширенного состояния <sup>1</sup>

$$x_a(\cdot) \doteq \begin{bmatrix} x(\cdot) \\ x_r(\cdot) \end{bmatrix} \in \mathbb{R}^n,$$

формируемого в дискретном времени уравнением

$$\begin{bmatrix} x(t_{i+1}) \\ x_r(t_{i+1}) \end{bmatrix} = \begin{bmatrix} \Phi & 0 \\ 0 & \Phi_r \end{bmatrix} \begin{bmatrix} x(t_i) \\ x_r(t_i) \end{bmatrix} + \begin{bmatrix} \Psi \\ 0 \end{bmatrix} u(t_i) + \begin{bmatrix} w(t_i) \\ w_r(t_i) \end{bmatrix}, \quad (2.1)$$

$$\mathbf{E} \left\{ \begin{bmatrix} w(t_i) \\ w_r(t_j) \end{bmatrix} \begin{bmatrix} w^T(t_i) & w_r^T(t_j) \end{bmatrix} \right\} = \begin{bmatrix} Q & 0 \\ 0 & Q_r \end{bmatrix} \delta_{i,j} \quad (2.2)$$

---

<sup>1</sup> Обозначение « $\doteq$ » вместо « $=$ » означает в этом подразделе «равенство по определению».



с управляющим воздействием  $u(\cdot)$  и белыми шумами  $w(\cdot)$ ,  $w_r(\cdot)$  с нулевым средним значением. Состояние  $x_a$  наблюдается посредством вектора

$$z_a(\cdot) \doteq \begin{bmatrix} z(\cdot) \\ z_r(\cdot) \end{bmatrix} \in \mathbb{R}^m,$$

$$z_a(t_i) = \begin{bmatrix} z(t_i) \\ z_r(t_i) \end{bmatrix} = \begin{bmatrix} H & 0 \\ 0 & C_r \end{bmatrix} x_a(t_i) + \begin{bmatrix} v(t_i) \\ v_r(t_i) \end{bmatrix}, \quad (2.3)$$

$$\mathbf{E} \left\{ \begin{bmatrix} v(t_i) \\ v_r(t_j) \end{bmatrix} \begin{bmatrix} v^T(t_i) & v_r^T(t_j) \end{bmatrix} \right\} = \begin{bmatrix} R & 0 \\ 0 & R_r \end{bmatrix} \delta_{i,j} \quad (2.4)$$

с белыми шумами  $v(\cdot)$ ,  $v_r(\cdot)$  с нулевым средним значением, где  $\delta_{i,j}$  – символ Кронекера.

Система проектируется с целью сведения к нулю ошибки слежения

$$e(t_i) \doteq C_a \begin{bmatrix} x(t_i) \\ x_r(t_i) \end{bmatrix}, \quad C_a \doteq \begin{bmatrix} C & -C_r \end{bmatrix}, \quad (2.5)$$

что должно быть выполнено минимизацией среднеквадратической функции стоимости

$$J_{\text{control}} \doteq \lim_{t_0 \rightarrow -\infty} \mathbf{E} \left\{ \frac{1}{2} \sum_{j=0}^i \begin{bmatrix} x_a(t_j) \\ u(t_j) \end{bmatrix}^T \begin{bmatrix} X_a & S_a \\ S_a^T & U \end{bmatrix} \begin{bmatrix} x_a(t_j) \\ u(t_j) \end{bmatrix} \right\} \quad (2.6)$$

с  $X_a \doteq C_a^T Y C_a$  и  $S_a \doteq C_a^T S$ , где матрица  $S$  может быть не равна нулю, и с некоторыми постоянными матрицами  $Y > 0$  и  $U > 0$ , такими, что составная матрица в (2.6) является положительно полуопределенной. Таким образом, следящая система определяется как система, состоящая из управляемого объекта и опорной модели при условии, что некоторые управляемые переменные  $y_c(\cdot) = Cx(\cdot)$  следят за заданными опорными переменными  $y_r(\cdot) = C_r x_r(\cdot)$ .

Ограничиваясь рассмотрением инвариантных во времени систем, можно найти постоянные коэффициенты усиления  $G_{c1}$  и  $G_{c2}$  в законе обратной связи

$$u(t_i) = - \begin{bmatrix} G_{c1} & G_{c2} \end{bmatrix} \begin{bmatrix} x(t_i) \\ x_r(t_i) \end{bmatrix} = -G_{c1}x(t_i) - G_{c2}x_r(t_i). \quad (2.7)$$

Этот закон [5] мог бы быть использован, если бы имелись в наличии истинные значения  $x(t_i)$  и  $x_r(t_i)$ . Поскольку эти значения не доступны, их заменяют на соответствующие оценки  $\hat{x}(t_i^+)$  и  $\hat{x}_r(t_i^+)$ , получаемые при помощи фильтра Калмана, и подставляют их в (2.7) вместо  $x(t_i)$  и  $x_r(t_i)$ , соответственно. Для того чтобы фильтр оказался разделен на два абсолютно независимых

фильтра, предполагается некоррелированность  $x(t_0)$  и  $x_r(t_0)$ ; такое же требование принято, соответственно, в отношении  $w(\cdot)$  и  $w_r(\cdot)$  в (2.1) и  $v(\cdot)$  и  $v_r(\cdot)$  в (2.3). Кроме того, чтобы гарантировать существование устойчивых фильтров, предполагаем, что матрица  $\Phi_r$  — *асимптотически устойчивая* (все ее собственные значения лежат строго внутри единичного круга на комплексной плоскости), пара  $(\Phi, Q^{1/2})$  — *стабилизируемая*, пара  $(\Phi, H)$  — *наблюдаемая*, и пара  $(\Phi, \Psi)$  — *управляемая*. Также для использования метода вспомогательного функционала качества (ВФК) при идентификации оптимальных фильтров [9, 10] предполагаем, что обе подсистемы в (2.1), (2.3) даны в виде *стандартной наблюдаемой модели*, СНМ.

При сделанных предположениях устойчивый фильтр Калмана (SSKF) определен уравнениями

$$\hat{x}_a(t_{i+1}^-) = \Phi_a \hat{x}_a(t_i^+) + \Psi_a u(t_i), \quad (2.8)$$

$$\hat{x}_a(t_i^+) = \hat{x}_a(t_i^-) + K_a[z_a(t_i) - H_a \hat{x}_a(t_i^-)] \quad (2.9)$$

с  $\Phi_a$  и  $\Psi_a$  из (2.1),  $H_a$  из (2.3), в которых  $K_a$  вычисляется при помощи рекуррентного уравнения Риккати [5]. Вычисляемая таким образом оценка  $\hat{x}_a(t_i^+) = [\hat{x}(t_i^+)^T : \hat{x}_r(t_i^+)^T]^T$  нужна для использования вместо  $x_a(t_i)$  в (2.7).

Рассматриваемая проблема обусловлена наличием параметрической неопределенности в уравнениях системы (2.1), (2.2), (2.3) и (2.4). Остановимся на первом уровне неопределенности.

**Случай 1:** Четыре матрицы в описании системы, названные  $Q$ ,  $R$ ,  $Q_r$  и  $R_r$  в (2.2) и (2.4), составляют вектор неопределенности  $\theta \in \Theta$ <sup>2</sup>. Каждое конкретное значение  $\theta$  определяет некоторый *режим*.

Оценки коэффициентов Калмана  $K$  и  $K_r$  в  $K_a = \text{diag}[K \mid K_r]$  обозначим соответственно как  $\bar{K}$  и  $\bar{K}_r$  в  $\bar{K}_a = \text{diag}[\bar{K} \mid \bar{K}_r]$  для субоптимального фильтра (SOF)

$$\bar{x}_a(t_{i+1}^-) = \Phi_a \bar{x}_a(t_i^+) + \Psi_a u(t_i), \quad (2.10)$$

$$\bar{x}_a(t_i^+) = \bar{x}_a(t_i^-) + \bar{K}_a[z_a(t_i) - H_a \bar{x}_a(t_i^-)] \quad (2.11)$$

и как  $\tilde{K}$  и  $\tilde{K}_r$  в  $\tilde{K}_a = \text{diag}[\tilde{K} \mid \tilde{K}_r]$  для адаптивного фильтра (AF)

$$\tilde{x}_a(t_{i+1}^-) = \Phi_a \tilde{x}_a(t_i^+) + \Psi_a u(t_i), \quad (2.12)$$

$$\tilde{x}_a(t_i^+) = \tilde{x}_a(t_i^-) + \tilde{K}_a[z_a(t_i) - H_a \tilde{x}_a(t_i^-)]. \quad (2.13)$$

Все AF, определенные в (2.12) и (2.13), различаются в соответствии с видом алгоритма адаптации (выбора)  $\tilde{K}_a$ . Для этого случая сравним два алгоритма: стандартный численный алгоритм (ЧА) и генетический алгоритм (ГА).

---

<sup>2</sup>  $\Theta$  — компактное множество, где уравнения SSKF (2.8), (2.9) существуют.

### 2.1.3. Общая схема процесса идентификации

Стандартные методы идентификации, такие, как стохастические аппроксимация (процедура Роббинса-Монро), метод наименьших квадратов и многие другие широко представлены в литературе [6]. Для сравнения выберем один из них — метод вспомогательного функционала качества (ВФК), разработанный для адаптивной фильтрации [9] и распространенный на задачи управления [10].

Чтобы связать эту работу с [9, 10], переобозначим переменные (2.12), (2.13) на

$$g(t_{i+1|i}) \doteq \tilde{x}_a(t_{i+1}^-), \quad g(t_{i|i}) \doteq x_a(t_i^+), \quad \eta(t_{i|i}) \doteq z_a(t_i) - H_a \tilde{x}_a(t_i^-) \quad (2.14)$$

и обозначим настраиваемый (регулируемый) параметр АФ как вектор-столбец

$$\hat{\theta} \doteq \text{col} \left[ \underbrace{\hat{\theta}_{11}, \dots, \hat{\theta}_{1m}; \dots; \hat{\theta}_{n1}, \dots, \hat{\theta}_{nm}}_{\text{все } p \text{ элементов матрицы } \tilde{K}_a} \right] \in \mathbb{R}^p, \quad \hat{\theta}_{ij} \doteq [\tilde{K}_a]_{ij}, \quad p = mn. \quad (2.15)$$

Как известно, метод ВФК сводится к методу МРЕ<sup>3</sup>, исключая одну особенность: он использует идею эквивалентного замещения *исходного функционала качества* (ИФК) (который является прямым, но не доступным) некоторым *вспомогательным функционалом качества* (который является доступным, хотя и косвенным). Главным требованием для такого замещения является то, что ИФК и ВФК *эквивалентны*, то есть они имеют один и тот же минимизирующий аргумент. Таким образом, исходя из метода ВФК, выполняем следующее:

1. Строим модель чувствительности (SM), беря частные производные от уравнений (2.12)–(2.13) для получения рекуррентных уравнений функций чувствительности

$$\mu_{jl}(t_{i+1|i}) \doteq \frac{\partial}{\partial \hat{\theta}_{jl}} g(t_{i+1|i}), \quad j = 1, \dots, n; \quad l = 1, \dots, m.$$

---

<sup>3</sup> Minimum Prediction Error — минимум ошибки предсказания.

2. Для  $j = 1, \dots, n$  ;  $l = 1, \dots, m$  , вычисляем

$$\begin{aligned} \eta_{i-s+1|i-s}^{i|i-1} &= [\eta^T(t_{i-s+1|i-s}) \mid \dots \mid \eta^T(t_{i|i-1})]^T , \\ \varepsilon(t_i, \hat{\theta}) &= \mathcal{P}(\tilde{K}_a) \eta_{i-s+1|i-s}^{i|i-1} , \end{aligned} \quad (2.16)$$

$$\begin{aligned} \frac{\partial \eta(t_{i|i-1})}{\partial \hat{\theta}_{jl}} &= -H_a \mu_{jl}(t_{i|i-1}) , \\ \frac{\partial \varepsilon(t_i, \hat{\theta})}{\partial \hat{\theta}_{jl}} &= \left[ \frac{\partial}{\partial \hat{\theta}_{jl}} \mathcal{P}(\tilde{K}_a) \right] \eta_{i-s+1|i-s}^{i|i-1} + \\ &+ \mathcal{P}(\tilde{K}_a) \frac{\partial}{\partial \hat{\theta}_{jl}} \eta_{i-s+1|i-s}^{i|i-1} , \end{aligned} \quad (2.17)$$

где  $\mathcal{P}(\cdot)$  — процедура, определенная в [9] следующим образом.

**Определение 2.1** Пусть  $s, p_1, \dots, p_m \in \mathbb{N}$  удовлетворяют условию

$$s \doteq \max(p_1, p_2, \dots, p_m) \leq p_1 + p_2 + \dots + p_m = n$$

и  $A$  —  $(ms \times k)$ -матрица,  $k \in \mathbb{N}$ , составленная из  $s$  подматриц с  $m$  строками и  $k$  столбцами каждая. Обозначим  $j$ -ю строку  $i$ -й подматрицы как  $a_i^j$  и переупорядочим все эти  $ms$  строк так, чтобы сформировать новую  $(s \times kt)$ -матрицу

$$A_T \doteq \begin{bmatrix} a_1^1 & \dots & a_1^m \\ \vdots & \ddots & \vdots \\ a_s^1 & \dots & a_s^m \end{bmatrix} .$$

Тогда  $\mathcal{S}(A)$ , что является  $(n \times k)$ -матрицей, называется  $\mathcal{S}$ -преобразованием матрицы  $A$ , если ее  $n$  строк получены взятием элементов  $a_i^j$  из  $A_T$  и помещением их в  $\mathcal{S}(A)$  как строк в следующем порядке:  $a_1^1, a_2^1, \dots, a_{p_1}^1$ ,  $a_1^2, a_2^2, \dots, a_{p_2}^2$ ,  $\dots$ ,  $a_1^m, a_2^m, \dots, a_{p_m}^m$ .

**Определение 2.2** Пусть  $(\Phi_\star, H_\star)$  — матрицы в СНМ такие, что  $\Phi_\star$  представлена в блочно-сопровождающем виде, где нетривиальные (ненулевые или неединичные) элементы расположены в строках с номерами  $p_1, p_1 + p_2, \dots, p_1 + p_2 + \dots + p_m = n$ , где  $p_j$  — частные индексы наблюдаемости пары  $(\Phi_\star, H_\star)$ , и  $H_\star$  есть  $(m \times n)$ -матрица,  $ij$ -й элемент которой есть 1, если  $j = p_1 + p_2 + \dots + p_{i-1} + 1$ , и 0 — в противном случае.

Тогда  $\mathcal{P}(D)$ , что является  $(n \times sm)$ -матрицей, называется  $\mathcal{P}$ -преобразованием  $(n \times m)$ -матрицы  $D$ , если

$$\mathcal{P}(D) \doteq \mathcal{S}(S_\theta(H_\star, \Phi_\star, \Phi_\star D))$$

со следующей  $(sm \times sm)$ -матрицей и  $G = \Phi_\star D$ :

$$S_\theta(H_\star, \Phi_\star, G) \doteq \begin{bmatrix} I & 0 & \cdots & 0 \\ H_\star G & I & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ H_\star \Phi_\star^{s-2} G & H_\star \Phi_\star^{s-3} G & \cdots & I \end{bmatrix}. \quad (2.18)$$

3. Упорядочим функции чувствительности, определенные в (2.17), в  $(n \times p)$ -матрицу чувствительности

$$S(t_i) \doteq \left[ \frac{\partial \varepsilon(t_i, \hat{\theta})}{\partial \hat{\theta}_{11}}, \dots, \frac{\partial \varepsilon(t_i, \hat{\theta})}{\partial \hat{\theta}_{n1}}, \dots, \frac{\partial \varepsilon(t_i, \hat{\theta})}{\partial \hat{\theta}_{nm}} \right]. \quad (2.19)$$

4. Построим модель градиента с коэффициентом экспоненциального сглаживания  $\beta$ :

$$\begin{aligned} \mathcal{G}(t_i) &= S^T(t_i) \varepsilon(t_i, \hat{\theta}), \\ \hat{\mathcal{G}}(t_i) &= \beta \hat{\mathcal{G}}(t_{i-1}) + (1 - \beta) \mathcal{G}(t_i). \end{aligned} \quad (2.20)$$

5. Проверим условие устойчивости (SC):  $\rho[(I - \tilde{K}_a H_a) \Phi_a] < 1$  для AF и SM, где  $\rho[\cdot]$  — обозначение для спектрального радиуса матрицы  $[\cdot]$ . Для этого нужно иметь характеристический многочлен матрицы  $(I - \tilde{K}_a H_a) \Phi_a$  в виде

$$q(\lambda) \doteq b_0 \lambda^n + b_1 \lambda^{n-1} + \cdots + b_{n-1} \lambda^1 + b_n, \quad b_0 > 0$$

с  $b_0, \dots, b_n$ , выраженными в виде некоторых функций  $b_k = b_k(\hat{\theta})$  от модельного параметра (2.15), и затем использовать критерий Джурри [11, разд. 3.3]. Если SC выполняется, запишем  $SC(\hat{\theta}) = \text{true}$  и  $SC(\hat{\theta}) = \text{false}$  — иначе, и назовем эту процедуру **checkSC** ( $\hat{\theta}$ ).

6. Используем **процедуру адаптации** (AP) для обновления параметра (2.15) внутри эффективного рабочего диапазона  $t_i$  от  $t_{\text{start}}$  к  $t_{\text{stop}}$ . Для AP можно использовать различные методы поиска оптимума, способные минимизировать функционал качества, принятый для характеристики качества AF (2.12)–(2.15). В роли такого функционала в этой работе используем ВФК

$$J(\tilde{\theta}) \doteq \frac{1}{2} \mathbf{E} \left\{ \varepsilon(t_i, \tilde{\theta})^T \varepsilon(t_i, \tilde{\theta}) \right\} \quad (2.21)$$

с обобщенной невязкой (2.16). Используя упрощенную процедуру наименьших квадратов (SLS) для минимизации (2.21), AP оформим следующим образом:

```

if ( $t_{\text{start}} \leq t_i < t_{\text{stop}}$ ) then
  begin if ( $t_{\text{start}} = t_i$ ) then
    begin  $k := 0$ ;  $\Lambda = I$  end else  $k := k + 1$ ;
    if ( $k > 0 \ \& \ k = 0 \bmod s$ ) then
      begin  $\tau := [k - 1/s]$ ; for  $j = 1$  to  $p$  do
         $\lambda_{\tau+1}^{(j)} := \lambda_{\tau}^{(j)} + \left\| \frac{\partial r(t_i, \hat{\theta}(\tau))}{\partial \hat{\theta}_j} \right\|^2$ ;  $\Lambda_{\tau+1} = \text{diag} \{ \lambda_{\tau+1}^{(j)} \}$ ;      \star SLS \star
         $\pi := \hat{\theta}(\tau) - \Lambda_{\tau+1}^{-1} \hat{\mathcal{G}}(t_i)$ ;      \star Num \star
        checkSC( $\pi$ );
        if ( $SC(\pi)$ ) then  $\hat{\theta}(\tau + 1) := \pi$ 
      end
    end
  end

```

**Замечание 2.1** Изменяя строку, отмеченную  $\backslash \star \text{SLS} \star \backslash$ , можно перейти к другому методу поиска минимума. Например, используя

$$\lambda_{\tau+1}^{(j)} := \lambda_{\tau}^{(j)} + 1,$$

переходим к методу простой стохастической аппроксимации  $\backslash \star \text{SSA} \star \backslash$ . Таким образом, вместе со строкой, помеченной  $\backslash \star \text{Num} \star \backslash$ , задаем конкретный численный метод оптимизации  $\hat{\theta}$ .

**Замечание 2.2** Изменяя только две отмеченные строки, можно перейти к генетическому методу.

**Замечание 2.3** В результате  $(s - 1)$ -шаговой задержки в схеме, показанной на рис. 2.5, временной шаг для AP становится в  $s$  раз больше, чем для системы. С этой целью выше введен AP-таймер  $\tau$  с использованием функции взятия целой части числа  $[\cdot]$ .

#### 2.1.4. Моделируемые генетические алгоритмы [7, 8, 12, 13]

Генетические алгоритмы (ГА) отличаются от других поисковых методов использованием принципов, заимствованных у генетики и эволюционной теории.

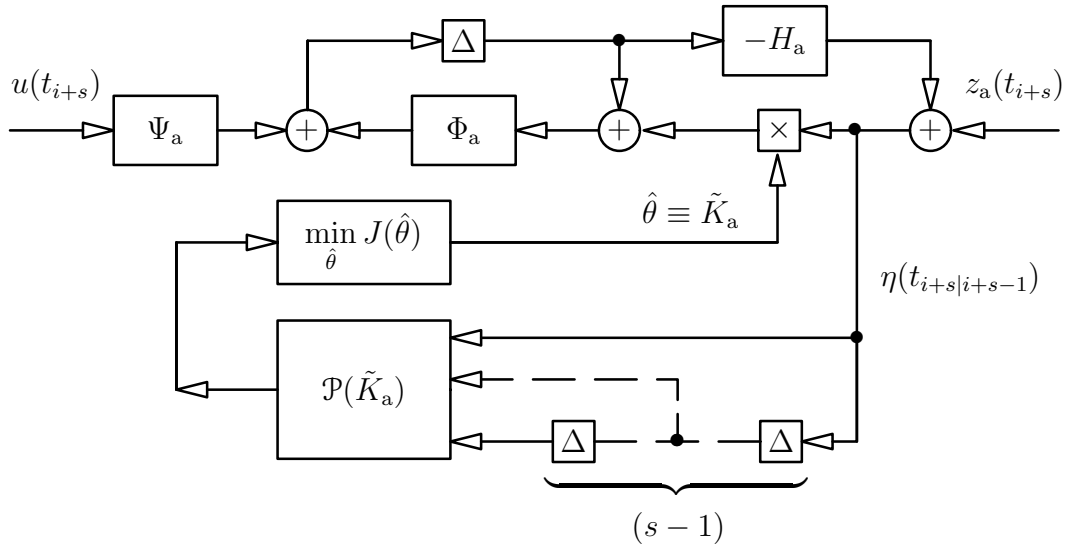


Рис. 2.1. Общая схема использования API.  $\Delta$  обозначает одношаговую задержку

Базовый элемент, обрабатываемый ГА, — строка, формируемая соединением подстрок, каждая из которых является бинарным кодом параметра поискового пространства [12]. Таким образом, каждая строка представляет собой возможное решение задачи. ГА работает с множеством строк, называемых *популяцией*. Эта популяция затем развивается от поколения к поколению посредством применения генетических операторов. ГА в своей простейшей форме использует следующие три оператора: репродукция (селекция, отбор), кроссовер (пересечение) и мутация.

**Репродукция.** Репродукция имитирует закон выживания сильнейшего. Значение приспособленности (пригодность),  $F(i)$ , связано с каждым индивидом в популяции, где бóльшие числа означают лучшую приспособленность. Функция пригодности (*фитнес-функция*) — это некоторая нелинейная, недифференцируемая, разрывная, положительная функция, поскольку алгоритму нужно иметь только значение приспособленности, связанное с каждой строкой. Сначала рассмотрим построение промежуточной популяции из текущей. Первое поколение текущей популяции является начальной популяцией. После вычисления  $F(i)/\bar{F}$  (где  $\bar{F}$  — среднее значение пригодности) для всех строк, в текущей популяции происходит отбор. В стандартном генетическом алгоритме вероятность того, что строки из текущей популяции будут скопированы (т. е. воспроизведены) и помещены в промежуточную популяцию, пропорциональна их приспособленности.

Имеется ряд вариантов оператора отбора. Популяцию можно рассматри-

вать как отображение на «колесо рулетки», где каждый индивид представлен пространством, которое пропорционально соответствует его пригодности. С каждым вращением колеса рулетки индивидуумы отбираются по правилу «стохастический выбор с заменой» для пополнения промежуточной популяции.

Процесс отбора, который в наибольшей степени соответствует ожидаемому значению пригодности, называется «остаточный стохастический отбор». Для каждой строки  $i$ , у которой  $F(i)/\bar{F}$  больше, чем 1.0, целая часть этого числа показывает, сколько копий этой строки непосредственно помещается в промежуточную популяцию. Все строки (включая те, для которых  $F(i)/\bar{F}$  меньше, чем 1.0) затем помещают дополнительные копии в промежуточную популяцию с вероятностью, соответствующей дробной части  $F(i)/\bar{F}$ . Например, строка с  $F(i)/\bar{F} = 1.36$  помещает первую копию в промежуточную популяцию и затем получает шанс с вероятностью 0.36 поместить вторую копию. Строка с приспособленностью  $F(i)/\bar{F} = 0.54$  имеет вероятность помещения одной копии в промежуточную популяцию, равную 0.54.

**Пересечение.** Отбор направляет поиск к лучшим существующим индивидам, но не создает новых индивидов. В природе потомки редко являются точными копиями родителей, они обычно имеют двух родителей и наследуют гены от обоих. Главный оператор, который работает с родителями — кроссовер. Он применяется с определенной вероятностью, называемой вероятностью кроссовера ( $p_c$ ). Кроссовер берет двух индивидов из промежуточной популяции и разрезает их хромосомы в некоторой случайно выбранной позиции, в результате появляются два «головных» сегмента и два «хвостовых» сегмента. Хвостовые сегменты затем обмениваются для производства двух новых хромосом полной длины:

$$\left\{ \begin{array}{cccc:cccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array} \right\} \times \Rightarrow \left\{ \begin{array}{cccc:cccc} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \end{array} \right\}.$$

Эта процедура известна как одноточечный кроссовер.

Двухточечный кроссовер использует две случайные точки пересечения. Строки обмениваются сегментами, которые лежат между этими двумя точками:

$$\left\{ \begin{array}{cc:cc:cc:cc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array} \right\} \times \Rightarrow \left\{ \begin{array}{cc:cc:cc:cc} 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \end{array} \right\}.$$



**Мутация.** Мутация применяется к каждому потомку индивидуально после пересечения. Она случайно изменяет каждый ген с малой вероятностью (обычно 0.001). Следующий пример показывает мутацию третьего гена в хромосоме:

$$\left\{ \begin{array}{cccccc} 0 & 0 & : & 0 & : & 0 & 0 \end{array} \Rightarrow \begin{array}{cccccc} 0 & 0 & : & 1 & : & 0 & 0 \end{array} \right\}.$$

#### 2.1.5. Некоторые результаты вычислительных экспериментов

Поведение ГА в процессе адаптации из пункта 2.1.4 показано на рис. 2.2. Для моделирования взята следящая система, описанная в [5]. В этом примере выбраны следующие постоянные величины:

$$\Phi = 0.82, \quad \Phi_r = 0.61, \quad \Psi = 0.18, \quad C = 1, \quad C_r = 1, \quad Y = 1, \quad U = 1$$

и  $S_a^T = [0 : 0]$  для LQG синтеза управления (2.7). Это дает

$$G_{c1} = [0.36], \quad G_{c2} = [-0.19].$$

Для синтеза фильтра Калмана имеем следующие данные:

$$\begin{aligned} H &= [1], \quad Q = [0.084\sigma^2], \quad Q_r = [0.63\sigma_r^2], \\ \sigma^2 &= \text{var}, \quad \sigma_r^2 = \text{var}, \quad R = \text{var}, \quad R_r = \text{var}, \end{aligned}$$

где «var» — обозначение переменных величин. При моделировании различаются две фазы, названные «до» и «после» (до и после переключения параметра). Фаза «до» занимает 300 моментов времени и фаза «после» занимает 10 000 моментов времени.

Эксперименты проведены со следующими параметрами:

1. Число итераций, (NI) ..... 10 000
2. Размер экспериментальной выборки (NES)  
для получения усредненных результатов, ..... 100
3. Моделирование численных алгоритмов и их параметров:
  - (a) Процедура Роббинса-Монро, ..... SSA
  - (b) Упрощенный метод наименьших квадратов, ..... SLS
  - (c) Параметр экспоненциального сглаживания  $\beta$ , ..... 0.5
4. Моделирование генетического алгоритма:

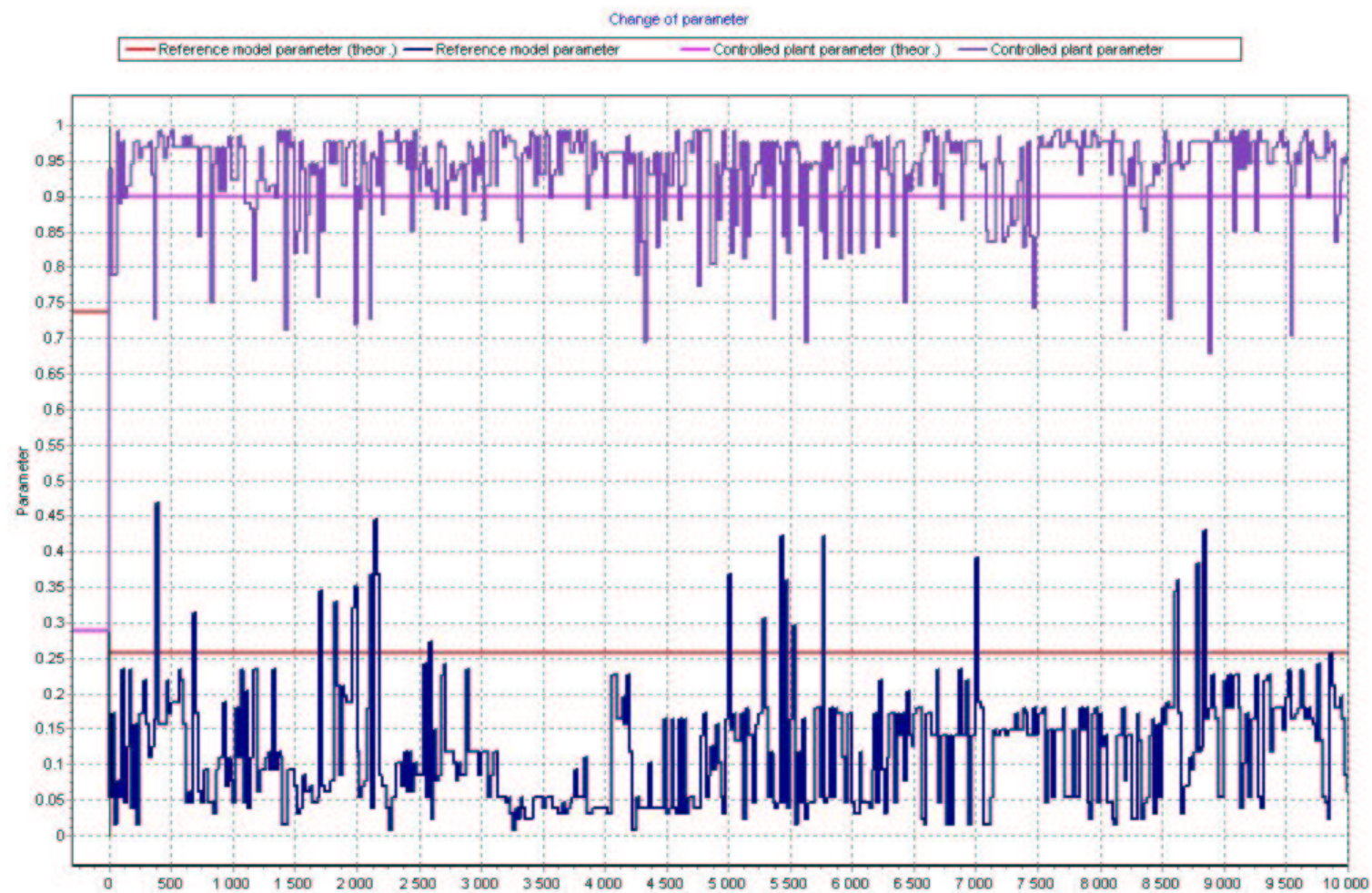


Рис. 2.2. Поведение генетического алгоритма. Случай 1

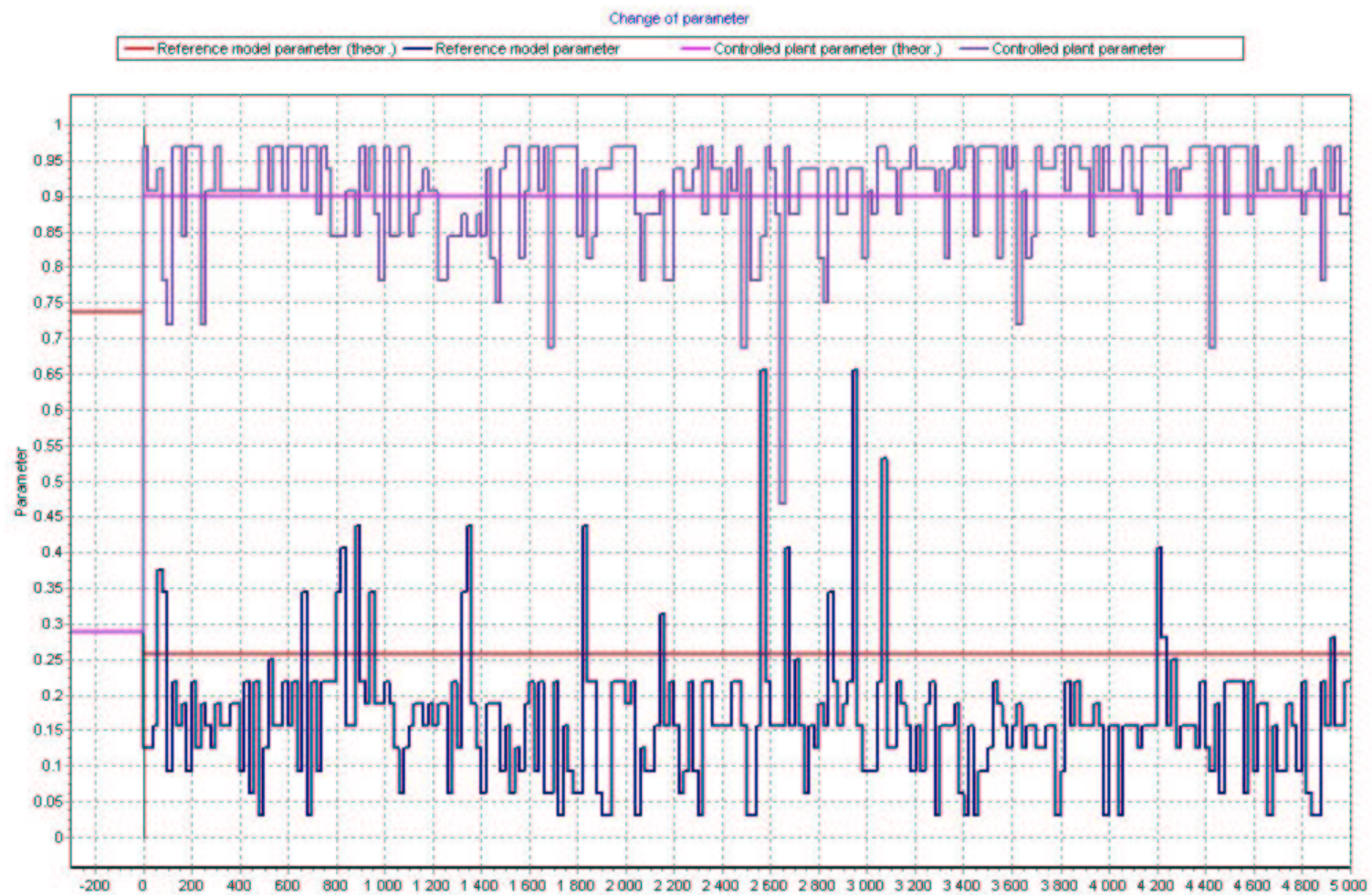


Рис. 2.3. Поведение ГА с модифицированными параметрами. Случай 1



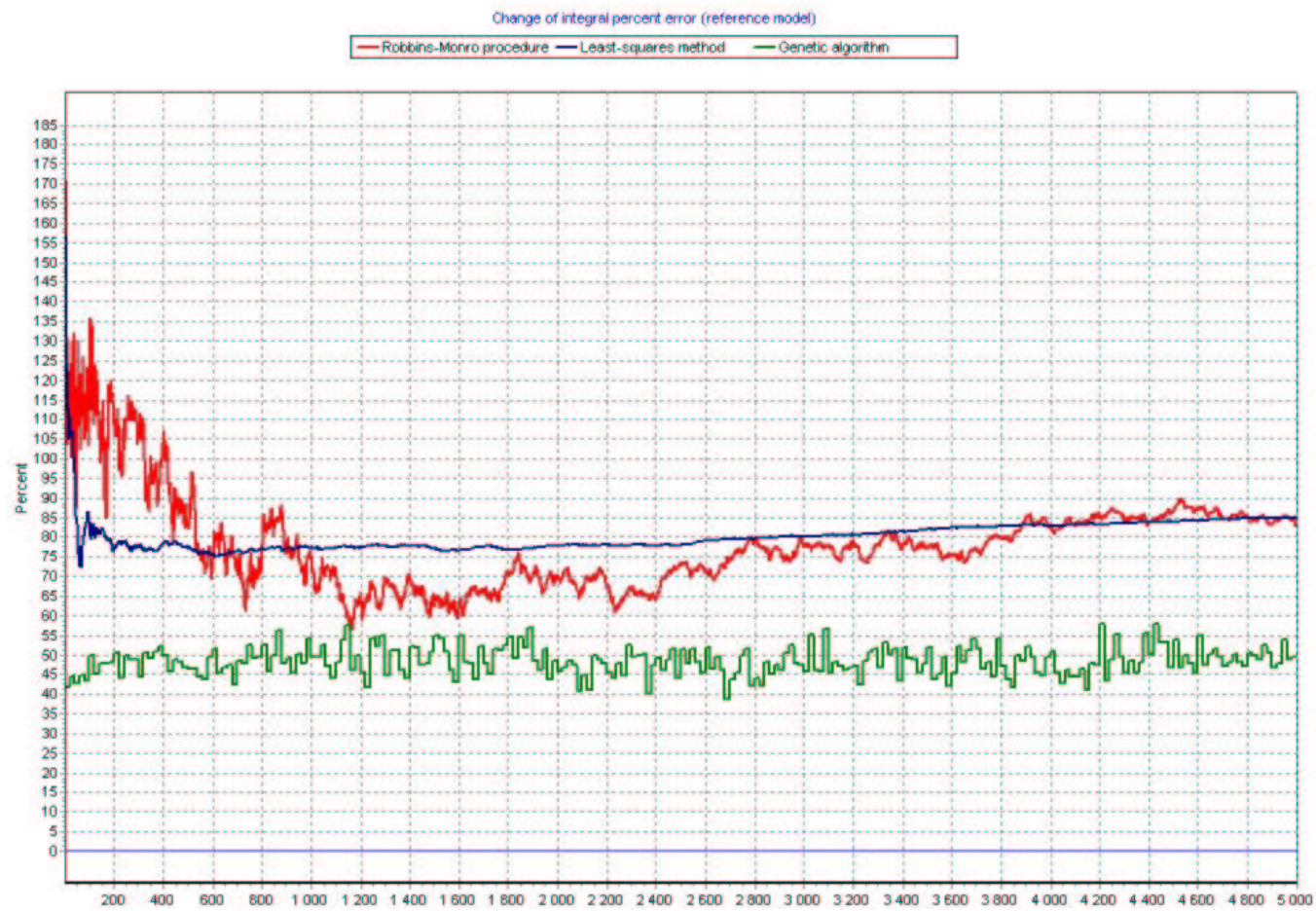


Рис. 2.4. Поведение IPE с ГА модифицированными параметрами. Случай 1

(a) Длина хромосомы, (CL) .....	7
(b) Мощность (размер) популяции, (PP) .....	50
(c) Коэффициент элитизма <sup>4</sup> , (EF) .....	2
(d) Вероятность мутации, (MPR) .....	0.100
(e) Метод селекции, (SM) .....	остаточный отбор
(f) Метод кроссовера, (CM) .....	двухточечный

В экспериментах рассмотрены следующие два случая.

### Случай 1.

«ДО»						«ПОСЛЕ»					
$\sigma^2$	$\sigma_r^2$	$R$	$R_r$	$K$	$K_r$	$\sigma^2$	$\sigma_r^2$	$R$	$R_r$	$K$	$K_r$
10.0	4.0	4.0	1.0	0.287	0.736	10.0	4.0	0.1	10.0	0.900	0.258

### Случай 2.

«ДО»						«ПОСЛЕ»					
$\sigma^2$	$\sigma_r^2$	$R$	$R_r$	$K$	$K_r$	$\sigma^2$	$\sigma_r^2$	$R$	$R_r$	$K$	$K_r$
10.0	4.0	4.0	10.0	0.287	0.258	10.0	4.0	0.1	1.0	0.900	0.736

Интегральная относительная ошибка (IPE) определяется (с усреднением по NES):

$$(a) \quad IPE(t_i) \doteq \frac{\|\tilde{K}_a(t_i) - K_a\|}{\|K_a\|} 100\% , \quad (2.22)$$

или

$$(b) \quad IPE(t_i) \doteq \frac{\|\tilde{K}(t_i) - K\|}{\|K\|} 100\% . \quad (2.23)$$

В другой серии экспериментов изменены следующие параметры: NI на 5000, NES на 50, CL на 5, PP на 30, и MPR на 0.200. Соответствующие результаты для Случая 1 показаны на рис. 2.3 – 2.4. IPE показана на графиках для всех трех методов: SSA – линией, занимающей среднее положение в интервале от 1000 до 2800, SLS – линией, занимающей верхнее положение в интервале от 1000 до 2800 и GA – линией, занимающей нижнее положение в этом интервале.

---

<sup>4</sup> Число хромосом, переходящих из текущей популяции в будущую популяцию без каких либо операций (отбор-пересечение-мутация) как считающихся самыми лучшими.

Как можно видеть из этих и многих других проведенных экспериментов, в большинстве случаев усредненное поведение ГА дает меньший уровень ИРЕ. Однако важно отметить, что индивидуальное поведение ГА более изменчиво, чем индивидуальное поведение численных методов.

Существенное отличие между этими типами методов, как и ожидалось, заключается в разных механизмах их поведения. Численные алгоритмы являются последовательными по принципу своего действия, тогда как генетический алгоритм — параллелен. Последнее требует множества индивидуумов, образующих текущую популяцию адаптивных фильтров, причем каждый индивидуум-фильтр действует по схеме рис. 2.5. Большой размер популяции приводит к возрастанию вычислительной нагрузки, что при обработке данных в режиме реального времени может привести к трудностям реализации.

### 2.1.6. Заключение

В этой работе сравнивается поведение генетического алгоритма с поведением классических численных методов оптимизации. Основу для этого сравнения составили стохастические следящие системы, в которых требуется проводить идентификацию оптимальных фильтров Калмана как в разомкнутом контуре обработки входных данных, так и в замкнутом контуре управления объектом. Необходимость идентификации (а значит, использования ГА или ЧА) связана с неопределенностью в ковариациях шумов, возбуждающих как формирующий фильтр полезного сигнала, так и управляемый объект.

Полученные результаты позволяют предположить, что ГА в среднем работают быстрее при поиске оптимума и с меньшей относительной ошибкой, чем ЧА. Однако использовать их нужно с осторожностью, поскольку их поведение во многом зависит от выбранной функции пригодности. В качестве такой функции в данном случае использован так называемый признак «статистической ортогональности» [14]:

$$\left\{ \begin{array}{ll} \mathbf{E} \{ \mathcal{G}(t_i) \} \doteq \mathbf{E} \left\{ S^T(t_i) \varepsilon(t_i, \hat{\theta}) \right\} = 0, & \text{когда АФ оптимален} \\ \mathbf{E} \{ \mathcal{G}(t_i) \} \doteq \mathbf{E} \left\{ S^T(t_i) \varepsilon(t_i, \hat{\theta}) \right\} \neq 0 & \text{иначе} \end{array} \right\},$$

выражающий необходимое и достаточное условие оптимальности адаптивного фильтра в форме вспомогательного (наблюдаемого) функционала качества.

Эта работа выполнена при частичной поддержке исследовательского гранта Министерства образования и науки РФ (грант № Т02-03.2-3427).

## 2.2. Метод взвешенных квадратов невязок для построения функций пригодности

### 2.2.1. Введение

Теория внезапных нарушений — интенсивно развиваемый раздел анализа временных рядов и идентификации [15]. Обычно под внезапными нарушениями подразумевают изменения в характеристиках системы, которые происходят в неизвестные моменты времени очень быстро (если не мгновенно) относительно интервала дискретной выборки при измерениях. Кроме чисто теоретического интереса, обнаружение таких изменений имеет большое практическое значение.

Ранее в большинстве случаев средства обнаружения и оценивания нарушений применялись при контроле качества, автоматической сегментации сигналов, контроле неисправностей на промышленных объектах, а также в алгоритмах пуска или останова идентификации для адаптивной и отказоустойчивой обработки данных и управления. Еще одно из возможных применений данных средств, которое появилось недавно, — репродукционно-селективные системы (РСС). Для генетических алгоритмов (ГА), используемых в РСС, требуется некоторая функция соответствия, уведомляющая ГА о том, кто из индивидов является наилучшим потомком внутри данной популяции. Селекция наилучшего потомка очень близка к распознаванию нескольких конкурирующих гипотез. На основании этого факта вполне естественно проводить поиск подходящей функции соответствия среди хорошо разработанных методов обнаружения.

Один из возможных методов обнаружения нарушений, который часто используется в технической литературе, основан на интуитивной идее обнаружения неаддитивного изменения по наблюдениям обновляющей последовательности (ОП)  $\nu(t_i)$  для авторегрессионной модели при помощи теста относительно его дисперсии  $\sigma^2$ . Кумулятивная сумма (CUSUM)

$$S_k = \frac{1}{\sqrt{k}} \sum_{i=1}^k s_i \quad \text{с} \quad s_i = \frac{1}{\sqrt{2}} \left( \frac{\nu(t_i)^2}{\sigma_0^2} - 1 \right) \quad (2.24)$$

асимптотически, когда  $k$  стремится к бесконечности, имеет стандартное нормальное распределение:  $S_k \sim \mathcal{N}(0; 1)$  при условии, что верна гипотеза  $\mathbf{H}_0 : \sigma^2 = \sigma_0^2$ .

Этот алгоритм, который называют *методом взвешенных квадратов невязок* (ВКН), был введен для авторегрессионной модели независимо в [16, 17] и [18] с целью автоматической сегментации сигналов электроэнцефалограмм.

Это испытание предполагает, что ОП — белый шум, и, таким образом, сначала нужна проверка этого свойства [19]. Хорошо известно, что ОП фильтра является белым шумом, если фильтр — калмановский (т. е. фильтр оптимален [5]), но из того, что ОП — белый шум, не следует оптимальность фильтра [20]. Автокорреляционные свойства ОП после нарушения могут быть очень сложными в зависимости от произошедшего изменения. Тем не менее наблюдение за ВКН представляет интерес для РСС из-за не слишком высоких вычислительных затрат, обещаая при этом предельное уменьшение затрат при большом размере популяции.

В данной работе формулируется и рассматривается метод ВКН в применении к стохастическим системам оценивания состояния и управления (пункты 2.2.2 и 2.2.3). В пункте 2.2.4 показан тест ВКН как полезный инструмент ГА для наилучшего отбора с точки зрения улучшения характеристик и сохранения работоспособности системы в меняющихся условиях. В пункте 2.2.5 характеризованы некоторые экспериментальные результаты обнаружения нарушения на примере инерциальной навигационной системы, применяемой в авиационном приборостроении. Пункт 2.2.6 описывает разработку программного обеспечения, пункт 2.2.7 резюмирует основное содержание, а Приложение с доказательством основных утверждений завершает работу.

### 2.2.2. Постановка задачи

Пусть система управления с обратной связью параметризована вектором  $\theta \in \mathbb{R}^p$ . Доступные данные  $z = \begin{bmatrix} y \\ u \end{bmatrix}$  рассматриваются как вектор, составленный из управляющего входа  $u \in \mathbb{R}^q$  и измеряемого выхода  $y \in \mathbb{R}^m$ . Система описывается при  $i \in \mathbb{Z}$  следующими уравнениями:

$$x(t_{i+1}) = \Phi_\theta x(t_i) + \Psi_\theta u(t_i) + w(t_i), \quad x \in \mathbb{R}^n \quad (2.25)$$

$$y(t_i) = H_\theta x(t_i) + v(t_i), \quad y \in \mathbb{R}^m \quad (2.26)$$

$$\hat{x}_0(t_{i+1}^-) = \Phi_0 \hat{x}_0(t_i^+) + \Psi_0 u(t_i), \quad \hat{x}_0 \in \mathbb{R}^n \quad (2.27)$$

$$\hat{x}_0(t_i^+) = \hat{x}_0(t_i^-) + K_0 \nu(t_i), \quad \nu(t_i) = y(t_i) - H_0 \hat{x}_0(t_i^-) \quad (2.28)$$

$$u(t_i) = f_R[\hat{x}_0(t_i^+)] = -G_0^* \hat{x}_0(t_i^+), \quad u \in \mathbb{R}^q, \quad (2.29)$$

причем  $\{w(\cdot)\}$  и  $\{v(\cdot)\}$  считаются независимыми последовательностями независимых одинаково распределенных случайных величин с нулевым средним значением и ковариациями  $Q_\theta$  и  $R_\theta$  соответственно. Разностное уравнение состояния (2.25) генерирует состояния во времени, начиная с некоторого начального значения  $x(t_{-s})$ , причем  $\mathbf{E} \{\|x(t_{-s})\|^2\} < \infty$ . Начальное состояние



помещено в момент времени  $t_{-s} \in \mathbb{R}$ , где  $s > 0$  определяет то, что называют *временем стабилизации*  $T_s = t_0 - t_{-s}$ , которое необходимо для того, чтобы все процессы в (2.25)–(2.29) считались стационарными в широком смысле при  $i \geq 0$ . Как обычно, уравнение (2.25) описывает объект, уравнение (2.26) – сенсор, и уравнения (2.27)–(2.29) – обратную связь. Обратная связь представлена уравнениями (2.27)–(2.28), типа уравнений фильтра Калмана, и регулятором (2.29). Регулятор задан некоторой функцией  $f_R[\cdot]$  от оценки  $\hat{x}_0(t_i^+)$  или выбран в соответствии со вторым равенством (2.29) с некоторой матрицей  $G_0^*$ .

Предполагается, что матрицы  $\Phi_0$ ,  $\Psi_0$ ,  $Q_0$ ,  $H_0$  и  $R_0$  данной системы (2.25)–(2.26) известны для *номинального режима*, то есть для *номинального значения*  $\theta_0$  параметра неопределенности  $\theta$ . Чтобы гарантировать существование устойчивого состояния фильтра с коэффициентом Калмана  $K_0$ , приняты предположения, что пара матриц  $(\Phi_0, Q_0^{1/2})$  – *стабилизируемая*, пара  $(\Phi_0, H_0)$  – *наблюдаемая* и пара  $(\Phi_0, \Psi_0)$  – *управляемая*. Матрица  $G_0^*$  может быть взята либо как заданная, либо как выбранная из условия LQG оптимальности номинального режима работы [5, 6, 21].

Параметр  $\theta$  изменяется (вследствие нарушения) в неизвестный момент времени  $t_c \in (t_0, t_i)$ , то есть происходит переключение  $\theta$  от известного значения  $\theta_0$  на некоторое другое неизвестное значение  $\theta_1$ ; для обнаружения момента нарушения необходим генератор решений (ГР) (рис. 2.5). Формализуя задачу синтеза ГР, рассмотрим

$$\left. \begin{aligned} x_j(t_{i+1}) &= \Phi_j x_j(t_i) + \Psi_j u(t_i) + w_j(t_i), \\ y_j(t_i) &= H_j x_j(t_i) + v_j(t_i), \end{aligned} \right\} \begin{array}{l} \mathcal{S}_0 \quad (j=0) \\ \mathcal{S}_1 \quad (j=1) \end{array} \quad \text{или} \quad (2.30)$$

как обобщенное описание двух систем: системы  $\mathcal{S}_0$  с  $\theta = \theta_0$  и системы  $\mathcal{S}_1$  с  $\theta = \theta_1$ . Фильтр (2.27)–(2.28), обозначенный на рис. 2.5 символом  $\mathcal{F}_0$ , спроектируем как фильтр Калмана для  $\mathcal{S}_0$

$$\left. \begin{aligned} K_0 &= \widetilde{P}_0 H_0^T C_0^{-1}, & C_0 &= H_0 \widetilde{P}_0 H_0^T + R_0, \\ \widehat{P}_0 &= \widetilde{P}_0 - \widetilde{P}_0 H_0^T C_0^{-1} H_0 \widetilde{P}_0, & \widetilde{P}_0 &= \Phi_0 \widehat{P}_0 \Phi_0^T + Q_0. \end{aligned} \right\} \mathcal{F}_0 \text{ для } \mathcal{S}_0. \quad (2.31)$$

Данные, подлежащие обработке, поступают на вход этого фильтра в виде

$$y(t_i) = \begin{cases} y_0(t_i), & \text{если } t_i < t_c \text{ (данные от } \mathcal{S}_0), \\ y_1(t_i), & \text{если } t_i \geq t_c \text{ (данные от } \mathcal{S}_1). \end{cases} \quad (2.32)$$

Задача состоит в обнаружении (за приемлемый промежуток времени) момента нарушения  $t_c$  при помощи подходящего решающего правила  $d_0(t_k) \in \{0, 1\}$  в соответствии с рис. 2.5.

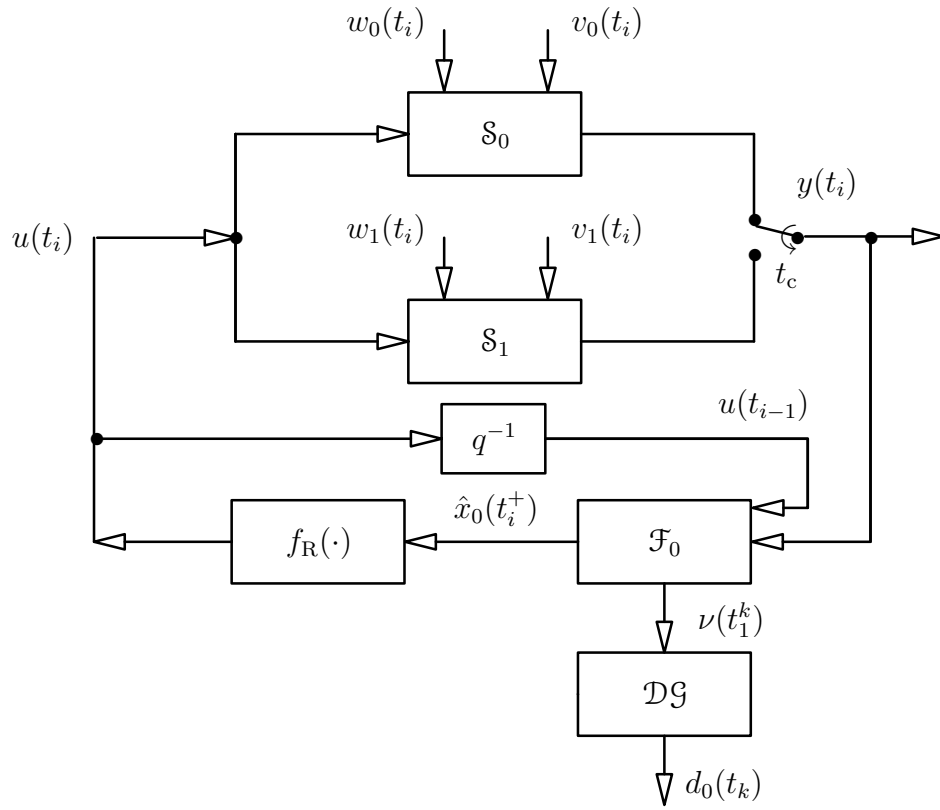


Рис. 2.5. Общая структура задачи и решения. Обозначения:  $\mathcal{S}_0$  – система с  $\theta = \theta_0$ , тогда как  $\mathcal{S}_1$  – система с  $\theta = \theta_1$ ;  $\mathcal{F}_0$  – фильтр Калмана системы  $\mathcal{S}_0$ ;  $\mathcal{DG}$  – генератор решений;  $q^{-1}$  обозначает запоминание на один шаг времени (единичная задержка);  $\nu(t_1^k) \triangleq \text{col}[\nu(t_1), \nu(t_2), \dots, \nu(t_k)]$

### 2.2.3. Метод обнаружения на основе ВКН

Многие системы, не только (2.25)–(2.29), имеют ОП, следовательно, можно рассматривать ВКН последовательности  $s_i$  в (2.24). Из-за того, что после нарушения ковариационные свойства ВКН могут стать — в зависимости от произошедшего изменения — очень сложными, для простоты будем считать, что функция ковариационного ядра  $\mathcal{K}_{ss}(j)$  последовательности  $s_i$  после момента изменения близка к экспоненциальной функции:

$$\mathcal{K}_{ss}(j) = D_s r^{|j|}, \quad j \in \mathbb{Z}, \quad r = \exp\{-\tau_s/\tau_c\}, \quad (2.33)$$

$$D_s \triangleq \mathbf{E} \{(s_i - \mathbf{E} \{s_i\})^2\},$$

где  $\tau_s$  – интервал времени при выборке и  $\tau_c$  – интервал корреляции последовательности  $s_i$ . Принимая это предположение, сформулируем метод ВКН для обнаружения нарушений следующим образом.

**Теорема 2.1** Пусть у динамической системы с ОП  $\{\nu(t_i)\}$ , где  $\nu(t_i) \in \mathbb{R}^m$ , есть два возможных режима работы: «нормальное функционирование» (гипотеза  $\mathbf{H}_0$ ) и «отказ» (гипотеза  $\mathbf{H}_1$ ). Для этих гипотез  $\{\nu(t_i)\}$  – гауссовская последовательность с  $\mathbf{E}\{\nu(t_i)\} = 0$ . Если выполнена гипотеза  $\mathbf{H}_0$ , то  $\{\nu(t_i)\}$  – последовательность независимых случайных величин с известной ковариацией  $C_0 = L_0 L_0^T$  каждого элемента  $\nu(t_i)$ , где  $L_0$  – квадратный корень матрицы  $C_0$ , найденный, например, при помощи разложения Холесского. Если выполнена гипотеза  $\mathbf{H}_1$ , то  $\{\nu(t_i)\}$  – последовательность коррелированных случайных величин с неизвестной ковариацией  $C_1 \neq C_0$  каждого элемента  $\nu(t_i)$ . Определим

$$\mu_i \triangleq L_0^{-1} \nu(t_i), \quad s_i \triangleq \sqrt{\frac{m}{2}} \left( \frac{1}{m} \mu_i^T \mu_i - 1 \right), \quad S_k \triangleq \frac{1}{\sqrt{k}} \sum_{i=1}^k s_i \quad (2.34)$$

и предположим, что при выполнении гипотезы  $\mathbf{H}_1$  ковариационное ядро последовательности  $s_i$  описано (возможно, приближенно) выражением (2.33).

Тогда асимптотически, при  $k$  стремящемся к бесконечности, справедливы следующие утверждения:

1. Законы распределения вероятностей  $\mathcal{L}(\cdot)$  величины  $S_k$  при этих гипотезах удовлетворяют свойствам<sup>5</sup>

$$\mathbf{H}_0 : \quad \mathcal{L}(S_k) \rightsquigarrow \mathcal{N}(0, 1), \quad (2.35)$$

$$\mathbf{H}_1 : \quad \mathcal{L}(S_k) \rightsquigarrow \mathcal{N}(m_{S_k}, D_{S_k}), \quad (2.36)$$

где  $\mathcal{N}(0; 1)$  – гауссовское (нормальное) распределение с нулевым средним значением и единичной дисперсией,  $\mathcal{N}(m_{S_k}, D_{S_k})$  – нормальное распределение, причем среднее значение  $m_{S_k}$  и дисперсия  $D_{S_k} = \sigma_{S_k}^2$  задаются формулами

$$\left. \begin{aligned} m_{S_k} &\triangleq \mathbf{E}\{S_k\} = m_s \sqrt{k}, \\ m_s &\triangleq \mathbf{E}\{s_i\} = (1/\sqrt{2m}) \operatorname{tr}\{\Delta\}, \\ D_{S_k} &\triangleq \mathbf{E}\{(S_k - m_{S_k})^2\} = D_s \frac{1+r}{1-r}, \\ D_s &= 1 + \frac{2}{m} \operatorname{tr}\{\Delta\} + \frac{1}{m} \|\Delta\|^2, \\ \Delta &= L_0^{-1}(C_1 - C_0)L_0^{-T}, \quad \|\Delta\|^2 \triangleq \sum_{i,j=1}^m |\Delta_{ij}|^2. \end{aligned} \right\} \quad (2.37)$$

<sup>5</sup> Обозначение  $\rightsquigarrow$  соответствует слабой сходимости [22].

2. Если  $S_k$  (2.34) используется в качестве решающей функции в некоторый момент времени  $k$  в решающем правиле

$$\begin{array}{c} \mathbf{H}_1 \\ |S_k| \geq h, \\ \mathbf{H}_0 \end{array} \quad (2.38)$$

где  $h$  – некоторый удобно выбранный порог (например,  $h = 3$ ), тогда вероятность ложной тревоги,  $P_F$ , и вероятность обнаружения,  $P_D$ , удовлетворяют следующим приближенным выражениям (они становятся точными, если законы  $\mathcal{L}(S_k)$  в (2.35), (2.36) приняты за нормальные):

$$\begin{aligned} P_F &\simeq 1 - \phi(h), \quad P_D \simeq 1 - \frac{1}{2} \left[ \phi\left(\frac{m_{S_k} + h}{\sigma_{S_k}}\right) - \right. \\ &\quad \left. - \phi\left(\frac{m_{S_k} - h}{\sigma_{S_k}}\right) \right] > \frac{1}{2} \left[ 1 + \phi\left(\frac{m_{S_k} - h}{\sigma_{S_k}}\right) \right], \end{aligned} \quad (2.39)$$

где использован интеграл вероятности

$$\phi(x) \triangleq \frac{2}{\sqrt{2\pi}} \int_0^x \exp(-t^2/2) dt.$$

3. Если порог  $h$  в (2.38) выбран как  $h = \phi^{-1}(1 - \alpha)$ , где  $\alpha$  – заданный уровень  $P_F$  в (2.39),  $P_F = \alpha$ , тогда гипотеза  $\mathbf{H}_1$  обнаруживается с вероятностью  $P_D = 1 - \beta$  (где  $0 < \beta < 1/2$ ) не позже, чем через

$$k_a^* \simeq [\phi^{-1}(1 - \alpha) + \sigma_{S_k} \phi^{-1}(1 - 2\beta)]^2 / m_s^2 \quad (2.40)$$

дискретных моментов времени, где  $x = \phi^{-1}(y)$  – решение уравнения  $\phi(x) = y$ .

Доказательство Теоремы 2.1 дано в Приложении к данному подразделу.

Очевидно, что последовательность  $S_k$  применима к любой системе, имеющей такую невязку  $\nu(t_i)$ , для обнаружения изменения, характеризуемого матрицей  $\Delta$  с  $\text{tr}\{\Delta\} \neq 0$ , при помощи правила (2.38). При этом важно иметь в виду следующие свойства данного правила:

1. Правило (2.38) содержит в себе те же квадратичные формы, что и логарифмическая функция правдоподобия

$$\ln p(\nu(t_1^k) \mid \mathbf{H}_0) = -\frac{km}{2} \ln(2\pi) - \frac{k}{2} \ln|C_0| - \frac{1}{2} \sum_{i=1}^k \nu^T(t_i) C_0^{-1} \nu(t_i),$$

где  $\nu(t_1^k) \triangleq \text{col}[\nu(t_1), \nu(t_2), \dots, \nu(t_k)]$ , так как

$$S_k = \sqrt{\frac{m}{2k}} \sum_{i=1}^k \left( \frac{1}{m} \nu^T(t_i) C_0^{-1} \nu(t_i) - 1 \right). \quad (2.41)$$

2. Достаточная статистика  $\ell(\cdot) \triangleq \ell(\nu(t_1^k))$  для обнаружения нарушений в параметрах системы при выполнении предположений Теоремы 2.1 задается следующими формулами [23, подразд. 2.6.2]

$$\begin{aligned} \ell(\cdot) &= \sum_{i=1}^k \nu^T(t_i) C_0^{-1} \nu(t_i) - \nu^T(t_1^k) \mathbf{C}_1^{-1}(k) \nu(t_1^k), \\ \mathbf{C}_1(k) &\triangleq \mathbf{E} \{ \nu(t_1^k) \nu^T(t_1^k) \mid \mathbf{H}_1 \}. \end{aligned}$$

Очевидный теоретический недостаток данного метода состоит в том, что решающая функция  $S_k$ , заданная формулой (2.34) или эквивалентной формулой (2.41), не является достаточной статистикой; однако этот метод нашел практическое применение во многих приложениях со времени его изобретения, потому что он работает без какой-либо информации о модели после нарушения, например, при проверке оптимальности фильтра [24].

3. Тип параметрических изменений, обнаруживаемых по правилу (2.38), ограничен теми, для которых  $\text{tr}\{\Delta\} \neq 0$ .
4. Строго удовлетворить условию  $\text{tr}\{\Delta\} = 0$  в большинстве реальных ситуаций маловероятно, и поэтому ограничение  $\text{tr}\{\Delta\} \neq 0$  не должно рассматриваться как критическое с практической точки зрения.

**Пример 1** Пусть  $m = 3$  и

$$C_0 = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 8 & 2 \\ 3 & 2 & 14 \end{bmatrix}, \quad C_1 = \begin{bmatrix} 2.0 & 1.8 & 2.4 \\ 1.8 & 9.0 & 1.0 \\ 2.4 & 1.0 & 13.0 \end{bmatrix}.$$

Тогда

$$L_0 = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 2 & 0 \\ 3 & -2 & 1 \end{bmatrix}, \quad \Delta = \begin{bmatrix} 1.0 & -1.1 & -5.8 \\ -1.1 & 1.45 & 6.3 \\ -5.8 & 6.3 & 31.0 \end{bmatrix}$$

и  $m_s \approx 13.7$ ,  $D_s \approx 395$ . Пусть  $\alpha = 0.005$  и  $\beta = 0.005$ . Мы получаем  $h = \phi^{-1}(1 - \alpha) = 2.8$  и  $\phi^{-1}(1 - 2\beta) = 2.6$ .

Если  $r = 0.5$ , тогда  $k_a^* \simeq 46$  и, если  $r = 0.95$ , тогда  $k_a^* \simeq 562$ . Следовательно, взяв за основу ожидаемые (например, наиболее вероятные или наиболее опасные) нарушения и исходя из желаемых значений вероятностей ошибок первого и второго рода  $\alpha$  и  $\beta$  решающего правила (2.38), мы имеем возможность численно прогнозировать величину промежутка времени<sup>6</sup>  $k_a^*$ , после которого нарушение будет обнаружено с вероятностью не меньше, чем  $P_D = 1 - \beta$ .

Для приведенного примера условие необнаружения нарушений задается уравнением

$$\text{tr}\{\Delta\} = 27a_{11} - 11a_{12} - 10a_{13} + \frac{5}{4}a_{22} + 2a_{23} + a_{33} = 0, \quad (2.42)$$

где  $a_{ij} = a_{ji}$  для  $A = C_1 - C_0$ . С учетом положительной определенности матриц ковариаций можно выбрать  $C_1$  в соответствии с условием (2.42), например,

$$C_1 = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 8 & 1 \\ 3 & 1 & 16 \end{bmatrix}, \quad A \triangleq C_1 - C_0 = [a_{ij}] = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & -1 & 2 \end{bmatrix}.$$

Этот простой пример делает очевидным тот факт, что определенные нарушения, специально подобранные так, чтобы удовлетворять уравнению  $\text{tr}\{\Delta\} = 0$ , не могут быть обнаружены этим методом.

Теорема 2.1 предполагает, что при изменении параметра  $\theta$  математическое ожидание обновляющей последовательности  $\{\nu(t_i)\}$  не меняется и остается равным нулю:

$$\mathbf{H}_0 : \nu(t_i) \sim \mathcal{N}(0, C_0) \quad \text{и} \quad \mathbf{H}_1 : \nu(t_i) \sim \mathcal{N}(0, C_1). \quad (2.43)$$

Если это условие не выполняется, то (2.43) следует переписать следующим образом:

$$\mathbf{H}_0 : \nu(t_i) \sim \mathcal{N}(0, C_0) \quad \text{и} \quad \mathbf{H}_1 : \nu(t_i) \sim \mathcal{N}(m_1, C_1) \quad (2.44)$$

с некоторым  $m_1 \neq 0$ . Это влечет за собой лишь небольшую замену выражений в Теореме 2.1, а именно,  $\Delta$  в формуле (2.37) изменяется так:

$$\left. \begin{aligned} \Delta &= L_0^{-1}(C_1 + m_1 m_1^T - C_0)L_0^{-T} \\ \text{вместо} \\ \Delta &= L_0^{-1}(C_1 - C_0)L_0^{-T}. \end{aligned} \right\}. \quad (2.45)$$

---

<sup>6</sup> В этом примере задержка  $k_a^*$  должна быть увеличена на  $k_{\text{norm}} \gtrsim 30$ , с технической точки зрения это необходимо для того, чтобы распределения в (2.35)–(2.36) можно было считать почти нормальными.

Следовательно, для обнаружения нарушений, метод ВКН сводится к прямому накоплению  $S_k$  (2.34) в форме последовательного алгоритма

$$S_k = S_{k-1} \sqrt{1 - 1/k} + s_k / \sqrt{k}, \quad k \in \mathbb{N}, \quad S_0 = 0 \quad (2.46)$$

в соответствии с правилом (2.38). При проверке гипотезы  $\mathbf{H}_1$  относительно гипотезы  $\mathbf{H}_0$  обозначим  $S_k$  в (2.46) символом  $S_k^{(0)}$ , подчеркивая надстрочным индексом  $^{(0)}$ , что основная гипотеза есть  $\mathbf{H}_0$ . Тогда получаем следующие две схемы с правилом останковки  $t_a$  (время обнаружения нарушения) для обнаружения альтернативной гипотезы  $\mathbf{H}_1$ .

(А) Решение принимается в некоторый текущий момент времени  $t_k$ :

$$d_0(t_k) = \begin{cases} 0 & \text{if } |S_k^{(0)}| < h; \quad \text{выбрана } \mathbf{H}_0 \\ 1 & \text{if } |S_k^{(0)}| \geq h; \quad \text{выбрана } \mathbf{H}_1 \end{cases},$$

$$t_a = \min \{t_k : d_0(t_k) = 1\}.$$

(В) Решение принимается в конце интервала выборки номер  $l = 1, \dots, L$ , каждый размера  $N$ :

$$d_0(l) = \begin{cases} 0 & \text{if } \forall k = 1, 2, \dots, N : |S_{N(l-1)+k}^{(0)}| < h; \quad \text{выбрана } \mathbf{H}_0 \\ 1 & \text{if } \exists k = 1, 2, \dots, N : |S_{N(l-1)+k}^{(0)}| \geq h; \quad \text{выбрана } \mathbf{H}_1 \end{cases},$$

$$t_a = \tau_s \left[ N(l-1) + \min \left\{ k : |S_{N(l-1)+k}^{(0)}| \geq h \right\} \right] \stackrel{r}{=} \tau_s N \min \{l : d_0(l) = 1\},$$

где  $\stackrel{r}{=}$  обозначает равенство с округлением.

Этот результат имеет важное значение в силу того, что многие стохастические динамические системы имеют или могут быть снабжены «выбеливающим» фильтром, который производит обработку невязок. Это условие выполнено для систем (2.25)–(2.29), в которых «выбеливающий» фильтр существует в виде фильтра Калмана (2.27)–(2.28) с ОП  $\{\nu(t_i)\}$ .

#### 2.2.4. Проверка и выбор фильтра обратной связи на основе ВКН

Из теории обновляющих процессов известны два факта [25]:

- (i) Случай (2.43) выполняется, если нарушения происходят только в ковариационных матрицах  $Q_\theta$  и  $R_\theta$ .
- (ii) Только нарушения в матрицах  $\Phi_\theta$ ,  $\Psi_\theta$  и/или  $H_\theta$  являются причиной для случая (2.44).

Если имеется случай (2.44), тогда должна быть использована формула (2.45) с  $m_1$  из процедуры, описанной, например, в [25].

**Пример 2** Рассмотрим простой случай, когда  $n = m = 1$ ,  $u(t_i) \equiv 0$ ,  $Q_1 = Q_0$ ,  $R_1 = R_0$ ,  $H_1 = H_0 = 1$  и  $\Phi_1 \neq \Phi_0$  ( $|\Phi_j| < 1$ ,  $j = 0, 1$ ). При  $t_i \gg t_c$  (т. е. при действии гипотезы  $\mathbf{H}_1$ , когда эффекты переходных процессов после нарушения уже исчезают) из (2.30) ( $j = 0$ ), (2.31), (2.32) и (2.27)–(2.28) имеем

$$\left. \begin{aligned} \nu(t_i) &= x_1(t_i) + v_0(t_i) - \hat{x}_0(t_i) = e(t_i) + v_0(t_i) , \\ e(t_i) &\triangleq x_1(t_i) - \hat{x}_0(t_i) . \end{aligned} \right\} \quad (2.47)$$

Обозначив

$$\left. \begin{aligned} m_1(t_i) &\triangleq \mathbf{E} \{ \nu(t_i) \mid \mathbf{H}_1 \} , & m_e(t_i^-) &\triangleq \mathbf{E} \{ e(t_i) \} , \\ m_{x_1}(t_i) &\triangleq \mathbf{E} \{ x_1(t_i) \} , & m_{\hat{x}_0}(t_i^\pm) &\triangleq \mathbf{E} \{ \hat{x}_0(t_i^\pm) \} , \end{aligned} \right\} \quad (2.48)$$

получим из (2.47), (2.30) (при  $j = 1$ ) и (2.27)

$$\left. \begin{aligned} m_1(t_i) &= m_e(t_i^-) , & m_e(t_i^-) &= m_{x_1}(t_i) - m_{\hat{x}_0}(t_i^-) , \\ m_{x_1}(t_i) &= \Phi_1 m_{x_1}(t_{i-1}) , & m_{\hat{x}_0}(t_i^-) &= \Phi_0 m_{\hat{x}_0}(t_{i-1}^+) . \end{aligned} \right\} \quad (2.49)$$

Взяв математическое ожидание от (2.28), найдем

$$m_{\hat{x}_0}(t_i^+) = (1 - K_0)m_{\hat{x}_0}(t_i^-) + K_0 m_{x_1}(t_i) .$$

Подстановка предыдущей формулы в последнее выражение (2.49) дает

$$m_{\hat{x}_0}(t_i^-) = \Phi_0 [(1 - K_0)m_{\hat{x}_0}(t_{i-1}^-) + K_0 m_{x_1}(t_{i-1})] .$$

Используя (2.48) и (2.49), последовательно находим

$$\begin{aligned} m_e(t_i^-) &= m_{x_1}(t_i) - \Phi_0 [(1 - K_0)m_{x_1}(t_{i-1}) - (1 - K_0)m_e(t_{i-1}^-) + \\ &\quad + K_0 m_{x_1}(t_{i-1})] = \\ &= m_{x_1}(t_i) - \Phi_0 [m_{x_1}(t_{i-1}) - (1 - K_0)m_e(t_{i-1}^-)] = \\ &= (\Phi_1 - \Phi_0)m_{x_1}(t_{i-1}) + \Phi_0(1 - K_0)m_e(t_{i-1}^-) , \\ m_1(t_i) &= \Phi_0(1 - K_0)m_1(t_{i-1}) + (\Phi_1 - \Phi_0)m_{x_1}(t_{i-1}) . \end{aligned}$$

Так как  $0 < \Phi_0 < 1$  и  $0 < \Phi_0(1 - K_0) < 1$ , то существует устойчивое состояние для следующих величин:

$$m_1 \triangleq \lim_{t_i \rightarrow \infty} \{m_1(t_i)\} = \lim_{t_i \rightarrow \infty} \{m_1(t_{i-1})\} , \quad m_{x_1} \triangleq \lim_{t_i \rightarrow \infty} \{m_{x_1}(t_{i-1})\} .$$



На этом основании окончательно получаем установившееся значение

$$m_1 = \frac{\Phi_1 - \Phi_0}{1 - \Phi_0(1 - K_0)} m_{x_1}.$$

Благодаря тому, что  $0 < \Phi_0 < 1$ , находим, что  $m_{x_1} = 0$ . Следует также заметить, что, хотя условие  $\text{tr}\{\Delta\} = 0$  не является необходимым и достаточным для оптимальности фильтра, количество нарушений, удовлетворяющих условию  $\text{tr}\{\Delta\} = 0$  (строгое равенство нулю), сравнительно невелико.

Как и для случая двух конкурирующих гипотез  $\mathbf{H}_0$  и  $\mathbf{H}_1$  в пункте 2.2.3, можно применить одну из двух схем, (А) или (В), для принятия решения. В отличие от этого случая каждая выбранная гипотеза  $\mathbf{H}_j$  ( $j = 0, 1, \dots, K$ ) проверяется относительно другой гипотезы  $\mathbf{H}_i$  ( $i = 0, 1, \dots, K; i \neq j$ ) с использованием своей собственной решающей функции  $S_k^{(j)}$ , которая равна  $S_k$ , вычисляется в соответствии с (2.46) и снабжена надстрочным индексом  $^{(j)}$ . Снова получаем две схемы:

(А) Решение принимается в текущий момент времени  $t_k$ . Для любой  $\mathbf{H}_j$ , где  $j \in \{0, 1, \dots, K\}$ , решающее правило выглядит следующим образом:

$$d_j(t_k) = \begin{cases} 0 & \text{if } |S_k^{(j)}| < h; \quad \mathbf{H}_j \text{ принимается,} \\ 1 & \text{if } |S_k^{(j)}| \geq h; \quad \mathbf{H}_j \text{ отвергается.} \end{cases}$$

Предельное время обнаружения нарушения (правило остановки) равно

$$t_a = \min \left\{ t_k : \left( \exists \kappa : d_\kappa(t_k) = 0 \ \& \ \forall_{j \neq \kappa} d_j(t_k) = 1 \right) \right\},$$

а предельное правило выбора наилучшей гипотезы определяется выражением

$$\kappa = \{j \in \{0, 1, \dots, K\} : d_j(t_a) = 0\} ; \quad \text{выбирается } \mathbf{H}_\kappa.$$

(В) Решение принимается в конце интервала выборки номер  $l = 1, \dots, L$ , каждый размера  $N$ . Для любой  $\mathbf{H}_j$ , где  $j \in \{0, 1, \dots, K\}$ , решающее правило выглядит следующим образом:

$$d_j(l) = \begin{cases} 0 & \text{if } \left\{ \begin{array}{l} \forall k = 1, \dots, N : |S_{N(l-1)+k}^{(j)}| < h, \\ \text{then } \mathbf{H}_j \text{ принимается,} \end{array} \right. \\ 1 & \text{if } \left\{ \begin{array}{l} \exists k = 1, \dots, N : |S_{N(l-1)+k}^{(j)}| \geq h, \\ \text{then } \mathbf{H}_j \text{ отвергается,} \end{array} \right. \end{cases} \quad (2.50)$$

$$t_a \stackrel{r}{=} \tau_s N \min \left\{ l : \left( \exists \kappa : d_\kappa(l) = 0 \ \& \ \forall_{j \neq \kappa} d_j(l) = 1 \right) \right\}, \quad (2.51)$$

а предельное правило выбора гипотезы выглядит так:

$$\kappa = \{j \in \{0, 1, \dots, K\} : d_j(l) = 0\} ; \quad \text{выбирается } \mathbf{H}_\kappa . \quad (2.52)$$

Этот метод представлен структурой на рис. 2.6.

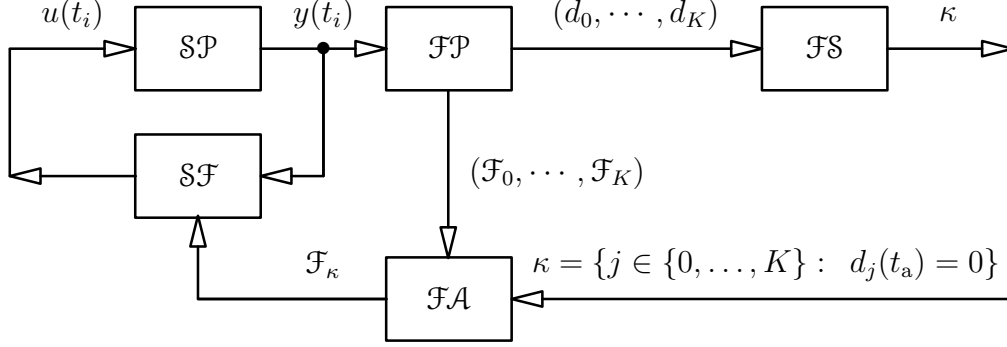


Рис. 2.6. Выбор наилучшего фильтра. Обозначения:  $\mathcal{SP}$  – совокупность систем;  $\mathcal{FP}$  – совокупность фильтров,  $\mathcal{FS}$  – выбор фильтра,  $\mathcal{FA}$  – эффектор обратной связи, и  $\mathcal{SF}$  – обратная связь системы. Правило остановки задается формулой  $t_a = \min \left\{ t_k : (\exists \kappa : d_\kappa(t_k) = 0 \ \& \ \forall_{j \neq \kappa} d_j(t_k) = 1) \right\}$ . В это время  $\mathcal{FA}$  загружает выбранный  $\mathcal{F}_\kappa$  в  $\mathcal{SF}$ , если его там еще нет

### 2.2.5. Некоторые результаты вычислительных экспериментов

Решающее значение для оптимальной обработки навигационных данных и безопасности полета имеет уверенность в безошибочности работы бортового оборудования и в отсутствии опасных изменений в движении объекта.

В качестве практической системы для приложения и испытания полученных выше результатов возьмем демпфированный контур Шулера, возбуждаемый экспоненциально коррелированным шумом (описание этого контура было впервые дано в [26]). В соответствии с [26] рассмотрим уравнения

$$\left. \begin{aligned} x_j(t_{i+1}) &= \Phi_j x_j(t_i) + \Gamma_j w_j(t_i) , \\ y_j(t_i) &= H_j x_j(t_i) + v_j(t_i) , \end{aligned} \right\} \begin{aligned} &\mathcal{S}_0 \text{ (для } j = 0) \text{ или} \\ &\mathcal{S}_j \text{ (для } j = 1, \dots, K) , \end{aligned} \quad (2.53)$$

которые взяты из (2.30), причем управляющий вход  $u(t_i)$  принят равным нулю. Здесь имеется  $(K + 1)$  режимов работы, каждый режим ассоциирован с некоторой системой  $\mathcal{S}_j$ . В данном случае  $\mathcal{S}_0$  – система без нарушений, а каждая  $\mathcal{S}_j$  (при  $j = 1, \dots, K$ ) является системой с нарушением. В системе

$\mathcal{S}_0$  имеем переходную матрицу  $\Phi_0$  и входную матрицу  $\Gamma_0$ , равные (соответственно)

$$\begin{bmatrix} 0.75 & -1.74 & -0.3 & 0 & -0.15 \\ 0.09 & 0.91 & -0.0005 & 0 & -0.008 \\ 0 & 0 & 0.95 & 0 & 0 \\ 0 & 0 & 0 & 0.55 & 0 \\ 0 & 0 & 0 & 0 & 0.905 \end{bmatrix}, \quad \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 24.64 & 0 & 0 \\ 0 & 0.835 & 0 \\ 0 & 0 & 1.83 \end{bmatrix}, \quad (2.54)$$

а также имеем матрицы

$$Q_0 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad H_0 = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix}, \quad R_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \quad (2.55)$$

Отвлекаясь на этом этапе от физической интерпретации данных характеристик (2.54)–(2.55), в этом пункте рассмотрим только такое множество «сбойных» систем, которое подчиняется следующей формальной зависимости:

$$\left. \begin{aligned} \Phi_j &= \Phi_0, \quad \Gamma_j = \Gamma_0, \quad Q_j = Q_0, \quad R_j = R_0, \\ H_j &= \begin{bmatrix} 1-e & 0 & 0 & 0 & 1-f \\ 0 & 1-g & 0 & 1-h & 0 \end{bmatrix}, \quad \{efgh\} \triangleq j_{[2]}, \end{aligned} \right\} \mathcal{S}_j, \quad (j = \overline{1, 15}).$$

где  $\{efgh\}$  – двоичный код  $j_{[2]}$  числа  $j$ . Например, если  $j = 2$  или  $j = 9$ , тогда  $\{efgh\} = \{0010\}$  или, соответственно,  $\{1001\}$ . Для этих случаев

$$H_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \quad H_9 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

Нарушение  $\mathcal{S}_0 \xrightarrow{t_c} \mathcal{S}_j$  ( $j = 1, \dots, 15$ ) возникает в момент времени  $t_c = 2300$ , но фильтр  $\mathcal{F}_0$  «не знает» об этом и продолжает удовлетворять уравнениям (2.31). Экспериментальные данные для всех  $K = 15$  случаев «отказа» системы показаны на рис. 2.7–2.8 вместе со случаем «нормального функционирования» системы, когда  $\{efgh\} = \{0000\}$ . Если значение порога  $h$  в (2.38) установить равным 3, тогда вероятность ложной тревоги,  $P_F$ , в случае  $\mathcal{S}_0$  будет не больше, чем 0.3% (из-за свойства (2.35)), в остальных случаях вероятность верного обнаружения,  $P_D$ , приближается к 100% из-за монотонного возрастания  $|\mathbf{E}\{S_k\}|$ . Величина запаздывания в обнаружении момента переключения систем зависит от типа нарушения и, как это можно видеть из рис. 2.7–2.8, находится на временном интервале между 100 и 1500.

Приведенные выше результаты и также те, что были получены для различных выборок последовательностей  $w(t_i)$  и  $v(t_i)$  в (2.25)–(2.26), показывают целесообразность данного подхода для наблюдения за инерциальными навигационными системами и, следовательно, позволяют ожидать, что

для него существуют и другие полезные приложения. Это предположение действительно нашло подкрепление в данной работе, благодаря похожим результатам, полученным из других экспериментов.

### 2.2.6. Разработка моделирующего программного обеспечения

Первые экспериментальные графики (см. рис. 2.7–2.8) сделаны в системе MATLAB 6. Они отображают результаты, полученные в программном комплексе ASPID (Adaptive System Parameter Identification), разработанном при помощи Visual C 6.0 (service pack 6) для исследования источников инструментальной погрешности инерциальных навигационных систем [29]. Однако для широкомасштабного исследования всех предложенных алгоритмов обнаружения/селекции, соответствующих рис. 2.6, потребовалась разработка специального программного обеспечения.

Такой продукт, названный Linear Stochastic Control System (LSCS), создан в данной работе. Он позволяет наблюдать поведение  $S_k$  до и после системного сбоя в полном соответствии с рис. 2.6. Приложение разработано с использованием системы Visual C++.NET и может работать на операционных системах Windows 98/2000/XP. Доступны следующие функции:

- ▷ построение графиков в реальном времени;
- ▷ поддержка многорежимности;
- ▷ ручное масштабирование;
- ▷ дружественная рабочая область с возможностью сохранения графиков в bmp-файл;
- ▷ ввод системных матриц  $\Phi_0$ ,  $\Gamma_0$  и  $H_0$ , входных значений, возможность их изменения в любой момент  $t_c$ ;
- ▷ полезные настройки для создания уникального стиля кривых и надписей на рабочей области;
- ▷ удобный интерфейс;
- ▷ и многое другое.

Приложение LSCS может работать в одном из трех доступных режимов:

<i>Режим по умолчанию</i>	★  Приложение вычисляет и рисует (на рабочей области в режиме реального времени) изменения $S_k^{(0)}$ , $S_k^{(0)}$ задана формулой (2.46), введенной в пункте 2.2.3; доступен только один фильтр $\mathcal{F}_0$ .
---------------------------	--

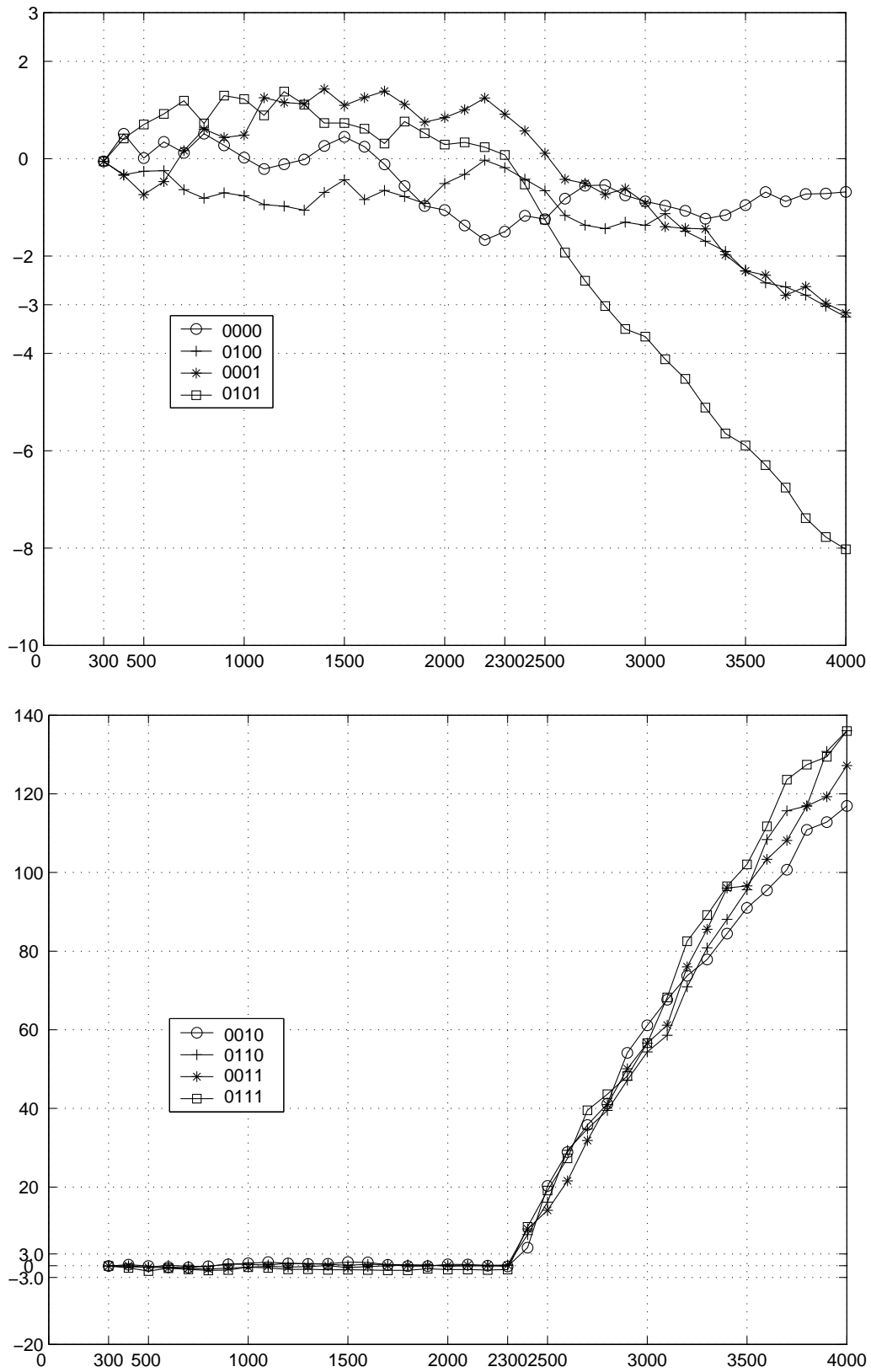


Рис. 2.7. Поведение  $S_k$  до и после нарушения в момент времени  $t_c = 2300$ ; *вверху*:  $j = 0, 4, 1, 5$ , *внизу*:  $j = 2, 6, 3, 7$

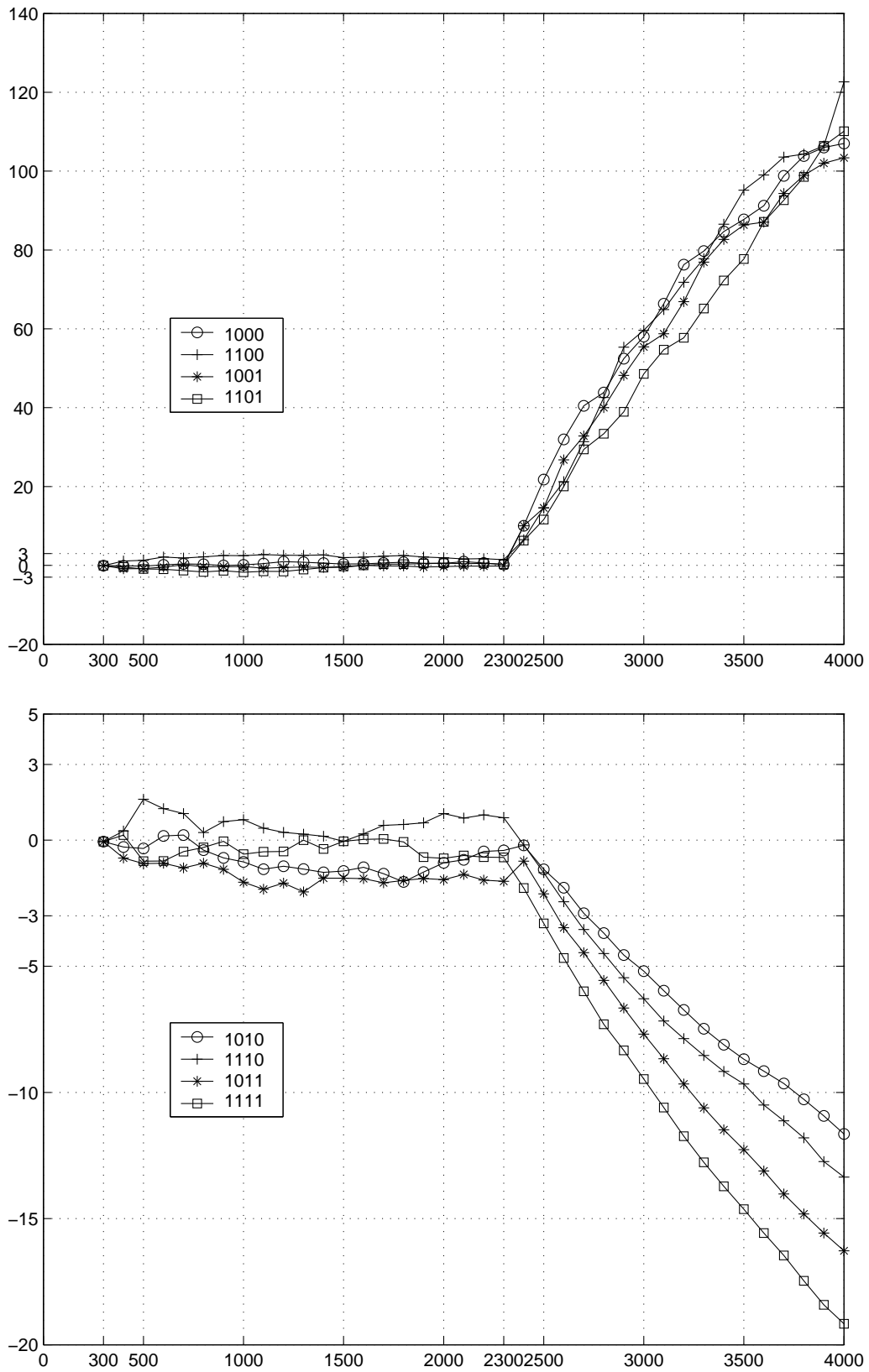


Рис. 2.8. Поведение  $S_k$  до и после нарушения в момент времени  $t_c = 2300$ ; *вверху:*  $j = 8, 12, 9, 13$ , *внизу:*  $j = 10, 14, 11, 15$

**Смешанный режим** |★| Этот режим поддерживает 4 различных фильтра для сбоя  $\mathcal{F}_j$ ,  $j \in \{1, 2, \dots, K\}$  в сравнении с одним (номинальным) фильтром  $\mathcal{F}_0$ ; 4 кривых изменений  $S_k^{(j)}$  могут быть взяты из  $\mathcal{F}_j$ , и можно наблюдать поведение их всех, как показано на рис. 2.7–2.8; пусть  $\mathcal{S}_0 \xrightarrow{2300} \mathcal{S}_j \mid j \in \{2, 6, 3, 7\}$  означает, что одно из четырех нарушений  $\mathcal{S}_0 \xrightarrow{t_c} \mathcal{S}_j$  возникает в момент времени  $t_c = 2300$  при  $j = 2, 6, 3$  или  $7$  – этот случай изображен на рис. 2.7, *внизу*.

**Режим обновления** |★| Приложение вычисляет и рисует (на рабочей области в режиме реального времени) изменения  $S_k^{(j)}$  для любого  $j \in \{0, \dots, K\}$ ; для этого случая введем следующее обозначение  $\mathcal{S}_0 \xrightarrow{397} \mathcal{S}_9 \mid \mathcal{FP} = \{\mathcal{F}_0, \mathcal{F}_2, \mathcal{F}_8, \mathcal{F}_9\}$  – данная запись означает, что нарушение  $\mathcal{S}_0 \xrightarrow{t_c} \mathcal{S}_j$  возникает в момент времени  $t_c = 397$  при  $j = 9$ ; в этом режиме Схема В (2.50)–(2.52) используется с каждой  $S_k^{(j)}$  и устанавливается на ноль через каждые  $N$  шагов формулы (2.46).

С помощью этого приложения в режиме обновления было сделано множество вычислительных экспериментов (наиболее интересные из них представлены на рис. 2.9–2.12). В будущем планируется продолжить эти исследования в большем объеме.

### 2.2.7. Заключение

Метод ВКН сформулирован для широкого класса линейных стохастических систем управления и проверен на практике в применении к системам оценки состояния. Полученные результаты могут иметь значительную практическую ценность, благодаря высокому качеству обнаружения метода при не слишком высоких вычислительных затратах. Предложенный метод испытан применительно к задаче обнаружения внезапных нарушений в модели контура Шулера инерциальной навигационной системы. Эта работа выполнена при частичной поддержке исследовательского гранта Министерства образования и науки РФ (грант № Т02-03.2-3427).

Дальнейшие исследования будут касаться качественных особенностей обнаружения момента нарушения и количественных аргументов «за и против» данного упрощенного метода в его сравнении с оптимальным последователь-

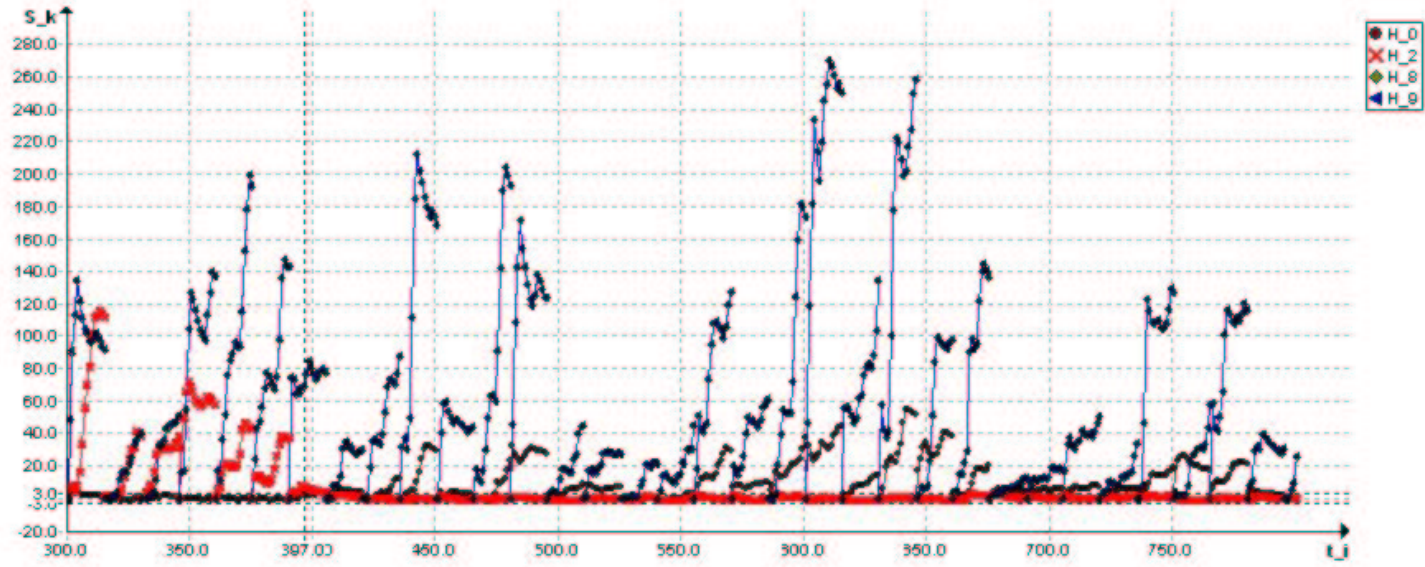


Рис. 2.9.  $S_0 \xrightarrow{397} S_2 \mid \mathcal{FP} = \{\mathcal{F}_0, \mathcal{F}_2, \mathcal{F}_8, \mathcal{F}_9\}$ . Объем выборки  $N = 15$  в Схеме В, (2.50)–(2.52). *Замечание:* В этом случае нарушение  $S_0 \xrightarrow{t_c} S_j$  возникает в момент времени  $t_c = 397$  при  $j = 2$ . Схема В (2.50)–(2.52) используется с каждой  $S_k^{(j)}$ ,  $j \in \{0, 2, 8, 9\}$ , и устанавливается на ноль через каждые  $N$  шагов алгоритма (2.46). Можно также получить отмасштабированную версию этого графика, – для сравнения см. рис. 2.10



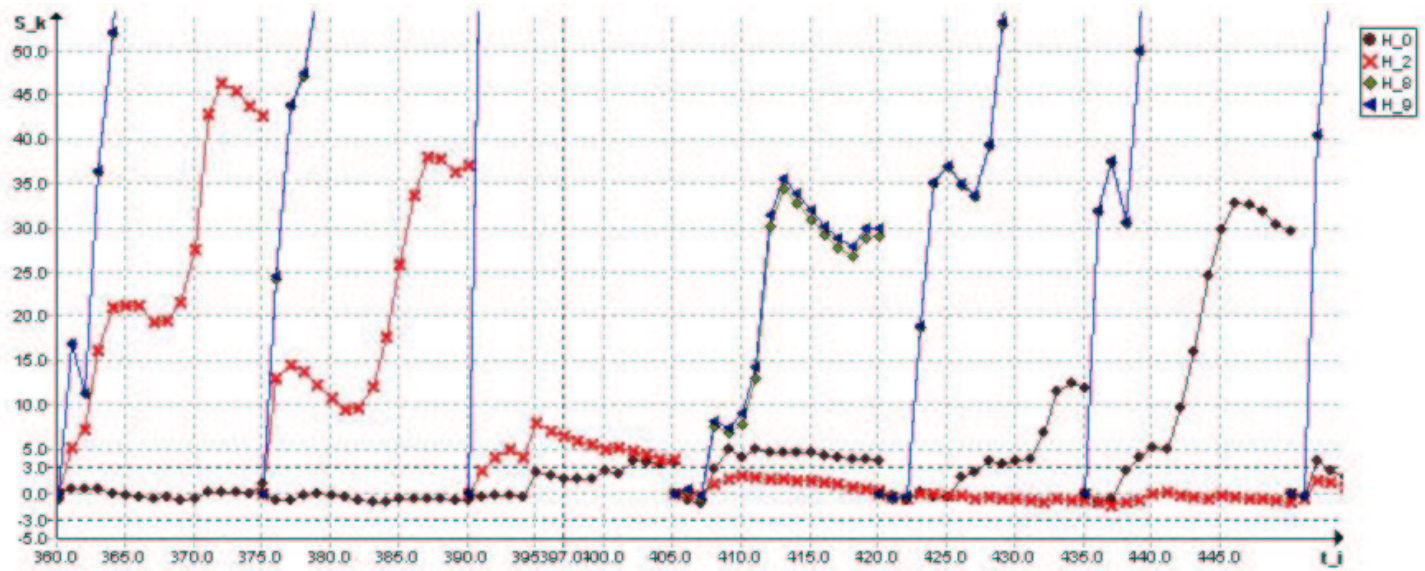


Рис. 2.10. Отмасштабированный график рис. 2.9 с  $\mathcal{S}_0 \xrightarrow{397} \mathcal{S}_2 \mid \mathcal{FP} = \{\mathcal{F}_0, \mathcal{F}_2, \mathcal{F}_8, \mathcal{F}_9\}$ . Объем выборки  $N = 15$  в Схеме В, (2.50)–(2.52). *Замечание:* Из этого рисунка видно, что до нарушения при  $t_i < 397$  только  $S_k^{(0)}$ , вычислявшаяся по формуле (2.46) в  $\mathcal{F}_0$ , остается в окрестности нуля. После переключения  $\mathcal{S}_0$  на  $\mathcal{S}_2$  в момент времени  $t_i = 397$ , поведение кривых резко изменяется:  $S_k^{(2)}$ , вычислявшаяся по формуле (2.46) в  $\mathcal{F}_2$ , входит в окрестность нуля, в то время как  $S_k^{(0)}$  покидает эту окрестность вместе с остальными кривыми, – в этом случае  $S_k^{(8)}$  и  $S_k^{(9)}$  взяты из  $\mathcal{F}_8$  и  $\mathcal{F}_9$ , соответственно. *Примечание:* При недостаточно уменьшенном масштабе одна кривая может визуальнo перейти в другую. На этом рисунке можно легко отличить кривую  $S_k^{(8)}$  от кривой  $S_k^{(9)}$  в моменты времени  $t_i = 413$  до  $t_i = 420$

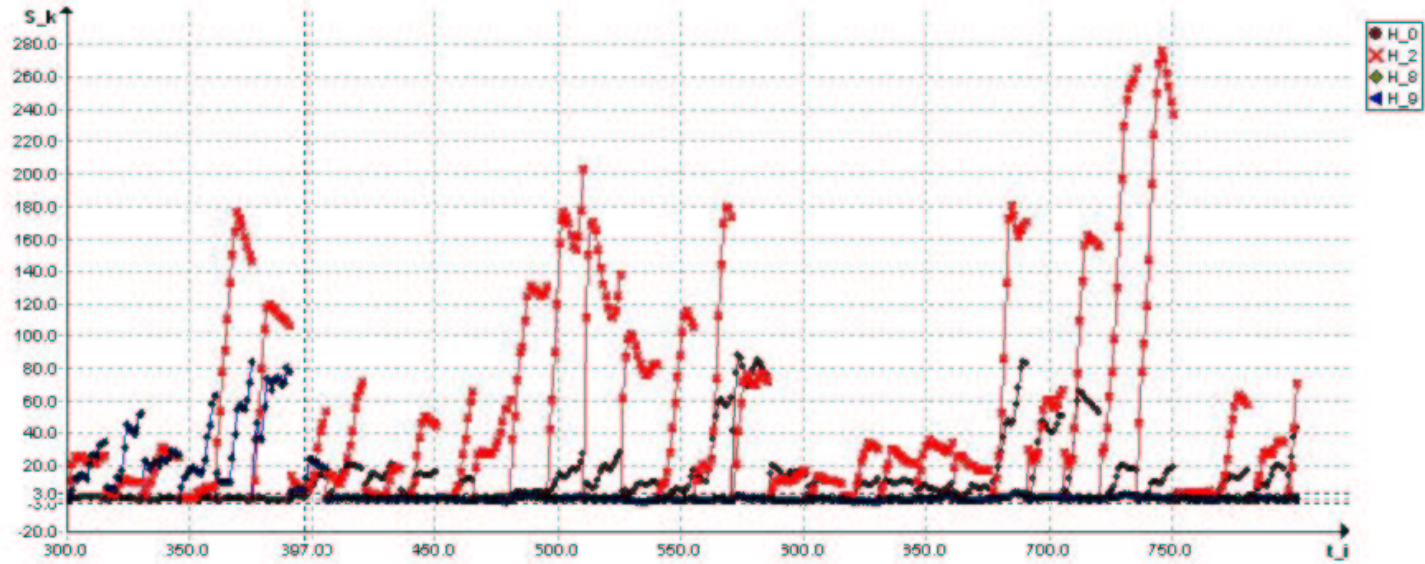


Рис. 2.11.  $S_0 \xrightarrow{397} S_9 \mid \mathcal{FP} = \{\mathcal{F}_0, \mathcal{F}_2, \mathcal{F}_8, \mathcal{F}_9\}$ . Объем выборки  $N = 15$  в Схеме В, (2.50)–(2.52). Замечание: В этом случае нарушение  $S_0 \xrightarrow{t_c} S_j$  возникает в момент времени  $t_c = 397$  при  $j = 9$ . Схема В (2.50)–(2.52) используется с каждой  $S_k^{(j)}$ ,  $j \in \{0, 2, 8, 9\}$ , и устанавливается на ноль через каждые  $N$  шагов алгоритма (2.46). Можно также получить отмасштабированную версию любого графика. Отмасштабированная версия для этого случая изображена на рис. 2.12 в форме скриншота

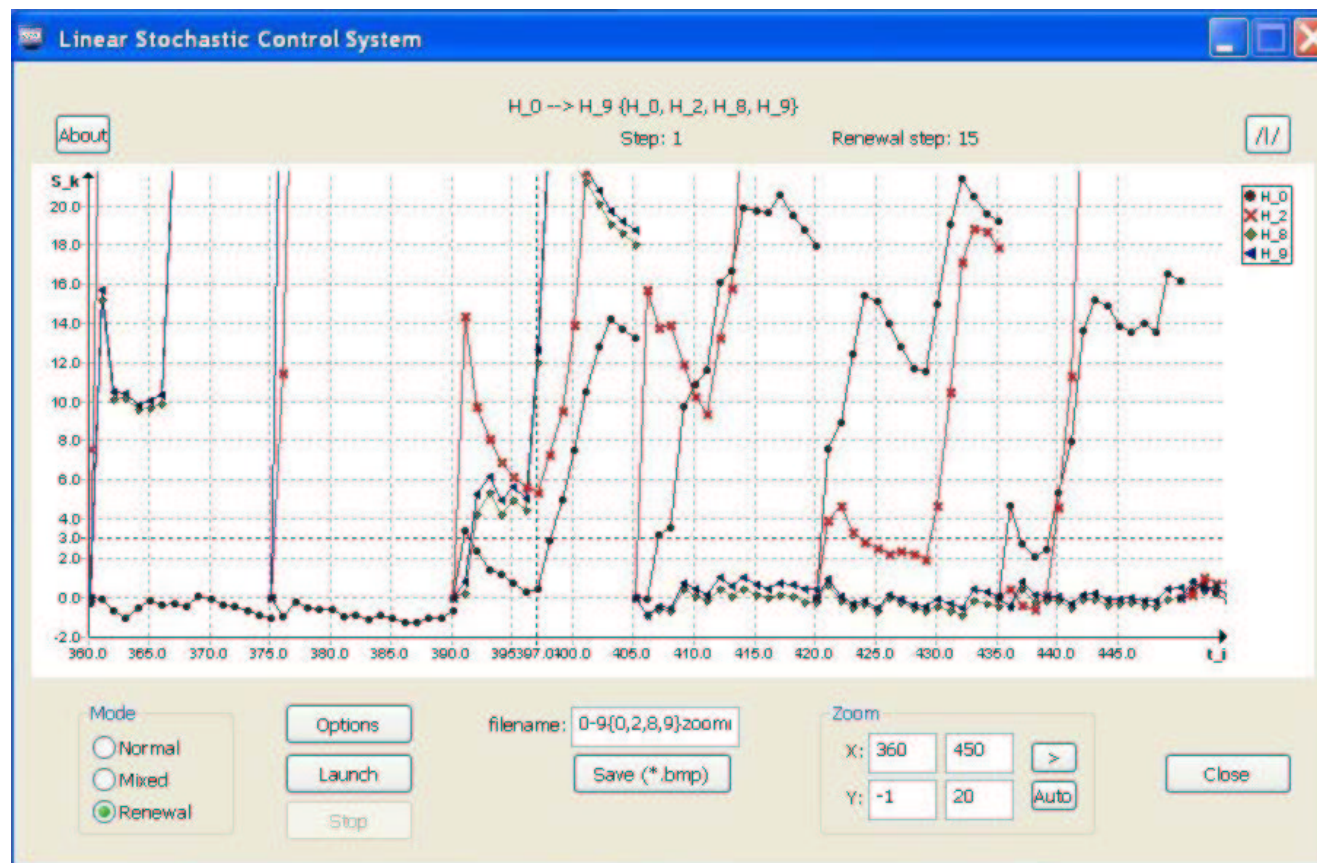


Рис. 2.12. Внешний вид приложения. Отмасштабированный график для рис. 2.11 с  $\mathcal{S}_0 \xrightarrow{397} \mathcal{S}_9 \mid \mathcal{FP} = \{\mathcal{F}_0, \mathcal{F}_2, \mathcal{F}_8, \mathcal{F}_9\}$

ным критерием Вальда, причем для случаев, когда нарушения могут возникать не только в матрице наблюдения  $H$ , но и в других матрицах системы.

### Приложение: Доказательство Теоремы 2.1

**Лемма 1** Если  $\xi \sim \mathcal{N}(0; 1)$ , тогда  $\mathbf{E} \{ \xi^{2k} \} = (2k - 1)!!$ .

*Доказательство.*

$$\begin{aligned} \mathbf{E} \{ \xi^{2k} \} &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} x^{2k} \exp \left\{ -\frac{x^2}{2} \right\} dx = \frac{2^k}{\sqrt{\pi}} \int_0^{\infty} t^{k-\frac{1}{2}} \exp \{ -t \} dt = \\ &= \frac{2^k}{\sqrt{\pi}} \Gamma \left( k + \frac{1}{2} \right) = \frac{2^k}{\sqrt{\pi}} \cdot \left( k - \frac{1}{2} \right) \cdot \Gamma \left( k - \frac{1}{2} \right) = \\ &= (2k - 1) \cdot (2k - 3) \cdot \dots \cdot 3 \cdot 1. \end{aligned}$$

*Доказательство теоремы.* Для гипотезы  $\mathbf{H}_0$   $\mu_i \sim \mathcal{N}(0; I)$  – гауссовская белая последовательность, где  $I$  – единичная матрица размера  $m \times m$ . Обозначим  $j$ -й элемент  $\mu_i$  через  $\mu_{ji}$ , тогда

$$\xi \triangleq \sum_{i=1}^k \sum_{j=1}^m \mu_{ji}^2 \sim \mathcal{F}_{\chi^2(mk)}(x),$$

где  $\mathcal{F}_{\chi^2(mk)}(x)$  – распределение  $\chi^2$  с  $mk$  степенями свободы. Когда  $k \rightarrow \infty$ , величина  $\xi$  распределена нормально с параметрами  $mk$  и  $2mk$ ,  $\mathcal{F}_{\chi^2(mk)}(x) \rightsquigarrow \mathcal{N}(mk; 2mk)$ . Из (2.34) имеем

$$S_k = \sqrt{mk/2} \left\{ \frac{1}{mk} \xi - 1 \right\},$$

следовательно,  $\mathcal{L}(S_k) \rightsquigarrow \mathcal{N}(0; 1)$ .

Для  $\mathbf{H}_1$  имеем

$$\begin{aligned} C_\mu &\triangleq \mathbf{E} \{ \mu_i \mu_i^T \} = L_0^{-1} C_1 L_0^{-T} = I + L_0^{-1} (C_1 - C_0) L_0^{-T} = I + \Delta, \\ m_s &\triangleq \mathbf{E} \{ s_k \} = \sqrt{m/2} [(1/m) \text{tr}\{C_\mu\} - 1] = (2m)^{-1/2} \text{tr}\{\Delta\}. \end{aligned}$$

Пусть  $[C_\mu]_{jk}$  –  $jk$ -й элемент матрицы  $C_\mu$ , тогда  $[C_\mu]_{jk} = \sigma_j \sigma_k \rho_{jk}$ , где  $\sigma_j^2$ ,  $\sigma_k^2$  суть дисперсии  $j$ -го и  $k$ -го элементов вектора  $\mu_i$  соответственно, а  $\rho_{jk}$  – коэффициент корреляции между ними. Из (2.34) следует, что

$$D_s = \frac{1}{2m} \left[ \mathbf{E} \left\{ (\mu_i^T \mu_i)^2 \right\} - (\mathbf{E} \{ \mu_i^T \mu_i \})^2 \right].$$

При помощи леммы 1 и непосредственных вычислений получаем

$$\mathbf{E} \{ \mu_{i,k}^4 \} = 3\sigma_k^4, \quad \mathbf{E} \{ \mu_{ji}^2 \mu_{ki}^2 \} = \sigma_j^2 \sigma_k^2 (1 + 2\rho_{jk}^2),$$

где  $j$  и  $k$  – индексы соответствующих элементов вектора  $\mu_i$ , так как

$$\begin{aligned}\mathbf{E} \left\{ (\mu_i^T \mu_i)^2 \right\} &= 3 \sum_{k=1}^m \sigma_k^4 + 2 \sum_{\substack{j,k=1 \\ j < k}}^m \sigma_j^2 \sigma_k^2 (1 + 2\rho_{jk}^2), \\ (\mathbf{E} \{ \mu_i^T \mu_i \})^2 &= \sum_{k=1}^m \sigma_k^4 + 2 \sum_{\substack{j,k=1 \\ j < k}}^m \sigma_j^2 \sigma_k^2, \\ D_s &= \frac{1}{m} \sum_{j,k=1}^m [C_\mu]_{jk}^2 = \frac{1}{m} \|C_\mu\|^2.\end{aligned}$$

Поскольку  $[C_\mu]_{jj} = 1 + \Delta_{jj}$  и  $[C_\mu]_{jk} = \Delta_{jk}$ , где  $\Delta_{jj}$  и  $\Delta_{jk}$  обозначают соответствующие вхождения в  $\Delta$ , то с учетом этих выражений получаем

$$D_s = 1 + (2/m) \text{tr}\{\Delta\} + (1/m) \|\Delta\|^2.$$

При помощи определения  $s_i$  (2.34) находим  $m_{S_k}$  (2.37) и дисперсию

$$D_{S_k} = D_s + 2 \sum_{j=1}^{k-1} (1 - j/k) \mathcal{K}_{ss}(j).$$

С учетом (2.33) находим

$$\lim_{k \rightarrow \infty} D_{S_k} = D_s(1 + r)/(1 - r).$$

Благодаря экспоненциальной корреляционной зависимости (2.33), условие перемешивания в теореме Биркгофа-Хинчина выполняется [27], [28], следовательно, сходимости (2.36) и  $P_D$  в (2.39) также выполняются.

## Библиографический список

1. Zhang L. et al. On rule checking and learning in an acupuncture diagnosis fuzzy expert system by genetic algorithm. In: *Proc. Fourth IEEE Interanational Conference on Fuzzy Systems*. Yokohama, 1995.
2. Buckley J.J., Reilly K.D. and Penmetcha K.V. Backpropagation and genetic algorithms for training fuzzy neural nets. In: *Genetic Algorithms and Soft Computing*, Eds F. Herrera and J. Verdegay. Physica Verlag, 1996.
3. Perneel C. et al. Optimization of fuzzy expert systems using genetic algorithms and neural networks. *IEEE Transactions on Fuzzy Systems*, **3**(3), 332–341, 1995.
4. Kahlert J. Programmsystem WinFACT. *VDE-Workshop "Regelungstechnische Programmpakete für IBM PC"*, Düsseldorf, 1993.
5. Maybeck P. *Stochastic models, estimation and control*. Academic Press, Vol. 1, 1979, Vol. 3, 1982.
6. Caines P. *Linear stochastic systems*. John Willey, 1988.
7. Beasley D., Bull D.R. and Martin R.R. An Overview of Genetic Algorithms: Part 1, Fundamentals. *University Computing*, **15**(2), 56–58, 1993.
8. Beasley D., Bull D.R. and Martin R.R. An Overview of Genetic Algorithms: Part 2, Research Topics. *University Computing*, **15**(4), 170–181, 1993.
9. Semoushin I.V. and Tsyganova J.V. Indirect Error Control for Adaptive Filtering. In: *Proc. Third European Conference on Numerical Mathematics and Applied Applications*, Eds. P. Neittaanmaki, T. Tiihonen and P. Tarvainen, World Scientific, 2000.
10. Semoushin I.V. and Tsyganova J.V. Kalman filter identifiability using API approach in control problems. In: *Proc. 13th International Conference on Systems Research, Informatics & Cybernetics*, Eds. A. Murgu and G. E. Lasker, The International Institute for Advanced Studies in Systems Research & Cybernetics: University of Windsor, 2002.

11. Antoniou A. *Digital filters: Analysis and design*. McGraw-Hill, 1979.
12. Kristinsson K. and Dumont G.A. System Identification and Control Using Genetic Algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, **22**(5), 1033–1046, 1992.
13. Neubauer André. The Circular Schema Theorem for Genetic Algorithms and Two-Point Crossover. *Genetic Algorithms in Engineering Systems: Innovations and Applications*, **446**, 209–214, 1997.
14. Semoushin I.V. Jointly Performed Computational Tasks in the Multi-Mode System Identification. *Lecture Notes in Computer Science*, **2658** 407–416, 2003.
15. Basseville M. and Nikiforov I. *Detection of abrupt changes: Theory and Applications*. Prentice-Hall, 1993.
16. Jones R.H., Crowell D.H. and Kapuniai L.E. Change detection model for serially correlated multivariate data. *Biometrics*, **26**(2), 269–280, 1970.
17. Borodkin L.I. and Mottl' V.V. Algorithm for finding the jump times of random process equation parameters. *Automation and Remote Control*, **37**(6), Part 1, 23–32, 1976.
18. Segen J. and Sanderson A.C. Detecting changes in a time series. *IEEE Trans. Information Theory*, **IT-26**(2), 249–255, 1980.
19. Mehra R.K. and Peschon J. An innovations approach to fault detection and diagnosis in dynamic systems. *Automatica*, **7**, 637–640, 1971.
20. Boozer D.D. On innovation sequence testing of the Kalman filter. *Teledyne Brown Engineering*, **1**, 251–256, 1971.
21. Mosca E. *Optimal, predictive, and adaptive control*. Prentice Hall, 1995.
22. Roussas G.G. *Contiguity of probability measures. Some applications in statistics*. Cambridge University Press, New York, 1972.
23. Van Trees H.I. *Detection, estimation, and modulation theory*. John Wiley and Sons, Inc., Part I: *Detection, estimation, and linear modulation theory*, 1968.
24. Semoushin I.V. Optimality testing for the adaptive Kalman filter by the use of a realization of a scalar process. *Transactions of The USSR Academy of Sciences – Technicheskaya Cybernetika*, **6**, 195–198, 1979.

25. Martin W.C. and Stubberud A.R. Innovation process in identification problems. In: C.T. Leondes (ed.) *Control and dynamic systems – Advances in theory and applications*. Academic Press, 1976.
26. Gaines H.T. Flight test of a Kalman filter Doppler aided strapdown inertial navigator. *Proc. Nat. Aerospace Electron. Conf.*, N.-Y., 1971.
27. Yoshihawa K. *Weakly dependent stochastic sequences and their application*. Tokyo, Vol. **1.4**, 1992–1996.
28. Ibragimov I.A. and Linnik Yu.V. *Independent and stationary related values*. Nauka, 1965. [*in Russian*]
29. Semoushin I.V., Yurjev A.D. and Nikonorov A.V. Built-in selection of the best adaptation mechanism for INS error model identification. In: P. Neittaanmäki, T. Rossi, S. Korotov, E. Oñate, J. Périaux, and D. Knörzer (eds) *Proceedings of the 4th European Congress on Computational Methods in Applied Sciences and Engineering, ECCOMAS 2004*, Vol. **II**, 996.pdf, ISBN 951-39-1869-6.



### **Глава 3. МЯГКИЕ ВЫЧИСЛЕНИЯ В ПРОЕКТИРОВАНИИ ВЫЧИСЛИТЕЛЬНЫХ СЕТЕЙ**

#### **3.1. Этапы проектирования вычислительных сетей. Место технологий мягких вычислений на разных этапах проектирования**

Вычислительная сеть (ВС) представляет собой эволюционирующий объект, который за время эксплуатации переживает несколько модификаций. Условия модификации существенно отличаются от условий проектирования тем, что ВС в текущем состоянии доступна для измерений. Результаты измерения параметров трафика и эксплуатационных параметров ВС могут быть использованы для прогнозирования параметров ВС в новом послепроектном состоянии. При проектировании с нуля гипотетические параметры могут быть получены в результате вычислительного эксперимента в ходе имитации или в результате экстраполяции результатов какого-то “типового” варианта на данный. Программа оптимизации в контексте настоящей постановки предназначена для ВС, которые можно разделить на два типа: ВС с маршрутизаторами; ВС без маршрутизаторов. Сеть представляется на уровне организации передающей Среды, то есть на физическом, канальном, сетевом, транспортном уровнях по классификации OSI/ISO. Оптимизация сеансового, уровня представления данных и прикладного уровней требует анализа программного обеспечения и режима эксплуатации прикладных задач.

Таблица 3.1.

Задачи и математические (алгоритмические) средства их решения.

Задача	Предлагаемое средство решения
Размещение (оптимизация) размещения технологического оборудования: коммутаторов/ концентраторов	Нечеткое линейное программирование
Выбор типа коммуникационного оборудования, перепроектирование топологии (изменение структуры каналов и переподключение узлов)	Алгоритм полного перебора или генетический алгоритм

Задача оптимального размещения коммутационного оборудования может быть поставлена и решена как задача возможностного программирования.

## **3.2. Система проектирования ВС на основе потоковых диаграмм данных**

### **3.2.1. Общая характеристика проблемы проектирования**

Проектирование сложных технических систем, таких как локальные, корпоративные и телекоммуникационные вычислительные сети – сложный многоуровневый процесс. Он заключается в построении оптимальной системы, максимально использующей свои ресурсы. В настоящий момент не существует четко определенной методики проектирования сетей, дающей оптимальный результат. Сейчас работы больше направлены в сторону моделирования уже существующей сети для проверки ее эффективности и выявления ошибок. При моделировании сети человек обладает рядом статистических данных, на основе которых он может делать оптимизацию модели. При проектировании же сети таких данных нет, и человек может лишь предвидеть, прогнозировать, как будет загружен тот или иной сетевой канал, насколько сильной будет занятость того или иного узла сети. Подобные прогнозы представляют собой некоторые лингвистические формы, которыми можно оперировать, используя аппарат нечетких вероятностных величин. Необходимо так же правильно строить сам процесс проектирования сети. Прежде всего нужно строить имитационную модель тех процессов, которые будут в ней происходить. Только осознавая круг задач сети, можно построить ее наиболее эффективно. Прикладная модель должна содержать и прогнозные данные о сети, которые используются при ее дальнейшей оптимизации.

Учитывая огромное распространение локальных, глобальных и телекоммуникационных вычислительных сетей, необходимо развивать категорию САПР, относящуюся к ним. Существует несколько различных систем, позволяющих моделировать процессы, происходящие в вычислительных сетях на физическом уровне. Однако до сих пор не создавался (или не получил широкого распространения) инструмент, который не просто моделирует, но и проектирует вычислительную сеть на всех ее уровнях.

Сейчас корпоративные сети создаются стихийно, привязываясь лишь к пространственному фактору. Сетевые администраторы опираются лишь на расположение вычислительных машин в кабинетах, залах и на этажах. В таком подходе кроется множество подводных камней – ошибок, которые становятся критичными при дальнейшем росте и развитии сети. Порой приходится переделывать целые сегменты вычислительной сети из-за того, что начальные этапы ее создания были слишком узки и не рассчитывались на развитие.

Первоначально необходимо создавать проект сети, с возможностью моделировать и оптимизировать его, до того, как начаты работы по созданию физической сети. Однако среди существующих программных пакетов не представ-

лены подобные решения. Не осознан до конца и сам процесс подобного проектирования, его этапы и методика. Основные методы моделирования и перестроения сетей построены на том, что с существующей сети путем многократных статистических измерений, снимаются данные о трафике, вычислительной нагрузке узлов. Затем эти данные анализируются, и эксперт выдает рекомендации по перестроению сегментов или всей сети в целом. Зачастую в этом процессе не учитывается структура потоков данных на предприятии, структура его бизнес-процессов, которые и являются основными источниками трафика и вычислительной нагрузки системы. Кроме этого, при проектировании сети с нуля, измерений для анализа не имеется, и проектировщик может оперировать лишь прогнозами. Следовательно, необходимо в систему проектирования вводить интеллектуальные составляющие, которые позволят даже на основании прогнозных данных вырабатывать оптимальные решения.

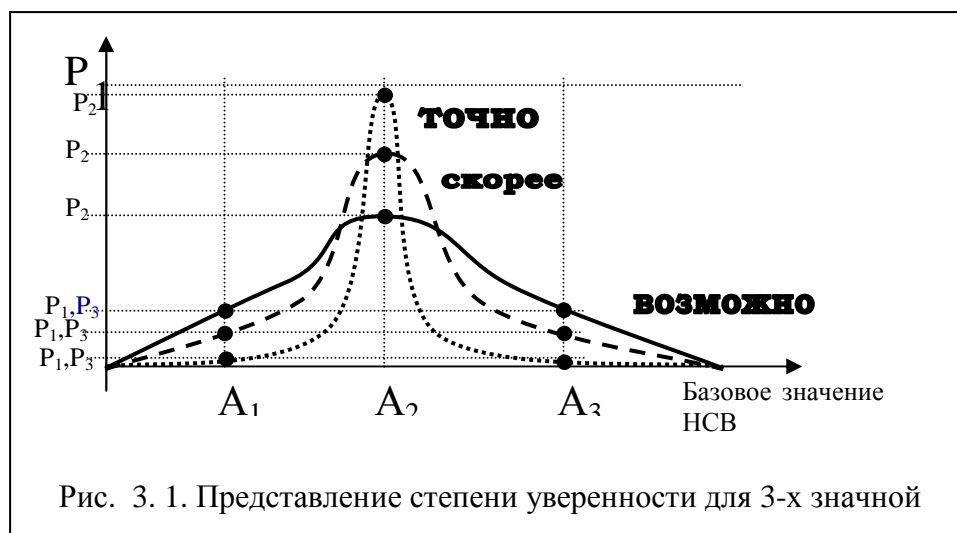
Для достижения поставленной цели необходимо решить ряд задач:

1. Необходимо провести сравнительный анализ существующих систем моделирования вычислительных сетей, а также анализ существующих стандартов имитационного моделирования.
2. Необходимо построить методику проектирования вычислительных сетей на уровне бизнес-процессов и физической структуры сети.
3. Необходимо адаптировать выбранный стандарт имитационного моделирования к моделированию вычислительных сетей.
4. Необходимо разработать модель обработки прогнозных данных о сети.
5. Необходимо разработать алгоритм оптимизации проектных результатов.

### 3.2.2. Нечеткое описание трафика вычислительной сети

Основными характеристиками для сети являются трафик и вычислительная нагрузка узлов. Трафик указывает на количество передаваемой по каналам сети информации, а вычислительная нагрузка (ВЗ) – на занятость вычислительной машины, как узла сети.

Необходимо понимать нечеткую и вероятностную природу этих двух параметров. Когда речь идет о проектировании сети с нуля, сетевой администратор не обладает никакой информацией о трафике и ВЗ и работает в условиях полной неопределенности. Однако, он может делать прогнозы о том, какой будет трафик на том или ином канале, и какой узел будет загружен и в какой степени. Трафик и ВЗ, как физические величины, по природе своей непостоянны, и зависят от множества внешних и внутренних факторов. В связи с этим оценивать трафик в единичный момент времени нельзя, а можно лишь давать интервальную, нечеткую оценку, оперируя такими нечеткими понятиями, как «трафик высокий» или «трафик низкий». В том случае, когда человек не просто оценивает подобный критерий, но еще и делает его прогноз в условиях неопределенности, нечеткой оценки недостаточно, необходимо и к ней добавить вероятностную оценку.



Таким образом, наиболее приближенным к условиям реальности прогноз вида «трафик скорее высокий» или «трафик точно низкий». Первое слово в оценке отражает вероятность, а второе интервальную оценку прогноза трафика или вычислительной загрузки.

Математически обрабатывать такие вероятностные нечеткие величины предлагается используя математический аппарат, разработанный А. В. Язениным.

В общем случае дискретная нечеткая случайная величина (НСВ) имеет вид:

$$Tr = \{A_1/P_1, A_2/P_2, \dots, A_n/P_n\},$$

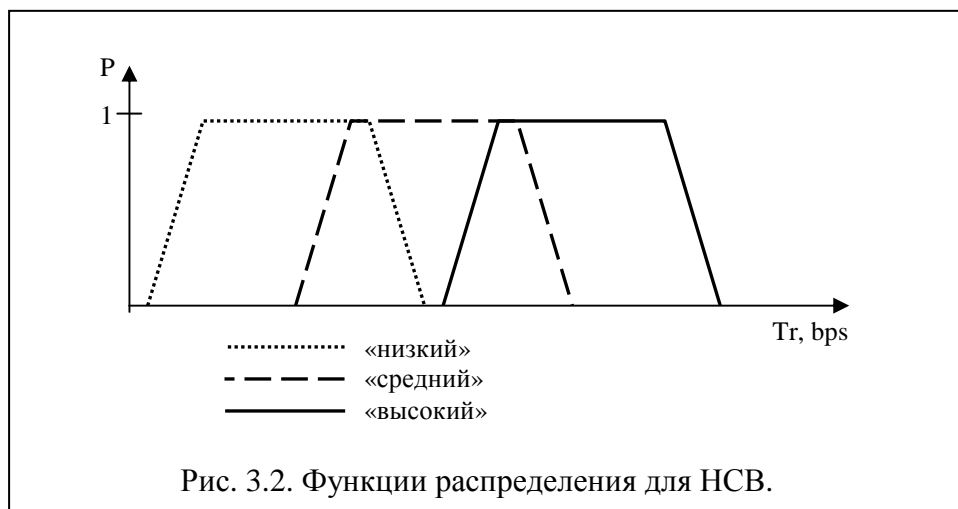
где  $A_1, A_2, \dots, A_n$  – нечеткие значения, которые величина принимает с вероятностями  $P_1, P_2, \dots, P_n$ . Значения НСВ являются взаимоисключающими, поэтому сумма вероятностей этих значений должна быть равна 1, то есть должно выполняться условие нормирования НСВ:  $P_1 + P_2 + \dots + P_n = 1$ .

Поскольку при проектировании вычислительной сети человек выдает прогноз словом, набор вероятностей  $P_1, P_2 \dots P_n$  можно также перевести в лингвистическую форму. Для этого вводится понятие «степень уверенности НСВ». Каждый вектор вероятностей кодирует степень уверенности НСВ, которая представляет собой лингвистическую оценку вида «точно», «скорее всего» («скорее»), «наверное» («возможно»). Каждую такую оценку можно представить в виде функции распределения НСВ (рис. 3.1).

Принято использовать три нечетких значения сетевого трафика: «высокий», «средний» и «низкий», которые определяются трапецевидными функциями принадлежности. В связи с этим трафик можно записать так:

$$Tr = \{ \text{«низкий»}/ P_1, \text{«средний»}/ P_2, \text{«высокий»}/ P_3 \}.$$

Здесь нечеткие множества заданы трапециевидными функциями принадлежности (рис. 3.2).



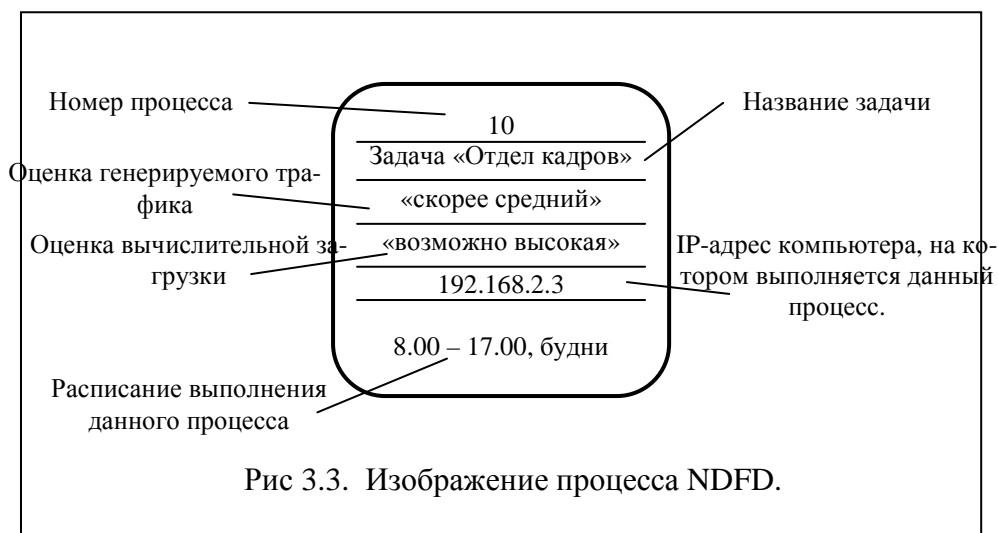
Степень уверенности кодируется в этом случае, как и на рис. 3.1, векторами вероятностей как показано в таблице 3.1.

Таблица 3.2

Базовое значение НСВ Степень уверенности НСВ	Значения трафика		
	«низкий»	«средний»	«высокий»
«точно»	{1,0,0}	{0,1,0}	{0,0,1}
«скорее»	{0.75,0.25,0}	{0.125,0.75,0.125}	{0,0.25,0.75}
«возможно»	{0.5,0.35,0.15}	{0.25,0.5,0.25}	{0.15,0.35,0.5}

Начинать проектирование ВС следует с создания функциональной модели бизнес-процессов сети. Параллельно создается структурная модель сети, и затем происходит слияние двух этих моделей с целью получения оптимального проектного результата.

Для функционального моделирования бизнес-процессов выбран стандарт DFD, который модифицирован до стандарта NDFD (Network DFD) (рис. 3.3). Поскольку любой узел сети работает как генератор сетевого трафика и вычислительной загрузки, то из всех сущностей стандарта DFD оставлена одна сущность - «процесс». К ее обычным свойствам добавлены нечеткие вероятностные оценки сетевого трафика и вычислительной загрузки, а также расписание работы данного процесса. Последнее является необходимым в связи с тем, что узлы сети генерируют трафик непостоянно, их сетевая и вычислительная загруженность привязана к временным интервалам, связанным с режимом работы данного рабочего места.



Слияние функциональной и структурной моделей проектируемой сети предлагается производить методом стандартного генетического алгоритма (СГА) для получения оптимального результата уже на начальных стадиях проектирования.

Хромосома СГА для данного приложения СГА состоит из двух частей – изменяемой и неизменяемой. В качестве изменяемой части выбрана кодировка структурной модели, а в неизменяемой части кодируется функциональная модель. Фактически СГА отыскивает наиболее оптимальное расположение блоков спроектированной функциональной диаграммы на спроектированной же структурной модели сети.

Расчет качества каждой хромосомы СГА как варианта слияния функциональной и структурной моделей производится при помощи имитационного моделирования. В каждом случае прорабатывается вероятностная составляющая суммарной оценки трафика или вычислительной загрузки сети.

### 3.2.3. Реализация САПР вычислительных сетей на основе потоковых диаграмм

Наиболее удобной реализацией предлагаемой методики проектирования является возможность параллельного составления функциональной и структурной модели. В этом случае проектировщик постоянно имеет перед собой полную картину задач проектируемой сети, а также возможности по составлению физической модели.

Интерфейс системы состоит из двух графических редакторов. В первом создается функциональная модель бизнес-процессов, происходящих в сети. Пользователю дается возможность расставлять блоки на рабочем поле, соединять их потоками, а также указывать прогнозные оценки трафика и ВН. Здесь же проектировщик формирует расписание работы каждого узла в отдельности.

Параллельно производится работа во втором графическом редакторе, предназначенном для построения структурной модели физической сети. Здесь

проектировщик выстраивает узлы, коммутирующие элементы и маршрутизаторы, соединяя их каналами. Здесь же отлаживается таблица маршрутизации. Как и при построении реальной сети администратор назначает каждому проектируемому узлу свой IP-адрес с назначением основного шлюза. На маршрутизаторах выстраивается таблица маршрутизации. Корректная настройка маршрутизации очень важна, так как она используется при трассировке связей бизнес-диаграммы по физической модели сети.

Финальным этапом построения проекта сети является поиск наиболее оптимального расположения узлов бизнес-диаграммы по структуре физической сети. Эта задача возложена на адаптированный к ней стандартный генетический алгоритм. В системе имеется возможность настройки СГА по параметрам размера популяции, порога стабильности и количества шагов эволюционного времени. Хромосомой для СГА является единичный вариант расположения блоков бизнес-диаграммы относительно узлов проектируемой сети. Оценка качества хромосом происходит по следующему алгоритму:

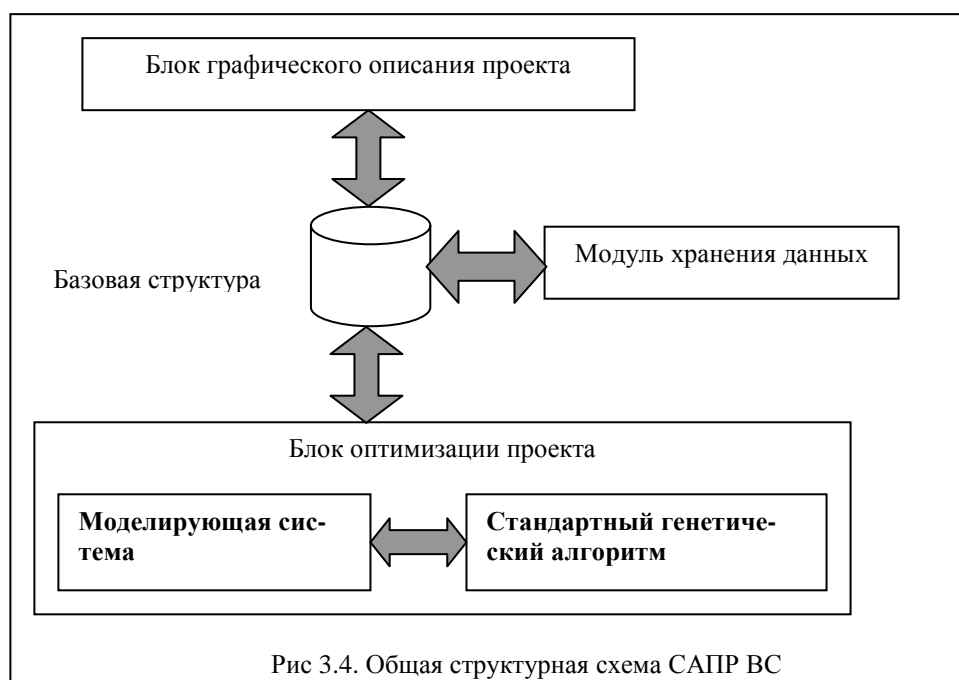
1. Трассировка каждой связи бизнес-диаграммы по спроектированной структурной модели сети. При трассировке учитывается та оценка сетевой загрузки, которая указана в блоке, являющимся источником данной связи. Эта указанная прогнозная вычислительная загрузка плюсуется в суммарные загрузки на всех каналах сети, по которым проходит данная связь. Фактически происходит подсчет суммарного трафика на каждом из каналов сети.
2. Подсчет суммарных вычислительных загрузок узлов сети. Нагрузка на узел сети складывается из установленных прогнозных оценок нагрузок тех блоков диаграммы, которые попали на этот узел.
3. Вычисление общего качества полученного варианта. Среди всех каналов сети находится тот, на котором по результатам трассировки получилась наибольшая сетевая загрузка. Среди узлов сети находится тот, на котором появилась наибольшая вычислительная загрузка. Сумма этих двух параметров и является оценкой качества исследуемой хромосомы СГА.

При обработке популяции СГА качественный параметр хромосомы тем лучше, чем он меньше. СГА осуществляет поиск того варианта, в котором сетевой трафик и вычислительная загрузка узлов наиболее распределены по системе.

Адаптация СГА к данной задаче заключается в построении хромосомы.

В реализованной системе хромосома состоит из двух частей – изменяемой и неизменяемой. В неизменяемой части по порядку указаны порядковые номера узлов сети, назначающиеся системой автоматически. В изменяемой части хромосомы располагаются порядковые номера блоков бизнес-диаграммы. Операции рекомбинации и мутации производятся с изменяемой частью, при чем в процессе обработки популяции перед расчетом качества хромосом происходит их проверка на корректность. Поскольку блоки диаграммы являются уникаль-

ными – их повторение в изменяемой части хромосомы недопустимо. Проверка на корректность отсеивает те хромосомы в изменяемой части, в которых присутствуют повторяющиеся блоки. При начальной генерации популяции так же происходит отсеивание таких хромосом.



Программная реализация построена с использованием объектно-ориентированного программирования. Интерфейс построен на основных визуальных компонентах Windows в среде Delphi 5.0. Основной интерфейсной частью являются специализированные графические редакторы, в которых строятся бизнес-диаграмма и структурная модель ВС (рис. 3.4).

В программной реализации системы используется механизм динамического распределения памяти. Все данные о моделях хранятся в динамических структурах данных.

Результаты внедрены в проектирование корпоративных вычислительных сетей на ОАО «Ульяновское конструкторское бюро приборостроения» (УКБП) и на ОАО «Ульяновский завод тяжелых и уникальных станков» (УЗТС).

ОАО УКБП – достаточно большое предприятие, имеющее локальную сеть на 300 вычислительных машин. При помощи предлагаемой САПР ВС было произведено начальное проектирование корпоративной вычислительной сети, а затем, в связи с быстрым ростом числа приобретаемых предприятием компьютеров, и полное перепроектирование.

На ОАО «УЗТС» при помощи предлагаемой системы было произведено проектирование корпоративной сети с нуля.



### **3.3. Проектирование вычислительных сетей на основе байесовских сетей доверия**

#### **3.3.1. Архитектура САПР ВС**

Создание крупных высокоэффективных систем обработки данных связано с объединением средств вычислительной техники, обслуживающей подразделения предприятий и организаций, с помощью системы передачи данных в единую вычислительную сеть. В настоящее время при проектировании вычислительных сетей активно используются САПР ВС. Поскольку процесс проектирования ВС является сложным и многоступенчатым процессом, связанным с принятием проектных решений, современная САПР ВС должна обладать интеллектуальностью.

Существующие современные САПР ВС позволяют выполнять построение структуры ВС с использованием различного оборудования, определять параметры спроектированной ВС и проводить ее моделирование, также некоторые САПР позволяют производить оптимизацию ВС. Среди недостатков существующих САПР следует отметить следующие: отсутствие привязки проектируемой ВС к автоматизируемым бизнес - процессам, необходимость задания трафика в четкой числовой форме. Кратко рассмотрим эти недостатки. Отсутствие привязки создаваемой ВС к автоматизируемым бизнес - процессам затрудняет модернизацию ВС по мере роста автоматизируемого предприятия или изменения бизнес - процессов. Трафик реальной вычислительной сети имеет нечеткий вероятностный характер, нечеткость трафика образуется вследствие неизвестности размера передаваемых по сети запросов к базам данных, а вероятностный характер обеспечивается вероятностью начала выполнения задачи, в ходе выполнения которой передаются запросы. Необходимость задания трафика в четкой числовой форме ограничивает способности САПР ВС к адекватному представлению динамики трафика на каналах ВС. Таким образом, в настоящее время существует потребность в САПР ВС, которая позволяет проектировать ВС исходя из информации об автоматизируемых бизнес - процессах, имеет возможность представления трафика в нечеткой вероятностной форме, использует наряду с генетическими алгоритмами иные методы оптимизации параметров ВС. В разработанном прототипе САПР ВС в качестве иных методов оптимизации используется механизм байесовских сетей доверия.

Представляемая САПР ВС позволяет произвести весь процесс создания вычислительной сети: от определения автоматизируемых процессов до оптимизации состава коммуникационного оборудования и подключения рабочих станций, определения пропускной способности каналов сети и размещения узлов сети. Интеллектуальность представленной САПР ВС сосредоточена в модуле байесовских сетей доверия.

Функционально разработанный прототип САПР ВС состоит из следующих модулей: DFD – диаграммер, мастер перехода, редактор ВС, модуль имитационного моделирования, модуль статического моделирования, модуль генетической оптимизации и модуль байесовской оптимизации. Связи между модулями изображены на рисунке 3.5:

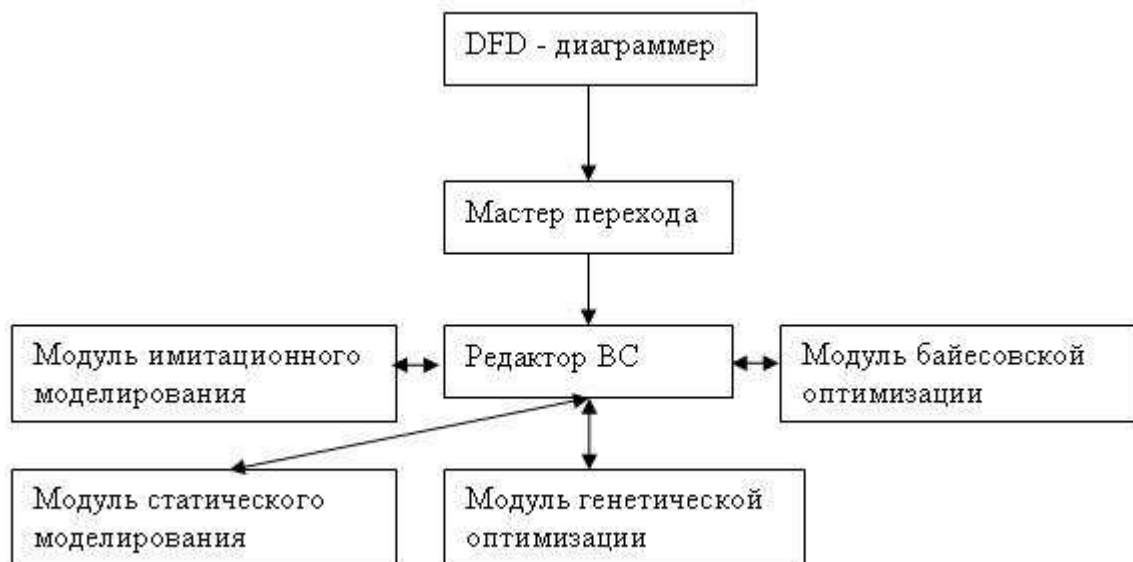


Рис. 3.5. Архитектура САПР ВС

Рассмотрим функции каждого из модулей:

DFD – диаграммер предназначен для построения DFD - диаграмм, описывающих бизнес - процессы. Мастер перехода предназначен для построения чернового варианта ВС на основании DFD - диаграмм и вводимой пользователем САПР информации. Редактор ВС - для визуализации текущего варианта ВС и сбора информации из остальных модулей. Модуль динамического моделирования определяет распределение трафика по каналам сети в реальном времени. Модуль не предназначен для определения пикового трафика. Модуль статического моделирования ВС определяет пиковый трафик ВС, получаемый при одновременном запуске всех выполняемых в сети задач. Модуль генетической оптимизации проводит генетическую оптимизацию следующих параметров: подключение рабочих станций и размещение коммуникационного оборудования. Модуль байесовской оптимизации - основная интеллектуальная составляющая САПР, предназначенная для проведения оптимизации состава коммуникационного оборудования и пропускной способности каналов сети.

В САПР ВС трафик представляется как последовательность передаваемых по каналам сети запросов, документов и ответов на них, которые также представлены документами и запросами. Размер передаваемых документов является величиной четкой, а размер запросов является величиной нечеткой. Это придает трафику нечеткий характер.

Трафик представлен в виде нечеткой величины с функциями трапецевидной формы. Такое решение позволяет упростить реализацию операций с трафиком и в то же время позволяет обеспечить покрытие нечеткими функциями всего интервала значений трафика. Вероятностные значения трафика определяются при проведении оптимизации с использованием байесовских сетей доверия.

В САПР ВС задано четыре градации трафика: «Крошечный», «Малый», «Средний», «Высокий». Эти нечеткие функции выбраны, с одной стороны, потому что они покрывают все возможные значения трафика, и с другой стороны, обеспечивают достаточную гибкость для работы с ним.

В представленной САПР для моделирования работы сети используются две модели: динамического и статического моделирования.

### 3.3.2. Динамическая модель элементов САПР ВС

В основе динамической модели лежит представление всех элементов проектируемой ВС как взаимодействующих потоков

Динамическая модель ВС состоит из трех множеств: множество компьютеров (включает в себя рабочие станции и сервера), множество узлов (включает в себя как сами узлы, так и различные мосты и маршрутизаторы на основе маломощных компьютеров) и множества каналов (всевозможная среда передачи данных). Ограничения в данной модели определяются лишь параметрами компьютера, на котором производится имитационное моделирование. Основное ограничение заключается в том, что элементы модели и служебная информация должна размещаться в оперативной памяти.

В случае нарушения данного условия часть элементов или служебной информации модели окажется в файле подкачки, что неминуемо приведет к задержкам в обработке и замедлению работы модели или нарушению адекватности.

Основной характеристикой компьютера является уникальное в пределах сети имя (в качестве его можно использовать IP-адрес). Необходимость уникального имени, отличного от IP-адреса, связана с маршрутизацией вида, отличающегося от ТСР/IP. Дополнительными характеристиками компьютера являются его тип (рабочая станция или сервер), хранимые базы данных, работающие на компьютере пользователи. Для моделирования загруженности процессора компьютер характеризуется индексом производительности.

Назначение компьютера состоит в генерации, обработке и поглощении трафика. Обработкой трафика занимаются серверы, рабочие станции генерируют поток документов и запросов пользователей и поглощением ответов на посланные ранее запросы.

Узлы ВС характеризуются типом (концентратор, коммутатор, маршрутизатор), количеством портов и уникальным именем. Размер буфера памяти каждого порта принят неограниченным с целью упрощения процедуры маршрути-

зации. Уникальность имени требуется для упрощения реализации протокола моделирования. Модель работы концентратора представляет собой схему «общая шина», при которой поступающий на любой порт пакет копируется на все остальные порты. Модель работы коммутатора следующая: если поступающий на порт пакет предназначен для компьютера, находящегося в подсети коммутатора, то он транслируется в порты подсети, в противном случае он транслируется в порты, не принадлежащие подсети. Модель работы маршрутизатора следующая: для поступающего на порт устройства пакета по внутренней таблице ищется порт назначения. Если такой порт найден, то пакет отсылается на этот порт, в противном случае он отсылается на все порты.

Каналы ВС представлены мьютексами. Такая реализация объясняется тем, что в каждый конкретный момент времени каналом может владеть только одно устройство. Единственное назначение канала состоит в получении и передаче данных по запросам.

Модель сети для статического моделирования состоит из трех множеств: множества компьютеров, множества узлов и множества каналов.

Множество компьютеров разделяется на множество рабочих станций и множество серверов данных. На рабочих станциях пользователи выполняют задачи, заданные в DFD - диаграммах. Серверы данных хранят базы данных, с которыми взаимодействуют задачи, выполняемые пользователями. Задача сервера заключается в получении элементов задач от рабочих станций, генерации ответа на полученные элементы и посылка сгенерированных ответов отправителю элемента.

Задача каналов сети заключается в соединении элементов задач между собой.

Основная идея алгоритма статического моделирования заключается в определении влияния взаимодействия двух компьютеров на глобальное распределение трафика на каналах ВС.

Для оптимизации размещения коммуникационного оборудования и подключения рабочих станций в САПР ВС используются генетические алгоритмы.

### 3.3.3. Генетическая оптимизация вычислительной сети

Основная идея генетического алгоритма, решающего задачу размещения коммуникационного оборудования, состоит в разбиении рабочего поля на 1024 части по обоим измерениям. Каждому узлу приписываются два коэффициента, описывающих местоположение узла. Таким образом, каждому узлу сети в хромосоме отвечает 20 генов (2 отрезка по 10 генов, каждый из отрезков дает 1024 варианта). Это справедливо для двумерной оптимизации, в трехмерном случае необходимо разбивать по трем измерениям, и каждому узлу сети в хромосоме будет отвечать 30 генов. Далее работает стандартный генетический алгоритм. Функцией оптимальности является длина всех соединительных кабелей, алгоритм направлен на минимизацию функции оптимальности.

В процессе работы алгоритм запоминает обработанные состояния, и в случае их повторения значения функции оптимальности получаются из сохраненных хромосом. В алгоритме заложена возможность размещения оборудования в определенных пользователем областях. Для получения такого эффекта функция оптимальности каждой хромосомы удваивается при наличии факта нахождения какого-либо из узлов вне заданных областей (если промахнулся один узел, функция удваивается, если два узла - увеличивается в четыре раза и так далее). Таким образом, в процессе поиска минимума алгоритм автоматически загонит все узлы в рамки заданных ограничений (аналогично можно сделать для трехмерного случая, хотя сложность задания областей возрастет многократно).

Основная идея алгоритма для решения задачи оптимизации подключения рабочих станций заключена в том, что каждому компьютеру сети присваивается номер узла, к которому он подключен. Этот номер является геном хромосомы алгоритма. Длина хромосомы равна количеству компьютеров в сети.

Основным недостатком алгоритма является то, что функцией оптимальности является максимальный трафик на каналах сети. Из этого следует, что для оценки оптимальности хромосомы необходимо производить статическое моделирование варианта сети, представленного хромосомой. С целью преодоления этого недостатка в алгоритм введен список промоделированных хромосом. Применение данного списка обусловлено частым повторением одних и тех же хромосом, особенно в конце процесса оптимизации.

### 3.3.4. Определение и механизм работы байесовских сетей доверия

Кратко рассмотрим структуру и механизм работы байесовских сетей доверия. Байесовская сеть доверия (БСД) представляет собой направленный ациклический граф, дуги которого представляют собой причинно - следственные связи, направленные от причин к следствиям, а узлы представляют собой состояния переменных, характеризующих конкретную задачу [3]. Базовым положением БСД является гипотеза о том, что человек – эксперт способен формулировать правила, связывающие не более двух – трех элементарных утверждений. Таким образом, база знаний должна состоять из частей, соответствующих этим правилам. Между такими частями базы знаний также существуют определенные взаимоотношения. Такое разбиение базы правил представлено графом БСД. Исходным данным соответствуют начальные узлы БСД, а результаты работы сети определяются на конечных узлах сети, представляющих переменные, характеризующие конкретную задачу.

В качестве меры истинности в аппарате БСД используется мера вероятности. Связь между переменными, участвующими в правиле, передается с помощью тензора условной вероятности. Например, при связи от  $x$  и  $y$  к  $z$  соответствующий тензор представлен в виде  $p(\tilde{z} | \tilde{x}, \tilde{y})$ , где  $\tilde{x} = (x, \bar{x})$ . Каждый тензор условной вероятности представляет собой правило, согласно которому опреде-

ляется значение выходной переменной по значениям входных переменных. При поступлении на вход сети значений исходных данных в БСД инициируется процесс распространения вероятностей. По завершении процесса распространения в конечных узлах сети формируются значения переменных, характеризующих решаемую задачу. Структура БСД и тензоры условных вероятностей задаются экспертом, что позволяет обосновать полученные результаты мнением эксперта.

При поступлении в БСД нового свидетельства о состоянии узла БСД, узел начинает передавать сообщения, представляющие собой степень поддержки данного свидетельства относительно истинности всех состояний узлов – предков. Этот процесс возбуждает пересчет вероятностей в других узлах сети, и они также начинают передавать сообщения своим соседям. Так как граф является ациклическим, то в пересчетах оценок наступает равновесие – сообщения перестают распространяться и величины вероятностей перестают изменяться. В работе [1] представлен алгоритм распространения вероятностей в БСД, описанный ниже.

Пример типичной байесовской сети приведен на рисунке 3.6:

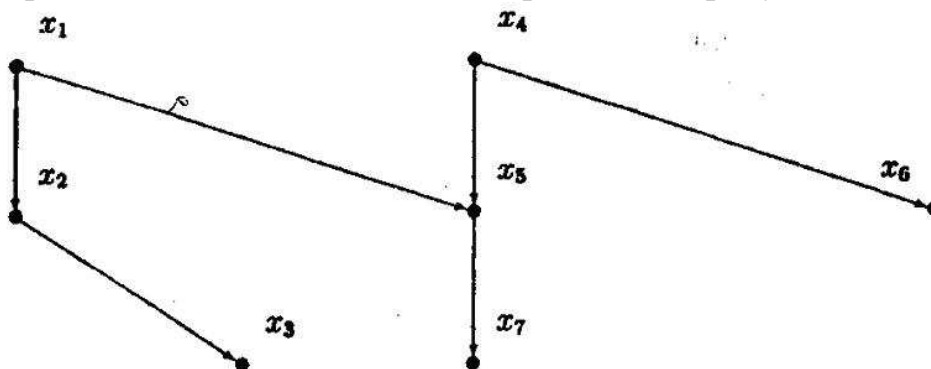


Рис. 3.6. Пример БСД

Эта сеть представляет собой следующее распределение:

$$p(\tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \tilde{x}_4, \tilde{x}_5, \tilde{x}_6, \tilde{x}_7) = p(\tilde{x}_1) * p(\tilde{x}_4) * p(\tilde{x}_2 | \tilde{x}_1) * p(\tilde{x}_3 | \tilde{x}_2) * p(\tilde{x}_5 | \tilde{x}_1 \tilde{x}_4) * p(\tilde{x}_7 | \tilde{x}_5).$$

В данной задаче БСД используется как интерпретирующая машина, а именно, вновь поступившая информация будет инициировать процесс пересчета вероятностей, продолжающийся до тех пор, пока сеть не придет в новое равновесное состояние.

Рассмотрим процесс пересчета вероятностей в БСД. Схематически алгоритм обновления вероятностей представлен на рисунке 3.7:

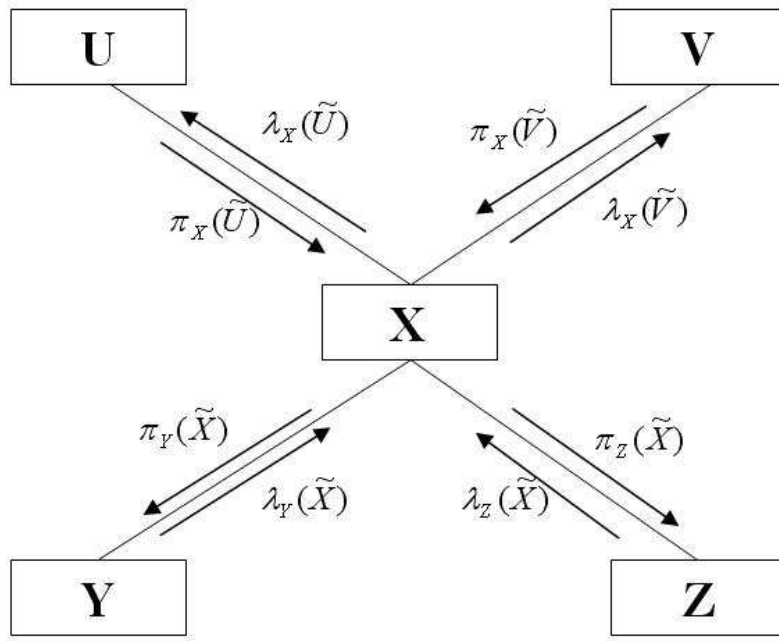


Рис. 3.7. Структура алгоритма пересчета вероятностей в БСД

Пусть  $Bel[A_i] = p(A_i | D)$  есть вероятность, отражающая всевозможные доверия, связанные с предположением  $A = A_i$  для всех полученных на данный момент свидетельств  $D$ . Распределение доверия каждой переменной  $\tilde{x}$  может быть получено если доступны три параметра:

$\pi$  - текущее значение силы «причинной поддержки»  $\pi_x(\tilde{u}) = p(\tilde{u} | E_1)$ , где  $E_1$  - множество всевозможных свидетельств всех предков узла  $\tilde{x}$ . Этот параметр показывает доверие к тому факту, что данная причина свершилась при данных вероятностях ее предков.

$\lambda$  - текущее значение силы «диагностической поддержки»  $\lambda_z = P(E_2 | \tilde{x})$ , где  $E_2$  - множество всевозможных свидетельств всевозможных потомков узла  $\tilde{x}$ . Этот параметр показывает доверие к тому факту, что свершились потомки данной причины при ее конкретной вероятности.

$p(\tilde{x} | \tilde{u}\tilde{v})$  - фиксированный тензор условных вероятностей, который соотносит переменную  $\tilde{x}$  с ее непосредственными родителями.

Алгоритм обновления вероятностей состоит в следующем:

Шаг 1. Обновление доверия. При активизации узла  $\tilde{x}$  для обновления параметров, то сразу же:

проверяется  $\pi_x(\tilde{u})$ ,  $\pi_x(\tilde{v})$  его родителей,

проверяется  $\lambda_y(\tilde{x})$ ,  $\lambda_z(\tilde{x})$  его потомков.

Тогда мера доверия узла  $X$  вычисляется так:

$$Bel[\tilde{x}] = \alpha \lambda_y(\tilde{x}) \lambda_z(\tilde{x}) \sum_{\tilde{u}, \tilde{v}} (p(\tilde{x} | \tilde{u}, \tilde{v}) \pi_x(\tilde{u}) \pi_x(\tilde{v})),$$

где  $\alpha$  является константой нормализации и находится из условия:  $\sum_{\tilde{x}} Bel[\tilde{x}] = 1$ .

Шаг 2. Обновление  $\lambda$ . Используя полученное сообщение, каждый узел вычисляет сообщение  $\lambda$ , которое будет послано его родителям:

$$\lambda_x(\tilde{u}) = \alpha \sum_{\tilde{v}} \left( \pi_x(\tilde{v}) \sum_{\tilde{x}} \lambda_y(\tilde{x}) \lambda_z(\tilde{x}) p(\tilde{x} | \tilde{u}, \tilde{v}) \right).$$

Шаг 3. Обновление  $\pi$ . Каждый узел вычисляет новое сообщение  $\pi$ , которое будет послано его потомкам:

$$\pi_y(\tilde{x}) = \alpha \lambda(\tilde{x}) \sum_{\tilde{u}, \tilde{v}} p(\tilde{x} | \tilde{u}, \tilde{v}) \pi_x(\tilde{u}) \pi_x(\tilde{v}).$$

Текущий процесс передачи данных как начинается, так и заканчивается в конечных узлах сети, при этом:

ожидающий узел есть переменная с неприсвоенным значением. Для таких узлов  $x$  полагаем, что  $\lambda_{\text{потомок}}(\tilde{x}) = 0.5$ ;

свидетельствующий узел есть означенная переменная. В этом случае  $\lambda_{\text{потомок}}(\tilde{x}) = 1$ , если  $\tilde{x}$  совпадает со значением переменной и  $\lambda_{\text{потомок}}(\tilde{x}) = 0$  в противном случае;

корневой узел представляет переменную без предков. Для каждой такой переменной вводим фиктивного предка  $u$ , постоянно представляющего значение  $\tilde{u} = u$ , и множество условных вероятностей на связи  $u \rightarrow x$ , каждая из которых равна априорной:  $p(\tilde{x} | \tilde{u}) = p(\tilde{x})$ .

### 3.3.5. Представление информации об интенсивности взаимодействия узлов ВС сетью доверия

Для построения графа сети доверия необходима информация об интенсивностях взаимодействия компьютеров ВС между собой. Эта информация извлекается из дополненных DFD – диаграмм, описывающих автоматизируемые бизнес – процессы. В стандартные DFD – диаграммы были добавлены новые элементы.

Каждому пользователю ВС приписывается количество рабочих мест и выполняемые им задачи. Одним пользователем считается группа пользователей ВС, выполняющих одинаковые задачи.

Каждая задача представляется в виде последовательности элементов типа «Документ» или «Запрос», представляющих документы и запросы, передаваемые пользователем по сети. Различие между документами и запросами заключается в том, что документы имеют четкий числовой размер, а размер запросов является величиной нечеткой.

В разработанном прототипе САПР ВС механизм байесовских сетей доверия используется для оптимизации следующих параметров ВС: состав коммуникационного оборудования и пропускная способность каналов. Преимуществом применения БСД является отсутствие необходимости в моделировании многочисленных промежуточных вариантов сети, среди недостатков следует отметить необходимость построения графа сети доверия для каждой ВС.



Для оптимизации состава коммуникационного оборудования используется БСД, пример структуры которой приведен на рисунке 3.8:

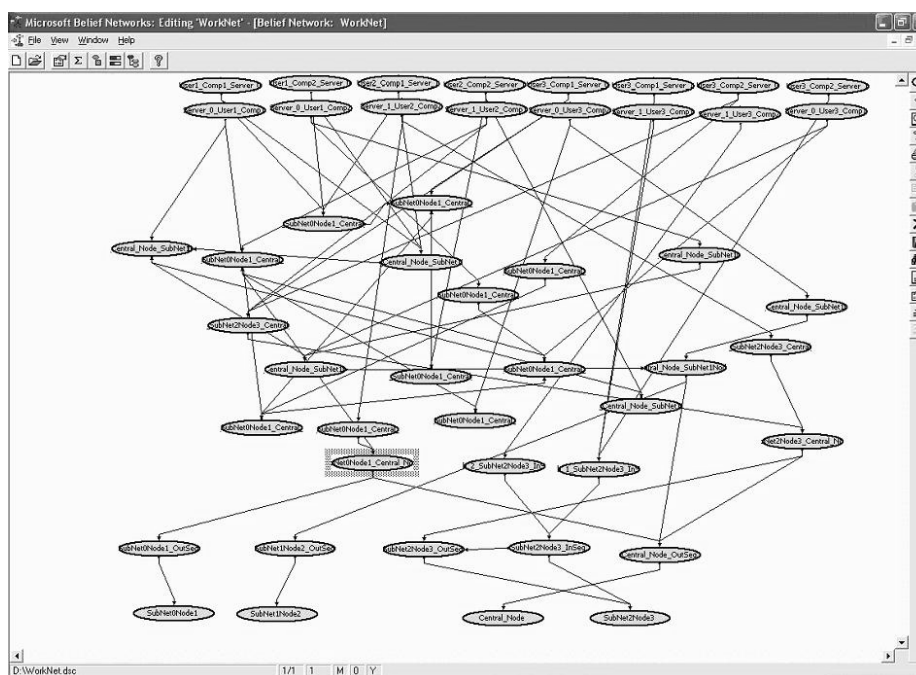


Рис. 3.8. Пример БСД для оптимизации коммуникационного оборудования ВС

БСД для решения задачи оптимизации коммуникационного оборудования состоит из четырех слоев: слой взаимодействий, слой интенсивности трафика, слой внутрисегментных и межсегментных трафиков, слой определения состава оборудования. Рассмотрим функции каждого слоя БСД.

**Слой 1 – слой взаимодействий.** Данный слой вводит в БСД начальную информацию об интенсивностях взаимодействия компьютеров ВС между собой. Эта информация определяется мастером перехода на основании информации об автоматизируемых бизнес – процессах, введенной в DFD – диаграммер. Величина интенсивности взаимодействия является нечеткой, так как передаваемый трафик состоит из документов, имеющих четкий размер, и запросов к базам данных, имеющих нечеткий размер.

**Слой 2 – слой интенсивности трафика.** На данном слое происходит группировка информации об интенсивностях взаимодействия компьютеров по каналам сети, через которые проходят взаимодействия. Основная трудность при построении данного слоя заключается в том, что у каждого узла имеется большое количество предков, что приводит к резкому возрастанию размера матрицы условных вероятностей (для трех предков размер матрицы равен 64 строки, для пяти предков – 1024 строки, для семи предков – 16 384 строки). Для преодоления данной трудности при построении графа БСД используется процедура группировки, принцип действия которой показан на рисунке 3.9:

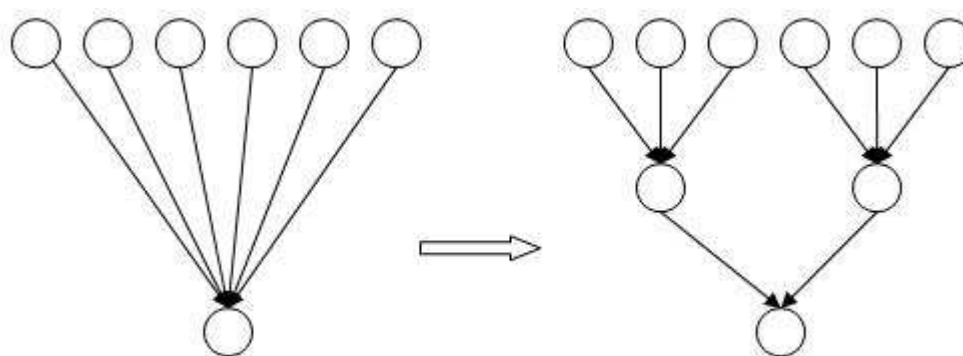


Рис. 3.9 Группировка узлов-предков в БСД

Основная идея работы процедуры заключается в введении новых узлов, на которые группируются узлы – предки, после чего введенные узлы становятся предками узлов второго слоя..

**Слой 3 – слой внутрисегментных и межсегментных трафиков.** На данном уровне определяется интенсивность внутрисегментных и межсегментных взаимодействий для каждого узла сети. Каждый узел сети образует сегмент. Для каждого узла сети в данном слое существуют два узла БСД – на одном формируется внутрисегментный трафик, на другом – межсегментный трафик. Трафик формируется по принципу протекания через сегмент или внутри него. Таким образом, у каждого узла сети следующего слоя оказывается минимум один предок.

**Слой 4 – слой определения состава оборудования.** Данный слой является конечным слоем, и в его узлах формируются вероятности нахождения в определенном сегменте определенного типа коммуникационного оборудования. Для каждого узла ВС в данном слое существует один узел БСД, имеющий как минимум одного предка. Матрицы условных вероятностей для этого слоя задаются экспертом.

Для второго и третьего слоя матрицы условных вероятностей представляют собой таблицы сложения нечетких величин трафика.

Таким образом, оптимизация состава коммуникационного оборудования при помощи БСД позволяет получить результат, обусловленный автоматизируемыми бизнес – процессами и мнением эксперта.

Для оптимизации пропускной способности каналов сети используется БСД, имеющая структуру, представленную на рисунке 3.10:

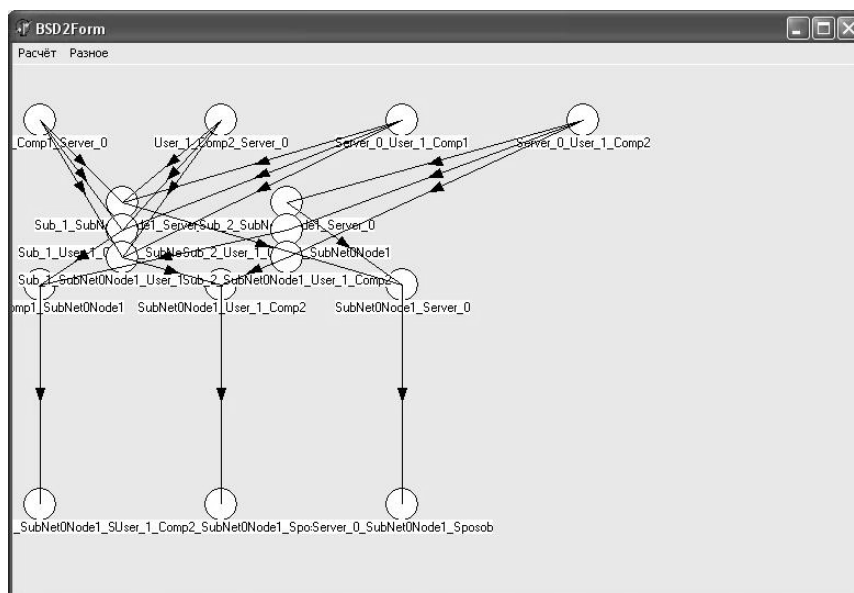


Рис. 3.10. БСД для оптимизации пропускной способности ВС

В данной сети доверия выделяются три слоя: слой задания интенсивностей, слой определения трафика, слой определения пропускной способности. Рассмотрим функции каждого из слоев БСД.

**Слой 1 – слой задания интенсивностей.** Данный слой вводит в БСД начальную информацию интенсивностях взаимодействия компьютеров ВС между собой. Эта информация определяется мастером перехода на основании информации об автоматизируемых бизнес – процессах, введенной в DFD – диаграммер.

**Слой 2 – слой определения трафика.** В данном слое происходит группировка интенсивностей взаимодействий по каналам, через которые протекают взаимодействия.

**Слой 3 – слой определения пропускной способности.** В данном слое определяется пропускная способность каналов сети, необходимая для того, чтобы пропустить трафик, определенный для канала в предыдущем слое. Каждый узел данного слоя имеет только одного предка, несущего информацию о передаваемом по каналу трафике. Матрица условных вероятностей для данного слоя задается экспертом, для второго слоя она представляет собой таблицу сложения нечетких величин интенсивностей.

Наряду с использованием байесовских сетей доверия в прототипе САПР ВС также используются генетические алгоритмы для решения задач оптимизации, для которых сложно сформулировать набор правил или представить граф сети доверия. Это задачи оптимизации подключения рабочих станций (сложно сформулировать набор правил) и задача оптимизации размещения коммуникационного оборудования (сложно представить граф сети доверия).

В разработанном прототипе САПР ВС основной единицей реализации БСД является узел БСД. Он реализован следующим образом:

```

TNode = Class(TObject)
public
  Id      : Integer;
  X       : Integer;
  Y       : Integer;
  number:integer;
  Name:string[50];//Наименование вершины
  Links:TList;//Список связей вершины
  Segment_num:integer;//Номер сегмента
  Segment1,Segment2:integer;
  Computer:TComputerImage;
  BSDNode:INode;
  constructor Create;
  destructor Free;
end;

```

Назначение свойств и методов данного класса приведено в таблице 3.3.

Таблица 3.3.

Назначение свойств и методов класса «узел БСД»

Свойство / метод	Назначение
Id, X, Y	Свойства предназначены для отображения узлов на форме работы с БСД. Id – уникальный номер узла, задаваемый автоматически, X, Y – координаты центра графического элемента
Number	Служебная переменная, используемая при построении структуры сетей доверия для оптимизации пропускной способности.
Name	Наименование переменной, представляемой узлом сети доверия (например, внутрисегментный трафик)
Links	Список связей узла сети со своими потомками
Segment_num, Segment1, Segment2	Служебные переменные, используемые при группировке трафика по каналам ВС.
BSDNode	Интерфейс к объекту Node библиотеки MSBNx. Этот объект хранит таблицы условных вероятностей узла и производит определение вероятностей для набора состояний узла.
Create	Конструктор объекта
Free	Деструктор объекта

Основная часть реализации БСД приходится на свойство узла BSDNode. Интерфейс к объекту библиотеки позволяет выполнять следующие действия: создание и удаление узла, заполнение таблицы условных вероятностей узла, за-

дание вероятностей состояний узла. Для заполнения матриц условных вероятностей используется компонент TRxMemoryData, представляющий собой таблицу данных, расположенную в оперативной памяти. В процессе заполнения матрицы в данном компоненте формируются условные вероятности, которые затем переносятся через интерфейс BSDNode в сеть доверия. Преимуществами такого подхода являются: простота формирования промежуточной таблицы и увеличение скорости работы с БСД за счет работы с оперативной памятью. Для проведения вывода с использованием БСД задаются вероятности начальных узлов сети. Задание вероятностей инициирует процесс распространения введенной информации по сети доверия, по завершении которого в конечных узлах сети формируются значения вероятностей, обоснованные структурой сети и информацией из матриц условных вероятностей.

В качестве тестового примера рассмотрим ВС, имеющую структуру, представленную на рисунке 3.11:

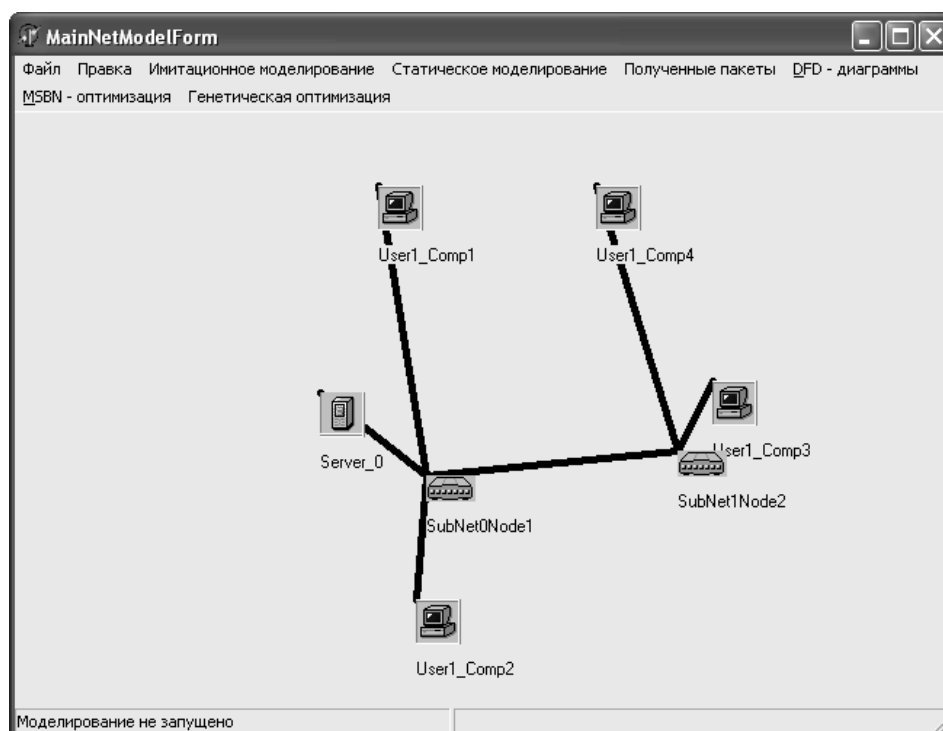


Рис. 3.11. Пример структуры ВС

В данной сети четыре рабочих станции сгруппированы в подсети по двое. Каждая рабочая станция передает в базу данных, расположенную на сервере Server\_0 документ размером 10 кБ и получает ответ размером 5 кБ. Коммутационные узлы представлены концентраторами, архитектурой сети является «общая шина».

Протокол моделирования выглядит следующим образом:

Канал User1\_Comp1 - SubNet0Node1 -

Среднее значение трафика: 60 кБ

Минимальное значение трафика: 59,9991 кБ  
Максимальное значение трафика: 60,0009 кБ

Канал User1\_Comp2 - SubNet0Node1 -

Среднее значение трафика: 60 кБ  
Минимальное значение трафика: 59,9991 кБ  
Максимальное значение трафика: 60,0009 кБ

Канал User1\_Comp3 - SubNet1Node2 -

Среднее значение трафика: 60 кБ  
Минимальное значение трафика: 59,9991 кБ  
Максимальное значение трафика: 60,0009 кБ

Канал User1\_Comp4 - SubNet1Node2 -

Среднее значение трафика: 60 кБ  
Минимальное значение трафика: 59,9991 кБ  
Максимальное значение трафика: 60,0009 кБ

Канал Server\_0 - SubNet0Node1 -

Среднее значение трафика: 60 кБ  
Минимальное значение трафика: 59,9991 кБ  
Максимальное значение трафика: 60,0009 кБ

Канал SubNet0Node1 - SubNet1Node2 -

Среднее значение трафика: 60 кБ  
Минимальное значение трафика: 59,9991 кБ  
Максимальное значение трафика: 60,0009 кБ

По результатам моделирования объем переданных данных на всех каналах сети одинаков и составляет 60 кБ. Протокол оптимизации коммуникационного оборудования с использованием БСД выглядит следующим образом:

Узел SubNet0Node1

Вероятности

Концентратор - 0,5

Коммутатор - 0,5

Маршрутизатор - 0

Узел SubNet1Node2

Вероятности

Концентратор - 0,5

Коммутатор - 0,5

Маршрутизатор – 0

Таким образом, из протокола видно, что БСД не считает нужным установку маршрутизатора в каком - либо из узлов. Вероятности установки концентратора и коммутатора равны, что означает предоставление проектировщику возможности выбора из нескольких (в данном случае четырех) вариантов, исходя из критериев, отличных от минимального трафика.

После проведенной оптимизации поставим в узлах сети коммутаторы и определим изменение трафика на каналах ВС. Оптимизированная сеть представлена на рисунке 3.12:

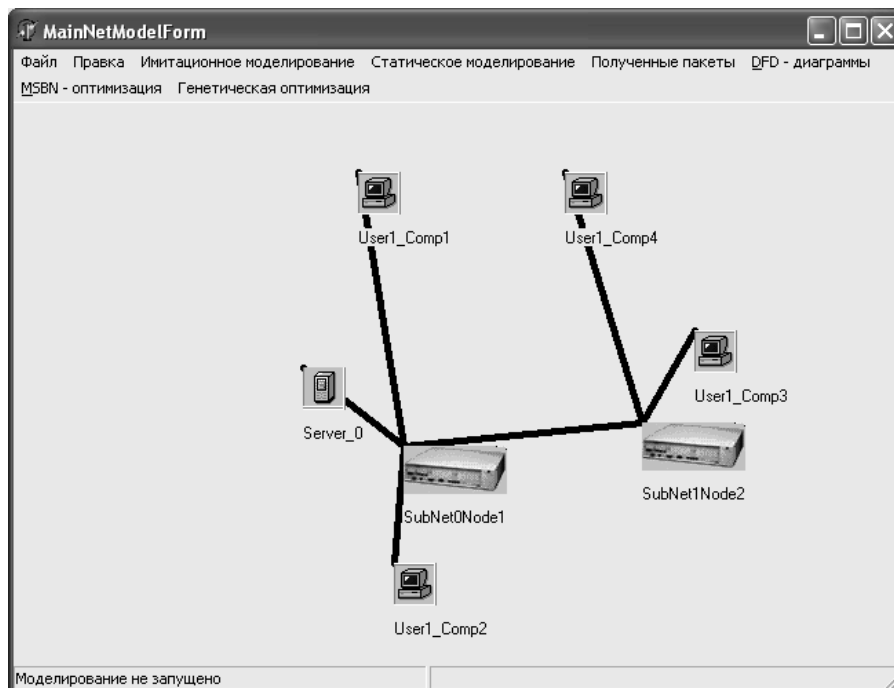


Рис. 3.12. Пример оптимизированной структуры ВС

Протокол моделирования оптимизированного варианта ВС выглядит следующим образом:

Канал User1\_Comp1 - SubNet0Node1 -

Средний объем данных: 15 кБ

Минимальный объем данных: 14,9997 кБ

Максимальный объем данных: 15,0003 кБ

Канал User1\_Comp2 - SubNet0Node1 -

Средний объем данных: 15 кБ

Минимальный объем данных: 14,9997 кБ

Максимальный объем данных: 15,0003 кБ

Канал User1\_Comp3 - SubNet1Node2 -

Средний объем данных: 15 кБ

Минимальный объем данных: 14,9997 кБ  
Максимальный объем данных: 15,0003 кБ

Канал User1\_Comp4 - SubNet1Node2 -  
Средний объем данных: 15 кБ  
Минимальный объем данных: 14,9997 кБ  
Максимальный объем данных: 15,0003 кБ

Канал Server\_0 - SubNet0Node1 -  
Средний объем данных: 60 кБ  
Минимальный объем данных: 59,9991 кБ  
Максимальный объем данных: 60,0009 кБ

Канал SubNet0Node1 - SubNet1Node2 -  
Средний объем данных: 30 кБ  
Минимальный объем данных: 29,9995 кБ  
Максимальный объем данных: 30,0005 кБ

Из протокола видно, что на каналах от рабочих станций объем передаваемых данных уменьшился в четыре раза, на канале между узлами сети – уменьшился в два раза, на канале от сервера данных – не изменился. Таким образом, оптимизация понизила трафик на каналах сети до четырех раз.

Результат оптимизации пропускной способности каналов оптимизированной ВС представлен ниже:

Канал User1\_Comp1 - SubNet0Node1

Вероятности

Малая - 1

Средняя - 0

Высокая - 0

Канал User1\_Comp2 - SubNet0Node1

Вероятности

Малая - 1

Средняя - 0

Высокая - 0

Канал User1\_Comp3 - SubNet1Node2

Вероятности

Малая - 1

Средняя - 0

Высокая - 0

Канал User1\_Comp4 - SubNet1Node2



Вероятности

Малая - 1

Средняя - 0

Высокая - 0

Канал Server\_0 - SubNet0Node1

Вероятности

Малая - 0,625078290742752

Средняя - 0,374921709257248

Высокая - 0

Канал SubNet0Node1 - SubNet1Node2

Вероятности

Малая - 0

Средняя - 1

Высокая - 0

Из протокола оптимизации пропускной способности видно, что для всех каналов ВС, кроме канала от сервера данных, достаточно малой пропускной способности. Для серверного канала можно применять канал средней пропускной способности, хотя трафик на нем пока недостаточно высок.

Для реализации механизма байесовских сетей доверия в прототипе САПР ВС использован модуль MSBNx фирмы Microsoft. Этот модуль представляет собой внутренний сервер автоматизации, реализованный в виде DLL - файла. Основные преимущества использования данного сервера заключаются в высокой скорости пересчета вероятностей, характеризуются удобным вариантом хранения структуры сети и обработкой ошибок на этапе распространения вероятностей. Главными недостатками являются: недоступность исходного кода, отсутствие возможности наблюдения за процессом распространения вероятностей, необходимость использования технологии COM для работы с сервером. Сервер предназначен для работы в среде Win32, что позволяет использовать прототип САПР ВС в операционных системах семейств Win9x и WinNT(NT, 2000, XP).

### **Библиографический список**

1. Тулупьев, А. Л. Алгебраические байесовские сети. Логико - вероятностный подход к моделированию баз данных с неопределенностью. Российская академия наук, Санкт - Петербургский институт информатики и автоматизации, Санкт - Петербург, 2000
2. Якубайтис, Э. А. Информационные сети и системы. - М.: Финансы и статистика, 1996.
3. Ярушкина Н. Г., Наместников А. М. Эффективность генетических алгоритмов для задач автоматизированного проектирования.

4. Сергей Гурин. Параллельное объектно–ориентированное программирование в среде Delphi. Документация к программной библиотеке Gala.

5. Eric Horvitz; David Hovel; Carl Kadie. MSR-TR-2001-67. MSBNx: A Component-Centric Toolkit for Modeling and Inference with Bayesian Networks July 2001.

6. Тейксейра Стив, Пачеко Ксавье. Delphi 5. Руководство разработчика.: Москва, Издательский дом «Вильямс», 2000.

## **Глава 4. МЯГКИЕ ВЫЧИСЛЕНИЯ В ТЕХНОЛОГИИ БАЗ ДАННЫХ**

### **4.1. Определение, возможности и ограничения нечеткого интеллектуального сервера данных**

#### **4.1.1. Возможности и перспективы нечеткого реляционного интеллектуального сервера данных**

Проектирование сложных технических изделий выполняется в наши дни распределенным коллективом проектировщиков, использующих информационные технологии и работающих в условиях развитой корпоративной сети. Средством согласования проектных решений служит репозиторий проекта, представляющий собой информационное хранилище (базу данных) всех проектных документов. В условиях, когда сроки разработки нового изделия определяют рыночный успех фирмы, архив старых проектных решений не может быть хранилищем микрофильмированных или тем более бумажных материалов. Как архив проектов, так и репозиторий проекта должны быть активными хранилищами данных. Международные стандарты (ISO-9000) требуют от предприятий иметь полное электронное представление изделий, которое должно строиться в ходе проектирования на базе репозитория текущего проекта. Построить такое информационное обеспечение автоматизированного проектирования можно только на основе современных серверов данных, работающих в рамках клиент-серверной технологии. Сложность использования систем управления базами данных (СУБД) для организации конструкторских баз данных (БД) состоит не только в ограничениях реляционной модели, но и в необходимости представлять неполностью определенные проектные решения. В процессе поискового творческого проектирования часть значимой информации обязательно будет неопределенной, неточной, нечеткой. Запрос, формируемый проектировщиком в таких условиях, должен быть гибким, содержащим нечеткие условия. Очень важным видом запроса, позволяющим сократить сроки разработки изделия, является запрос аналогичного проектного решения. Аналогия между объектами может быть выражена на основе нечетких отношений. Вместе с тем необходимо учитывать распространенность языка запросов SQL. Необходимо расширение современной реляционной модели, которое позволило бы учесть такие осо-

бенности автоматизированного проектирования изделий, как хранение и обработка неопределенной информации. При этом новые возможности необходимо предложить проектировщикам в форме привычного SQL-запроса.

В работах Д. Дюбуа и Г. Прада предлагается расширение реляционной модели данных на основе теории возможности. Предложено формировать результаты запросов к реляционной модели, содержащей возможностный атрибут на основе распределения возможности. Но в данных работах не решены многие вопросы организации среды хранения и обработки нечетких данных средствами SQL-языка, являющегося фактическим стандартом языка запросов. Большинство работ других авторов носит теоретический характер и иллюстрируются экспериментальными программами на языках типа ЛИСП или ПРОЛОГ. Разработка новой нечеткой реляционной модели данных и реализация ее для промышленного сервера данных, например, класса СУБД Oracle, имеет научно-практический смысл.

В работах отечественных ученых А. Н. Мелихова, Л. С. Бернштейна, А. В. Боженюка успешно решены задачи поиска проектного решения по аналогии, основанной на теории нечетких систем. Взаимодействие интеллектуальных нечетких систем и систем данных в составе интегрированной интеллектуальной САПР требует от СУБД организации хранения кортежей с нечеткими атрибутами. Предложенные перечисленными авторами методы применены для САПР машиностроения. В работах М.Р.Кагаловского, А.Р.Саймона и других предложены перспективные идеи разработки постреляционных баз данных, в том числе активных.

Для достижения поставленной цели необходимо решить ряд задач:

1. Исследовать особенности проектной информации. Провести сравнительный анализ результативности современных моделей неопределенности для задач организации информационного обеспечения САПР. Изучить ранее разработанные реляционные модели данных, учитывающие нечеткость атрибутов объектов, и сделать вывод о возможности их применения для организации баз данных проектирования.
2. Разработать модель, расширяющую реляционную модель с точки зрения возможности представления и обработки нечетких данных. Модель должна обладать свойством реализуемости в современных серверах данных.
3. Разработать алгоритмы выполнения операций модифицированной реляционной алгебры для случая, когда отношения и кванторы запроса являются нечеткими. Построить подмножество языка запросов SQL (Fuzzy SQL).
4. Разработать структурно-функциональное решение нечеткого реляционного сервера данных для автоматизированного проектирования.
5. Реализовать программную систему, выполняющую хранение неполной проектной информации и обработку гибких нечетких запросов.

6. Выполнить на основе нечеткого реляционного сервера данных конструкторскую базу данных, допускающую гибкие нечеткие запросы для конкретных проектных организаций. Оценить на основе внедрения результативность нечеткого реляционного сервера.

#### 4.1.2. Ограничения существующих нечетких реляционных моделей

Анализ показывает, что заложенные в теории нечетких множеств возможности представления и обработки субъективной информации (в том числе применимость положений теории к единичным объектам) делают ее наиболее привлекательной для моделирования рассуждений человека. Следовательно, теория нечетких систем может быть применена в таких интеллектуальных системах, как САПР и экспертные системы. В настоящее время реляционная модель данных остается доминирующей моделью данных. Но реляционные системы (СУБД второго поколения) были сфокусированы на приложениях, обрабатывающих бизнес-данные и, как указывают многие исследователи, не являются адекватными решениями для более широкого класса приложений. Системы автоматизации проектирования, системы CASE и гипертекстовые приложения нуждаются в технологии СУБД, обладающих дополнительными специализированными возможностями.

На сегодняшний день назрела потребность в СУБД нового, третьего поколения. Современные САПР требуют решения новых сложных исследовательских задач: поддержки мультимедийных объектов, распределенного хранения информации, новых видов приложений баз данных, управления транзакциями и потоками работ, обеспечения простоты управления базами данных и их использования, перехода от баз данных к базам знаний. Тенденцию интеллектуализации баз данных отражает и задача представления и обработки нечеткой информации.

Анализ научно-исследовательских работ, связанных с представлением и обработкой нечеткой информации в базах данных, показывает, что, несмотря на множество разработок, исследования по применению нечетких множеств в СУБД пока не выросли в отдельное направление. Такое положение связано с тем, что сами СУБД пока не готовы к представлению и обработке таких сложных объектов, какими являются нечеткие множества, функции принадлежности и лингвистические переменные, хотя такая потребность становится все актуальнее. Широкое применение СУБД в САПР начинает определять направление развития самих СУБД. Следовательно, создаваемая в рамках данной работы модель должна быть технически реализуема, и эта реализация должна ориентироваться на современные промышленные решения.

#### 4.1.3. Определение нечеткого реляционного сервера данных

Нечеткая реляционная модель данных находится на стыке двух математических теорий: теории нечетких множеств и реляционной алгебры. До недавнего времени эти направления развивались независимо друг от друга и в каждой области сложились свои способы описания данных и операций над ними.

Пусть схемой отношения  $R$  называется конечное множество имен атрибутов  $\{A_1, A_2, \dots, A_n\}$ . Каждому имени атрибута  $A_i$  ставится множество  $D_i$ , называемое доменом атрибута  $A_i$ . Домены являются произвольными непустыми конечными или счетными множествами. И пусть  $D = \langle D_1, D_2, \dots, D_n \rangle$ .

**Определение 1.** Домен атрибута реляционного отношения будем называть нечетким, если для него определены:

- имя атрибута  $A_i$ ;
- универсальное множество  $X$ ;
- терминальное множество значений  $T$ , представляющих собой нечеткие метки.

**Определение 2.** Нечетким отношением будем называть конечное множество отображений  $\{t_1, t_2, \dots, t_p\}$  из  $R$  в  $D$ , если хотя бы одно  $t_i \in D_i$  и  $D_i$  – нечеткий домен.

Модель рассчитана на представление нечетких чисел, следовательно, доменом атрибута нечеткого числа является множество действительных чисел. Нечеткое число определяется на основе:

- функции принадлежности;
- лингвистической оценки.

Под лингвистической оценкой будем понимать одно из возможных значений лингвистической переменной, которое определяется соответствующим термом.

Функция принадлежности, задающая нечеткое число, удовлетворяет следующим свойствам:

- ограниченности (по определению  $\max(\mu(x)) \leq 1$ ,  $\min(\mu(x)) \geq 0$ );
- однозначности (каждое множество  $\mu(x) = \{z\}$ ,  $x \in X$ , состоит только из одного элемента);
- непрерывности (в каждой точке существует предел функции).

Если для некоторого аргумента значение функции принадлежности неопределенно, то предполагается, что это значение равно нулю.

Для представления точных (четких) значений используется вырожденный вид функции принадлежности, возвращающей 1 для представляемого точного значения и 0 для всех остальных значений.

В модели принят табличный способ задания функции принадлежности. Количество пар  $\mu(x_i)/x_i$ , задающих функцию принадлежности, неограниченно. Предполагается, что множество таких пар задает ломаную линию, которая и

является графиком функции принадлежности. Такой подход хорошо согласуется с представлением данных в реляционной модели. В разработанной модели функция принадлежности в общем случае задается субъективно в зависимости от решаемой задачи. Никаких других ограничений на вид функции принадлежности не накладывается.

В разработанной модели, кроме представления нечетких данных, заложены и механизмы их обработки:

- функции одного аргумента (дефазификация, максимализация, нормализация, альфа-срез);
- множественные операции (дополнение, пересечение, объединение);
- арифметические операции (сложение, вычитание, умножение, деление);
- операции сравнения.

Множественные операции реализованы с использованием функций минимума и максимума, так как предполагается субъективный характер функций принадлежности. Арифметические операции основываются на принципе обобщения Заде:

$$C = A \nabla B = [\sup \min(\mu_A(a), \mu_B(b))] / [a \nabla b], a \in S_A, b \in S_B,$$

где  $S_A, S_B, \mu_A, \mu_B$  – соответственно носители и функции принадлежности нечетких чисел  $A$  и  $B$ ,  $\nabla$  – одна из четырех арифметических операций.

Операция сравнения является основой в механизме обработки нечетких данных, так как на ней базируются определения реляционных операторов.

**Утверждение 1.** Операция сравнения для нечетких чисел дает нечеткий результат: определенный уровень уверенности, с которым можно утверждать о равенстве (неравенстве) нечетких чисел.

**Утверждение 2.** Пересечение нечетких чисел может являться критерием при их сравнении. Степень уверенности определяется посредством дефазификации методом среднего максимума полученного пересечения. Пустое множество равнозначно нулевой уверенности.

Пусть  $A$  и  $B$  два нечетких числа, тогда степень их равенства определим с помощью следующего индекса ранжирования:

$$E(A, B) = \max(\min(\mu_A(x), \mu_B(x))),$$

где  $x \in R$  ( $R$  – множество действительных чисел).

Выделяют два класса реляционных операторов. К первому относятся операторы, в основе которых лежат операции сравнения (выбор, объединение), ко второму – соответственно те, в которых операции сравнения не используются (булевы операторы, проекция, переименование). Второй класс операторов с точки зрения рассматриваемой модели интереса не представляет, так как эти операторы будут выполняться для нечетких данных так же, как и для традиционных точных. Особенностью выполнения операторов первого класса над нечеткими данными является тот факт, что в общем случае все кортежи отношения с разной степенью уверенности удовлетворяют условию (условиям), составляющих основу этих операторов.

Пусть  $r(R)$  – нечеткое отношение,  $A$  – нечеткий атрибут в  $R$  и  $a \in \text{dom}(A)$ . Из определения нечеткого атрибута следует, что  $\text{dom}(A) = D$  (множество действительных чисел). Тогда операцию выбора («выбрать из  $r$  кортежи, в которых значение  $A$  равно  $a$ ») для нечетких чисел можно формально записать:

$$\sigma_{A=a}(r) = r' = \{ t \in r \mid t(A)=a \} = \{ t \in r \mid E(t(A), a) \leq \theta_a \},$$

где  $E$  – введенный нами индекс ранжирования на основе пересечения нечетких множеств,  $\theta_a$  – уровень (порог) уверенности, минимально достаточный для утверждения о равенстве двух нечетких чисел (обычно равный 0.5).

Соединение определяется как бинарный оператор для комбинирования двух отношений. Пусть  $r(R)$ ,  $s(S)$  и  $RS=T$ . Тогда соединение  $q(T)$  будет содержать кортежи  $t_r \in r$  и  $t_s \in s$  с  $t_r=t(R)$  и  $t_s=t(S)$ , такие что  $t_r(R \cap S)=t_s(R \cap S)$ , то есть каждый кортеж в  $q$  является комбинацией кортежа из  $r$  и кортежа из  $s$  с равными  $(R \cap S)$ -значениями. Соответственно для нечетких атрибутов получаем:  $E(t_r(R \cap S), t_s(R \cap S)) \leq \theta_a$ , где  $E$  – введенный нами индекс ранжирования на основе пересечения нечетких множеств,  $\theta_a$  – уровень уверенности, минимально достаточный для утверждения о равенстве двух нечетких чисел (обычно равный 0.5). Формально операцию соединения можно представить следующим образом:

$$r \bowtie s = q(T) = \{ t = t_r t_s \mid t_r \in r \text{ и } t_s \in s \text{ с } t_r=t(R) \text{ и } t_s=t(S), t_r(R \cap S)=t_s(R \cap S) \} = \{ t = t_r t_s \mid t_r \in r \text{ и } t_s \in s \text{ с } t_r=t(R) \text{ и } t_s=t(S), E(t_r(R \cap S), t_s(R \cap S)) \leq \theta_a \}.$$

Так как  $(R \cap S)$ -значения будут включены в результирующее отношение, то возникает вопрос, какое именно значение:  $t_r(R \cap S)$  или  $t_s(R \cap S)$  должно быть включено. В случае нечетких атрибутов эти значения не тождественны, а равны с некоторой степенью уверенности.

**Предложение 1.** При соединении двух отношений в качестве значений на общих нечетких атрибутах целесообразно использовать объединение значений на соответствующих атрибутах исходных отношений.

Важное практическое значение в реляционной модели имеют понятия функциональной зависимости и нормальных форм.

**Утверждение 3.** Вхождение нечеткого атрибута в подмножество, образующее ключ, недопустимо.

Данное утверждение основано на том, что два нечетких числа могут быть равны с некоторым уровнем уверенности, который может быть задан субъективно, и поэтому в одном случае этого уровня может быть достаточно для утверждения, что эти числа равны, а в другом – нет. А если нельзя однозначно утверждать равенство значений, то для кортежей  $t_1$  и  $t_2$  нельзя говорить о выполнении свойства  $t_1(K) \equiv t_2(K)$ , где  $K$  – ключ отношения  $r(R)$ ,  $K \in R$ . Поэтому понятие нечеткой функциональной зависимости приобретает другую форму.

Реляционная алгебра порождает реляционное исчисление или систему запросов. Запрос – это операция над отношениями, результатом которой также является отношение. Система запросов – это формальная система для выражения запросов, образующая базисную структуру языков запросов, то есть спе-



специализированных языков программирования, используемых в системах баз данных для формулировки команд.

#### 4.2. Нечеткий интеллектуальный реляционный сервер данных

Рассмотрим применения нечеткой модели в САПР для задачи поиска аналогов. Процесс поиска изделий, которые аналогичны проектируемым, является одним из первых этапов функционирования САПР. Обычно описания (конструктивные характеристики) изделий хранятся в базе данных САПР.

Предлагается использовать в качестве степени аналогичности нечеткое сравнение:

$$p \sim q = E(x_i^{(p)}, x_i^{(q)}),$$

где  $E$  – операция нечеткого сравнения,  $p$  и  $q$  – рассматриваемые изделия с заданными  $i$ -ми параметрами  $x_i^{(p)}$  и  $x_i^{(q)}$  соответственно.

Базовой операцией расширенной реляционной алгебры, учитывающей нечеткость данных, является операция сравнения нечетких чисел. Критерием равенства нечетких чисел может являться их пересечение. Расширение реляционной модели приводит к изменению ее свойств; при этом отмечается, что вхождение нечеткого атрибута в подмножество, образующее ключ, недопустимо.

Практическая реализация системы представления хранения и обработки нечетких данных в СУБД (FuzzyData Manager) основывается на существующей промышленной базе данных Oracle8i. Основное внимание уделяется описанию схемы данных, позволяющей представить в реляционной базе данных нечеткие данные, а также механизмам и алгоритмам обработки этих данных. Кроме этого, обоснован выбор инструментальных средств и рассмотрены направления дальнейшего развития системы.

В качестве инструментального средства использовались процедурные расширения сервера баз данных Oracle8i. Так как база данных под управлением Oracle8i не имеет средств для хранения нечеткой информации, то был спроектирован набор служебных таблиц – репозиторий, который предназначен для хранения функций принадлежности, лингвистических оценок и т. д. Репозиторий является как бы логическим дополнением словаря данных, в котором хранятся описания объектов базы данных (таблиц, представлений, индексов и т. д.). Все механизмы обработки нечетких данных собраны в четыре пакета. Пакет – это объект базы данных, в котором собраны логически связанные типы, объекты и подпрограммы PL/SQL (процедурное расширение языка SQL). Разработанные пакеты содержат набор процедур и функций для работы с нечеткими отношениями и репозитарием (рисунок 4.1).

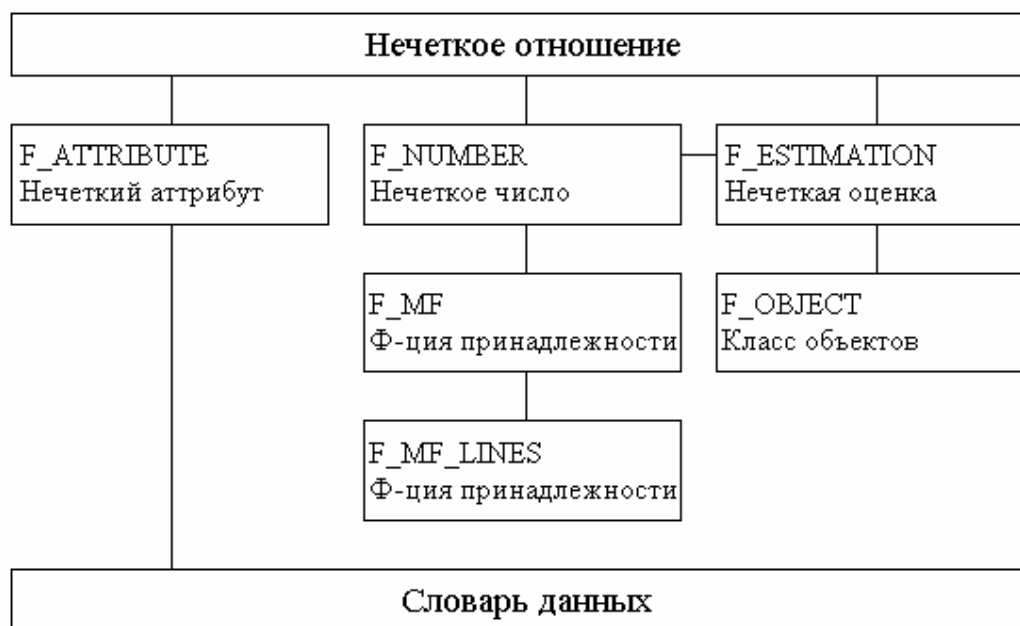


Рис. 4.1. Схема данных.

Нечеткое отношение базируется на обычной таблице базы данных, у которой нечеткий атрибут представлен в виде символьной строки. Все добавления, модификации и удаления данных из нечеткого отношения приводят к срабатыванию триггера, который преобразовывает данные во внутренний формат, заносит (или удаляет) их в репозиторий. Во внутреннем формате нечеткие числа имеют уже числовую форму. Все функции обработки данных работают уже с внутренним форматом и репозитарием, а возвращают результат в символьном виде, доступном для представления в таблице. Пользователю доступно как само нечеткое отношение, так и процедуры обработки (рисунок 4.2).

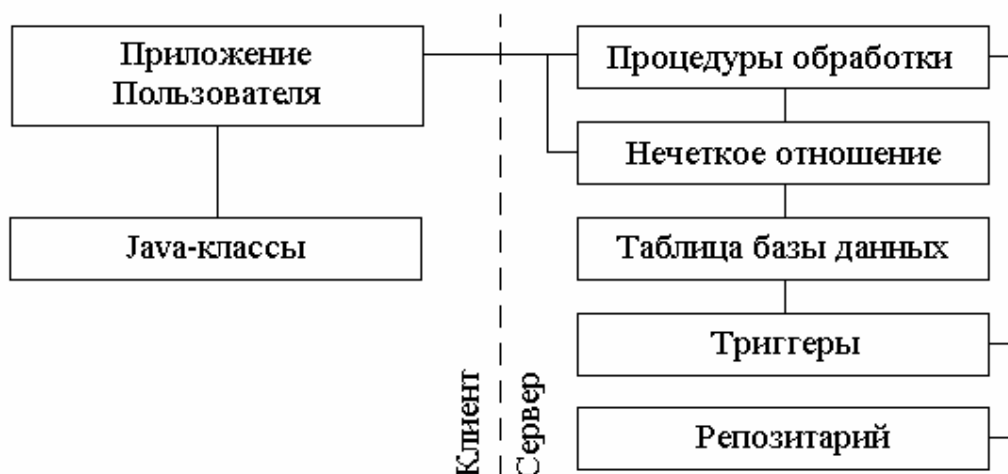


Рис. 4. 2. Компоненты пакета FuzzyData Manager.

Доступ к процедурам и функциям пакета FuzzyData Manager осуществляется стандартными средствами СУБД Oracle8i, вследствие чего возможен доступ как к локальной, так и удаленной БД.

Несмотря на самодостаточность серверной части и стандартность методов вызова необходимых процедур и функций, в рамках разрабатываемой системы была разработана клиентская часть. Эта часть была написана на языке Java и представляет собой набор классов. Клиентская часть также обладает средствами хранения и обработки нечетких данных. Это позволяет, во-первых, распределять нагрузку между клиентом и сервером, а, во-вторых, значительно упрощает написание конечных приложений.

Реализация классов имеет много общего с разработанными пакетами базы данных: внутреннее представление функций принадлежности аналогично представлению в базе данных; методы классов реализуют те же функциональные возможности, что и процедуры и функции пакетов. Клиентская часть реализована таким образом, что выходные данные процедур СУБД могут быть использованы в конструкторах объектов на клиенте, а объекты могут преобразовывать нечеткие данные в вид, доступный для хранения и обработки в базе данных.

Рассмотрим решения задач автоматизированного проектирования, в которых была использована нечеткая СУБД с расширенной реляционной моделью, учитывающей нечеткость данных. Основная особенность этих решений состоит в возможности представления и обработки нечеткой информации о проектируемом изделии или его компонентах на всех этапах автоматизированного проектирования.

Первая задача – это задача поиска оптимальных типоразмеров корпусов. Для ее решения использовалась интеллектуальная система поддержки принятия решений (ИСППР). Данная система состоит из трех традиционных компонент: СУБД, подсистема логического вывода (ПЛВ) и подсистема многокритериального анализа (ПМА). Отличительной особенностью рассматриваемой ИСППР является возможность обработки не полностью определенной информации. Для этого в основу ПЛВ положено использование механизма вывода на базе нечетких продукций.

Использование механизма вывода на базе нечетких продукций приводит к потребности в хранении в БД нечетких данных. В рамках данной работы была произведена интеграция разработанной нечеткой СУБД с ИСППР. Это позволяет наряду с четкими техническими характеристиками элементов хранить нечеткие данные и экспертные предпочтения, а также осуществлять нечеткий поиск на различных этапах обработки данных.

В базе данных ИСППР была создана схема и занесена информация об элементной базе (габариты корпусов и радиоэлементов) и ряде готовых проектов (состав СТО); проведено ранжирование доменов атрибутов, содержащих характеристики элементов и корпусов, путем задания на них лингвистических оценок.

Взаимодействие с БД может происходить в двух режимах: интерактивном и автоматическом. Автоматический режим предполагает работу всех компонентов системы, и в этом случае с БД взаимодействует ПЛВ. Так как разработка ПЛВ не входит в данную работу, то этот режим взаимодействия не рассматривается. Интерактивный режим взаимодействия с БД позволяет пользователям получать непосредственный доступ к данным с помощью стандартных средств.

В разработанной системе выбор необходимых элементов осуществляется с помощью команд языка SQL. Решение задачи поиска оптимального типоразмера корпусов сводится к автоматической обработке запросов, подобных нижеприведенному запросу.

```
-- Выбрать фонарь с большим диаметром.  
--  
select e.name, e.d, fuzzy.to_lingvo( e.d, 'ДИАМЕТР',  
'with_m' ) lingvo  
  from element e, etype t  
 where e.etype=t.id and t.name='Фонарь' and  
        fuzzy_math.equal( 'ДИАМЕТР.БОЛЬШОЙ', e.d )>0.75;
```

Вторая задача – реализация проекта автоматизации производства на основе создания электронного представления изделия (блоков изделий). Техническим ядром реализации проекта является разработка базы данных «Состав изделия» на основе СУБД Oracle8i.

Кроме непосредственного создания БД, предполагалось включение в данную БД механизмов поиска аналогов на основе нечеткого поиска. Анализ существующих изделий и поиск аналогов, которые могут быть взяты за основу нового проекта, играют важную роль на начальных этапах проектирования.

Основной особенностью БД «Состав изделия» является тот факт, что схема данных может изменяться во времени. Это связано с тем, что в процессе создания новых изделий постоянно расширяется набор параметров, описывающих эти изделия, увеличивается количество различных типов компонентов и так далее. В этой связи было предложено решение в виде схемы данных, не содержащей таблиц, описывающих конкретные изделия или компоненты. Все данные хранятся в нескольких универсальных таблицах. Одна таблица содержит все объекты (компоненты, изделия), вторая – все возможные параметры, третья – значения всех параметров для всех компонентов, четвертая содержит непосредственно состав каждого изделия.

Предложенная схема БД «Состав изделия» позволяет:

- создавать неограниченные по сложности зависимости,
- представлять точные и нечеткие атрибуты,
- создавать для каждого объекта свой собственный набор параметров,
- модифицировать набор параметров объекта, не изменяя структуру таблиц и синтаксиса поисковых запросов.

При реализации механизмов поиска аналогов учитывался тот факт, что аналогичность двух изделий можно рассматривать с точки зрения их свойств (параметров) или с точки зрения состава (структуры) этих изделий.

В общем виде алгоритм поиска аналогов выглядит следующим образом:

1. Проектируемое изделие описывается набором параметров и определяется его приблизительный состав.
2. Информация о новом изделии заносится в базу данных. Значения параметров и количество используемых компонентов может быть задано нечетко.
3. Затем анализируются все изделия (или определенный класс) в БД архива проектов.
4. Если поиск аналогов осуществляется по параметрам, то:
  - 4.1. Для каждого изделия из архива определяются значения параметров, по которым осуществляется поиск.
  - 4.2. Значение каждого параметра сравнивается с предполагаемым значением соответствующего параметра нового.
  - 4.3. Минимальная степень равенства, полученная при сравнении всех параметров, и есть искомая степень аналогичности изделий.
5. Если поиск аналогов осуществляется по составу, то:
  - 5.1. Для каждого изделия из архива определяется, сколько компонентов нового изделия входит в его состав.
  - 5.2. Полученное количество по каждому типу компонента сравнивается с количеством данного компонента в новом изделии.
  - 5.3. Минимальная степень равенства, полученная при сравнении количества всех типов компонент, и есть искомая степень аналогичности изделий.
6. Изделия с максимальной степенью аналогичности есть искомый результат.

Рассмотрим пример. При выборе типоразмера корпуса большое значение имеет количество элементов, выводимых на переднюю панель. Эксперт своими оценками, выраженными в лингвистической форме, определяет возможный «разброс» в значениях данной характеристики, а включенные в СУБД средства обработки нечеткой информации позволяют автоматически извлекать элементы, удовлетворяющие этой оценке с заданной степенью уверенности. Так, например, в разработанной системе найти в архиве проектов стенд с определенным количеством элементов на передней панели можно с помощью следующих команд:

```
-- Выбрать изделия с небольшим количеством элементов на
-- передней панели
--
create view o_by_q as
    select id_parent, sum(q) as sq
```

```

        from a_depends
        group by id_parent;
select id_obj, name
    from a_object, o_by_q
    where id_obj=id_parent and name='П.Панель' and
        fuzzy_math.equal( 'КОЛИЧЕСТВО.НЕМНОГО', sq
    )>0.6;

```

В результате получим множество изделий - аналогов по одному отдельно взятому конструктивному параметру. Также можно выявить аналоги по другим параметрам:

```

-- Выбрать изделия с большой массой корпуса
--
select id_obj, name
    from a_object, a_depends
    where id_obj=id_parent and id_child=(
        select o.id_obj, o.name as name,
            from a_object o, a_param p, a_value v
            where o.id_obj=v.id_obj and
p.id_param=v.id_param and
            o.name='Корпус' and p.name='Масса' and
            fuzzy_math.equal( 'МАССА.БОЛЬШАЯ',
v.value)>0.5);

```

В результате этих двух запросов получаем два множества, каждое из которых содержит аналоги по одному отдельно взятому конструктивному параметру. Пересечение этих множеств можно получить с помощью стандартных средств языка SQL, что и будет искомым множеством аналогов.

Так как рассмотренные задачи, при решении которых была использована нечеткая СУБД, содержали недостаточный объем данных для анализа эффективности выполнения нечетких запросов, то был проведен отдельный эксперимент, позволяющий провести такой анализ. Эксперимент состоял в нахождении по определенному критерию ряда элементов из большого массива данных. При выборе элементов измерялось затраченное на это время и количество выбранных элементов.

Во время эксперимента была создана таблица, содержащая 90 000 записей. Каждая запись содержала идентификатор - целое уникальное число в диапазоне от 1 до 90 000, название параметра – набор символов переменной длины от 1 до 26 и значение параметра - целое число в диапазоне от -21 475 до 21 474. Во время эксперимента из полученной таблицы выбирались элементы, у которых значение параметра равнялось приблизительно 100. Для этого была создана соответствующая лингвистическая оценка. Затем из таблицы удалялось 10

000 элементов и эксперимент повторялся. Были получены следующие результаты (табл. 4.1):

Таблица 4.1.

Результаты эксперимента анализа эффективности.

Количество строк, всего	Время обработки, сек	Количество строк, в тест-группе	Количество строк, выбранных нечетким запросом		
			$\mu=0.50$	$\mu=0.75$	$\mu=1.00$
10 000	17	5	4	3	2
20 000	34	14	10	7	5
30 000	50	25	18	13	9
40 000	67	36	26	20	14
50 000	85	44	31	24	17
60 000	100	47	32	25	18
70 000	116	52	36	27	18
80 000	135	59	41	30	20
90 000	151	65	45	32	22

Вычислительный эксперимент проведен на компьютере HP9000 K360. Если сравнивать количество элементов, получаемых в результате нечеткого запроса, с реальным количеством элементов на аналогичном диапазоне, то нетрудно посчитать, что при уровне уверенности равному 0.50 мы получаем 71% релевантных элементов, при 0.75 – 53% и при 1.00 – 37%. Таким образом, выбираются элементы, наиболее удовлетворяющие заданному критерию, и, кроме этого, они могут быть ранжированы по степени соответствия этому критерию.

Анализ временных показателей показывает, что полученные новые функциональные возможности требуют ощутимых вычислительных затрат. Из этого сделан вывод, что разработанная система может быть использована при проектировании объектов средней сложности, содержащих от  $10^2$  до  $10^3$  составных частей.

### 4.3. Система анализа нечетких тенденций

#### 4.3.1. Проблема Data Mining

В настоящее время накоплены огромные объемы информации, описывающие различные сложные системы. Анализ этих данных, содержащих свойства объектов, позволяет описать реальность и построить модель для дальнейшего изучения и прогнозирования поведения системы. Результатом анализа является нахождение взаимосвязей, динамики развития свойств объектов, т.е. знание о предметной области. Знание может выражаться функциональными зависимостями между свойствами, логическими связями. Исследования данных и их методов анализа оформились в виде отдельного направления называемого Data

Mining (DM). Современные комплексы DM позволяют «добывать» знания о системах в виде простых или сложных функции, множества продукций «если...то...» и представляют собой инструменты интеллектуального анализа данных. Каждый метод DM имеет свои достоинства и недостатки, что обусловлено характером данных и их взаимосвязей.

Один из видов извлекаемых знаний – это знания о динамике развития объекта. Существует статистическое решение задачи выделения тренда – постоянной составляющей временного ряда. Аналитическое выражение тренда ищется методами регрессионного анализа, сезонной декомпозиции, анализа Фурье, различными методами сглаживания. К их общим недостаткам можно отнести необходимость задания явной параметрической модели тренда и требования стационарности остатка.

Подход с точки зрения мягких вычислений позволит использовать для распознавания тенденций аппарат нейронных сетей. Исходные данные могут быть представлены в нечетком выражении, что может быть обусловлено экспертным оцениванием состояния системы, а так же несовершенными способами измерения параметров. Для описания изменений этих нечетких величин вводится понятие нечеткого ряда: упорядоченная последовательность нечетких меток, – другими словами, нечеткая зависимость от аргумента. Наиболее часто изменения параметра рассматриваются во времени, в этом случае говорится о нечетком временном ряде, где время есть свойство системы. Тенденция изменений так же выражается нечеткостью, являясь «нечеткостью второго порядка». Носителем нечеткого множества тенденций является множество всевозможных функций. В простейшем случае тенденция может описываться лингвистическими терминами «рост», «стабилизация», «падение», оценивающие знак производной функции. Но под нечеткой тенденцией можно понимать образы любых функции, не только «рост», «падение», «стабилизация», но и экспертные оценки тенденций: «колебания», «хаос». Таким образом, текущие состояние системы можно описать нечеткими тенденциями ее параметров.

#### 4.3.2. Задача обработки нечеткой тенденции

*Нечетким временным рядом (НВР)* будем называть упорядоченную последовательность наблюдений над некоторым явлением, характер которого изменяется во времени, если значения, которые принимает некоторая величина в момент времени, выражена с помощью нечеткой метки. Нечеткая метка представлена лингвистическим термом и функцией принадлежности.

*Тенденцией нечеткого временного ряда или нечеткой тенденцией (НТ)* будем называть нечеткую метку, выражающую динамику (систематическое движение) НВР. К значимым наименованиям тенденций относятся следующие нечеткие метки:

- рост;
- падение;



- стабилизация;
- колебания;
- хаос.

Разумеется, для нечетких термов, обозначающих тенденцию, возможно применение модификаторов “очень”, “более-менее” и т.д.

Нахождение НТ – основополагающая задача в построении системы анализа данных, решение которой позволит приступить к выявлению связей между параметрами исследуемого объекта. Данная задача представляет собой систему методов и инструментов, имеющая следующие входные и выходные параметры:

- вход: изменение параметра системы (во времени) за интервал (период);
- выход: вероятность типа тенденции.

Метод нахождения НТ строится на основе решаемых подзадач: описание НТ, определение инструмента распознавания, подготовка данных.

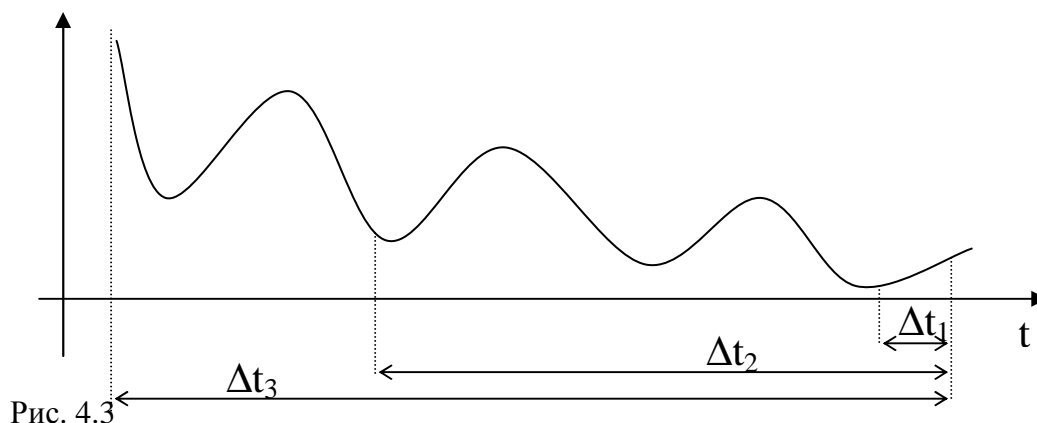
Эксперт, анализируя исходные четкие данные, должен определить, какие типы тенденций наблюдаются, описать их на основе спецификации выбранного метода. По выбору метода описания НТ подбирается подходящий инструмент определения НТ, и это в свою очередь накладывает требования на подготовку данных. Инструмент распознавания на основе описания НТ и преобразованных данных выдает тип тенденции.

Выделяются требования к системе распознавания НТ:

1. распознавание НТ на любом периоде (день, неделя, год);
2. универсальность метода определения тенденций независимо от ее типа.

НТ должна обнаруживаться независимо от выбранного периода, что позволит для каждого вида тенденции разбивать исходные данные на периоды наблюдения этой НТ. Например, для текущей точки (момента времени) определяются следующие тенденции: за период  $\Delta t_3$  - падение параметра, за  $\Delta t_2$  – периодические колебания, а за  $\Delta t_1$  – рост.

Универсальность метода описания и определения НТ позволит эксперту самостоятельно описывать НТ, четкий аналог которой невозможно описать в виде функций. Например, нечеткая тенденция «последовательные изменения параметра: два положительных и один отрицательный скачок» (рис.4.3).



#### 4.3.3. Метод анализа НТ

Реализованный метод определения НТ позволяет эксперту означивать виды тенденций и на основе алгоритмов нейронных сетей (НС) обучать систему распознавать их. Для обучения НС пользователем подготавливаются выборка данных, представляющая собой ряды нечетких меток и соответствующие им тенденции. Обученные НС (набор весов) представляют собой отдельные модули для каждого вида НТ, которые можно сохранять для дальнейшего использования. Алгоритм нахождения позволяет находить тенденции на любом отрезке данных, что дает возможность пользоваться модулями НС видов тенденций в независимости от их первоначального (обученного) периода, что реализуется процессом масштабирования входных данных.

#### 4.3.4. Реализация метода

НТ- множество нечетких меток, упорядоченных относительно некоторого параметра (далее время), исследуемой системы. Исходные данные преобразуются в множество, которое можно представить в виде плоскости в трехмерном пространстве  $(t-x-w)$ , где  $t$  – время,  $x$  – исследуемый параметр,  $w$  – вероятность (рис.4.4.).

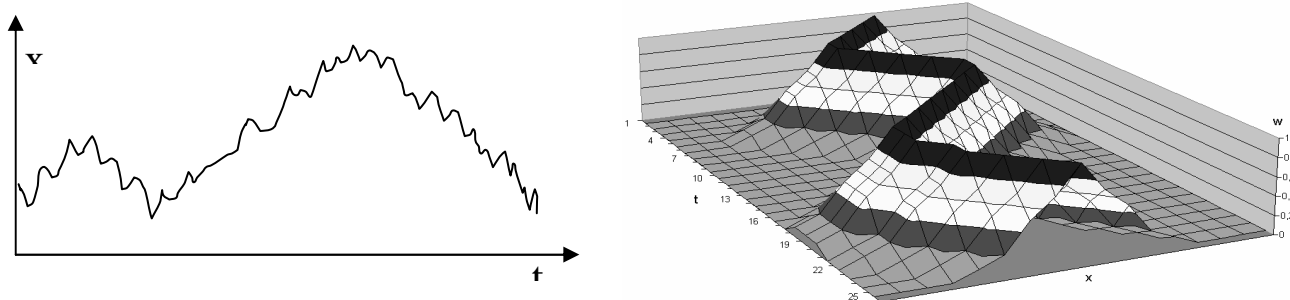


Рис.4.4. Нечеткий временной ряд

Срез по оси  $t$  – есть нечеткая метка со своей функцией принадлежности. Такое представление тенденции можно рассматривать как идеальное, т.к. на практике эксперт заранее определяет основные значения параметра (лингвистические термы) и может описывать изменения на их основе. Задача эксперта описать НТ как множество пар: время – терм. Время ограничено экспертом и задается периодом, т.е. всегда нужно указывать для какого интервала описывается НТ. Допустим, имеются следующие описания параметра: низкое, среднее, высокое. Тогда тенденция роста описывается в следующем виде:

РОСТ	Низкое	Среднее	Высокое
$T_1$	1	0	0
$T_2$	0	1	0
$T_3$	0	0	1

Описание НТ представляет собой следующие действия эксперта:

1. определение лингвистических термов;
2. определение периода НТ;
3. описание нечетких меток в каждом моменте периода.

В качестве инструмента определения НТ может выступать многослойная нейронная сеть, входом которой является множество нечетких меток, а выходом значения тенденций (0 или 1). Количество выходов может варьироваться в зависимости от выбранного варианта: одна нейронная сеть (НС) – один тип НТ, одна НС – на все типы. Задача эксперта не только описать тенденцию, а так же обучить НС.

Для обучения НС пользователем подготавливаются выборка данных, представляющая собой ряды нечетких меток и соответствующие им тенденции. Каждая нечеткая метка выражается вероятностью лингвистических термов: «низкое», «среднее», «высокое», -т.е. тройкой чисел от 0 до 1. Например, тенденцию «Период» можно представить следующим рядом чисел: «высокое» (0,0,1), «среднее» (0,1,0), «низкое» (1,0,0), «среднее» (0,1,0), «высокое» (0,0,1), «среднее» (0,1,0) и т.д. Следует отметить такую характеристику тенденции, как период ее протекания, выражающий глубину просмотра данных для текущей точки. Эксперт должен заранее определить, каков будет период НТ и исходя из этого строить обучающие ряды (количество меток в ряде есть период тенденции). Обученные НС (набор весов) представляют собой отдельные модули для каждого вида НТ.

Преобразование исходных данных, включающих точку нахождения НТ (текущий момент времени для временного ряда) и глубину просмотра (период), происходит следующим образом:

- фаззификация данных: преобразование четких данных в нечеткую метку на основе лингвистических термов, заданных на этапе обучения системы;

- определение коэффициента масштабирования: глубины периода, на котором обучили систему, и периода, определенного пользователем;
- свертка нечеткого ряда: нечеткие метки, найденные на основе коэффициента масштаба, преобразуются в метку путем нахождения средних вероятностей принадлежности к термам.

Подготовка данных заключается в фаззификации и получении нечетких меток, а так же в масштабировании (свертке) исходного периода до периода, на котором эксперт описал НТ. Для этого период разбивается на интервалы, соответствующие нечетким меткам описанного периода НТ. Вероятность термина определяется средним значением этого термина в нечетких метках интервала. Пример свертки интервала в одну нечеткую метку:

	T <sub>1</sub>	T <sub>2</sub>	T <sub>3</sub>		T
Низкий	0	0	0,1		0,03
Средний	0,5	0,4	0,9		0,6
Высокий	0,5	0,6	0		0,55

Т.о. данными действиями обеспечивается независимость выбора исходного периода распознавания и сглаживания данных, который может быть подвержен шуму, ошибками измерения и т.д.

В результате обработки временного ряда нейронной сетью во всех временных метках и на разных интервалах получится множество векторов оценок нечетких тенденций  $(t_i, \Delta t_j, w_{ij})$ , где  $t_i$  – момент времени определения НТ,  $\Delta t_j$  – временной интервал НТ,  $w_{ij}$  – оценка НТ. Данное множество, представленное в виде матрицы оценок, может служить источником для дальнейшего анализа выделенных тенденций.

#### 4.3.5. Оценка

Рассмотрим результат анализа нечетких тенденций временного ряда. В качестве объекта исследования взяты данные по доходам одного из подразделений Ульяновского государственного технического университета. Временной ряд представляет выборку недельных доходов в течение четырех лет.

Для анализа определены нечеткие тенденции: рост доходов, падение доходов, стабильность, периодические колебания. Обучение нейронных сетей проводилось на основе следующих описаний эталонных тенденций:

- лингвистические термы нечетких меток: низкий, средний, высокий;
- период обучающей тенденции 7.

В качестве размерности матрицы оценок выбраны следующие значения: временной ряд на моментах от 7 до 217, интервалы нечетких тенденций от 7 до 60, что соответствует поиску тенденций на интервалах от 1.5 месяца до 1.5 года. Рассмотрим результат анализа полученной матрицы оценок тенденции рос-

та. На рисунке 4.5 представлены исходные данные и матрица оценки нечеткой тенденции роста доходов.

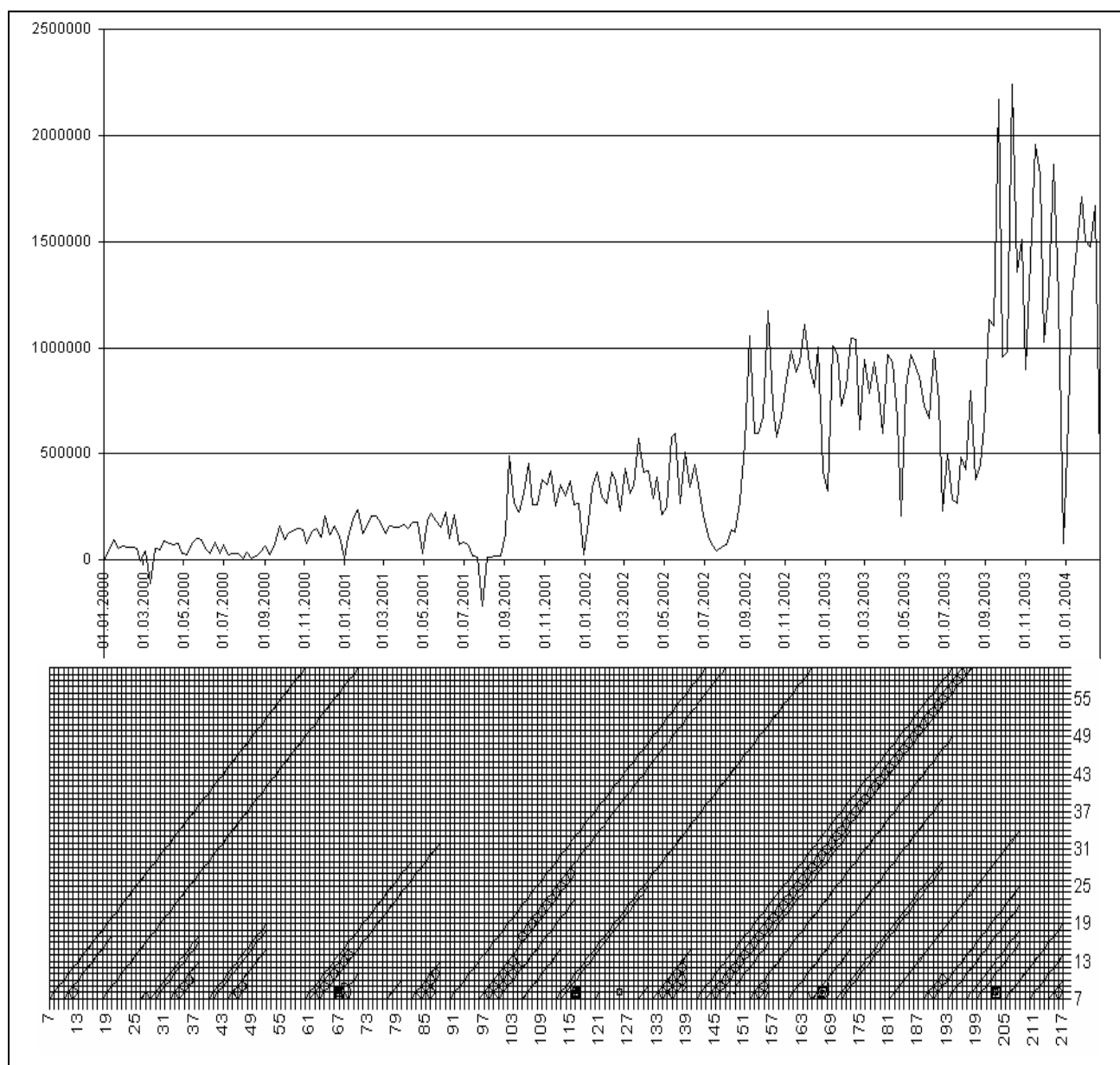


Рис. 4.5 Временной ряд и нечеткая тенденция роста доходов

Выявлены основные группы динамики роста: долгосрочный рост в интервале 9 месяцев - 1.5 года, и краткосрочный рост доходов - 2 месяца. Долгосрочная тенденция роста отмечается в течение следующих периодов:

- 2000 года;
- с января 2001г. по июнь 2001г.;
- с августа 2001г. по июнь 2002г.;
- с августа 2002 по июнь 2003г.

Краткосрочная тенденция роста отмечается в течении следующих периодов:

- январь-февраль каждого года;
- июнь-июль каждого года;
- сентябрь-октябрь каждого года.

Аналитически данные тенденции объясняются общим ростом цен на платные услуги, периоды оплаты обучающихся в начале каждого семестра.

## **Глава 5. МЯГКИЕ ВЫЧИСЛЕНИЯ В СОВРЕМЕННЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЯХ**

### **5.1. Основные задачи и мягкие технологии в реализации интеллектуального хранилища информационных ресурсов**

Проектная информация, хранимая в электронном виде, дает разработчикам возможность повторного использования своих и чужих разработок. Быстрый рост объемов такой информации требует особых средств для хранения и обработки информации и способствует развитию специального класса программных комплексов – *информационных хранилищ* (репозиториев).

Единицей обработки и хранения в таких комплексах является информационный ресурс. *Информационный ресурс* – это файл или совокупность файлов, объединенных общим смыслом и имеющих текстовую аннотацию. В частном случае, информационный ресурс – это один или несколько текстовых файлов. Текст аннотации (или текст самого ресурса) однозначно отражает смысловое содержание данного ресурса.

Основными функциями информационного хранилища являются:

- хранение информационных ресурсов;
- обеспечение доступа к ресурсам с рабочих мест;
- обеспечение возможности публикации новых ресурсов;
- возможность поиска интересующих ресурсов по определенным критериям.

Самым простым примером может являться использование корпоративного файл-сервера в виде хранилища различной электронной документации. В этом случае, хранение ресурса осуществляется на файловой системе сервера. Доступ с рабочих мест организован через доступ по ftp-, http- или smb-протоколам в локальной сети. Публикация производится путем загрузки файла/файлов на сервер по тем же протоколам. Поиск нужного ресурса происходит в виде обычного путешествия по каталогам файл-сервера. На первый взгляд, очень дешевое, удобное и простое решение. Но удобное и дешевое оно только до определенных размеров.

Приведенный выше набор функций информационного хранилища не учитывает несколько неочевидных фактов, которые всегда остаются в тени. У лю-

бого хранилища есть список лиц, ответственных за наполнение и работу репозитория. Это сотрудники организации, которые занимаются публикацией ресурсов. Их задача – наполнение репозитория новыми ресурсами и обеспечение многопользовательского доступа к ним. В нашем случае, это может быть отдел разработки программного обеспечения, наполняющий репозиторий своими результатами в виде исходных кодов. Основной вопрос в их работе: «Где опубликовать тот или иной ресурс?».

Также у хранилища есть пользователи, для которых самые востребованные функции репозитория – это доступ к ресурсам и их поиск. Их основной вопрос: «Где найти тот или иной ресурс?».

На определенной стадии развития хранилища наступает момент, когда решение этих вопросов начинает отнимать заметное время. Действительно, уже два десятка каталогов на файл-сервере заставляют задуматься, куда лучше поместить ресурс. Пользователь, в свою очередь, задумывается, где искать ресурс. Успешная реализация хранилища должна свести к минимуму время на решение этих вопросов.

## **5.2. Реализация интеллектуального хранилища**

### **5.2.1. Публикация ресурсов**

Рассмотрим проблему подробнее. По сути, структура каталогов отражает структуру категорий хранимой информации. А значит, задача размещения ресурса по каталогам есть задача классификации ресурса. При классификации ресурса должны быть решены следующие вопросы. Во-первых, должно быть составлено *дерево категорий*. В нашем случае это дерево каталогов на файл-сервере, составленное группой публикаторов. Во-вторых, требуется определить, какой (каким) из этих категорий, ресурс более всего соответствует. От решения этих вопросов напрямую зависит результат классификации, а значит, и последующего поиска ресурса. Но результат решения этих вопросов сильно зависит от субъективных представлений эксперта о предметной области. В случае независимой публикации ресурсов из одной предметной области разными людьми, они могут оказаться в разных категориях. Кроме того, даже у одного человека эти представления могут меняться с течением времени. Это неминуемо скажется на качестве и скорости поиска нужного ресурса конечным пользователем.

Таким образом, мы не можем полагаться на субъективность человека, когда речь идет о крупном информационном хранилище с большими объемами поступающих ресурсов. Нужно какое-то унифицированное представление о соответствии ресурсов дереву категорий.

Предлагаемым решением проблемы построения дерева категорий является *машинная кластеризация ресурсов*. При кластеризации мы полагаемся на гипотезу о том, что *смысловое содержание текста можно извлечь из статисти-*



ческого распределения слов. То есть по частотному распределению слов, составляющих текст ресурса (или аннотации), мы можем определить его семантику.

Согласно этому предположению кластеризация основывается на данных частотного анализа текста. Процесс частотного анализа связан с морфологическим анализом слов, составляющих текст. В этом процессе все словоформы одного слова должны быть учтены как одно слово, чтобы снизить шум на входах сети. Для этого применяется *механизм стемминга*.

*Стемминг* – это формальное выделение основы слова – стабильной, графически неизменной при склонении и спряжении части слова. Задача стемминга – это задача морфологического анализа языка. Словоформа представляется в виде:

*префикс+корень+суффикс+окончание+постфикс*

Основу слова составляют префикс и корень. Все словоформы в процессе стемминга приводятся к их основам, и уже основы участвуют в подсчете относительных частот.

По результатам подсчета частот выделяются наиболее часто встречаемые словоформы, так называемые «ключевые слова». *Ключевыми словами* считаются основы, которые чаще всего встретились в тексте и являются семантически значащими. Например, ключевыми словами не могут считаться предлоги, частицы, союзные слова, местоимения. Это так называемые «*стоп-слова*», которые отбрасываются при частотном анализе.

Основываясь на результаты частотного анализа, строится нейронная сеть Кохонена, которая, собственно, и проводит кластеризацию ресурсов. На входы сети подаются относительные частоты встречаемости ключевых слов по каждому ресурсу. Сеть обучается по алгоритму «победитель получает все». Выходы сети соответствуют списку категорий. После окончания обработки всех ресурсов сеть можно считать обученной. Для завершения построения дерева категорий эксперту необходимо назвать образованные нейронной сетью кластеры. При этом он может основываться на наборы ключевых слов и ресурсов, отнесенных сетью к данной категории.

После кластеризации задача публикации ресурсов решается этой же сетью. Она переводится в режим классификатора и выбирается наиболее подходящая для задачи функция активации. Поступающие ресурсы проходят стемминг и частотный анализ. Результаты частотного анализа подаются на входы сети. По состоянию выходов сети можно принимать решение о принадлежности ресурса к категориям.

Таким образом мы избавляем публикаторов и экспертов от работы по созданию дерева категорий и принятия решений о публикации ресурса. Вопрос «куда опубликовать ресурс» решен.

### 5.2.2. Поиск ресурсов

Рост объемов проектной информации, хранимой в электронном виде требует развития средств поиска нужных ресурсов. Обычно используются следующие виды поиска:

- обычный текстовый поиск по подстроке с некоторыми шаблонами;
- поиск, учитывающий морфологию языка;
- нечеткий поиск;
- браузеринг по категориям-каталогам.

При больших объемах и семантической разнородности хранимой информации поиска по подстроке с шаблонами и морфологического поиска уже становится недостаточно. Четкий поиск с заданными критериями сменяется так называемым «нечетким поиском», позволяющим указывать неточные значения критериев поиска. Но зачастую нет возможности точно сформулировать критерии поиска. Известно только, что ресурс может принадлежать определенным (достаточно общим) категориям. В таком случае поиск по категориям в виде простого выбора категории из дерева с просмотром всех ресурсов, наполняющих ее, бывает значительно удобнее. Этот вид поиска хоть и является исторически самым старым, но до сих пор не имеет отработанных стандартов на автоматизированные средства для построения индекса.

Автоматизированная *индексация информационных ресурсов* для последующего поиска в виде *браузинга* по категориям требует создания особого вида индекса. *Индекс* должен отражать структуру категорий и их взаимосвязь с информационными ресурсами. Для этого предложен следующий формат:

$$I_R = \{F_R, NN\},$$

где  $I_R$  – индекс ресурса;  $F_R$  – относительные частоты слов текста, составляющего ресурс;  $NN$  – обученная нейронная сеть.

Здесь структура категорий и их взаимосвязь с информационными ресурсами будет неявно заключена в матрице весов нейронной сети. Относительные частоты ( $F_R$ ) вычисляются по результатам стемминга и частотного анализа текста, составляющего ресурс. Нейронная сеть в индексе ( $NN$ ) – это нейронная сеть Кохонена, обученная в процессе кластеризации ресурсов. То есть процесс кластеризации ресурсов также дает нам и уникальный по возможностям индекс. На основе этого индекса мы можем осуществлять все вышеописанные виды поиска, включая браузеринг, а также комбинировать их. Например, можно дать возможность осуществлять полнотекстовый нечеткий поиск только в определенных категориях.

### 5.2.3. Управление хранилищем

В тени осталась также задача управления хранилищем. При последующем росте объемов информации, поступающей в хранилище, возможно возникновение ситуации, когда текущее дерево категорий не будет более покрывать всех

поступающих ресурсов. Симптомами такой ситуации будет являться появление ресурсов, не принадлежащих ни одной категории, или наоборот, принадлежащих сразу большому количеству категорий. В этом случае требуется перестройка дерева категорий с учетом вновь поступивших ресурсов. То есть повторная кластеризация всего объема ресурсов с последующим повторным именованием полученных кластеров.

**Итак, что нам дало наделение информационного хранилища искусственным интеллектом в виде нейронной сети?**

1. Информационный репозиторий сам может построить дерево категорий для хранящихся в нем ресурсов и разнести ресурсы по дереву. Теперь время экспертов требуется только для того, чтобы поименовать полученные кластеры на основе списков ключевых слов и ресурсов, отнесенных к данной категории.
2. Репозиторий сам решает, к какой категории отнести тот или иной ресурс. Это позволяет публикаторам не задумываться над тем, в каких категориях опубликовать ресурс. А пользователи не становятся заложниками субъективности экспертов и публикаторов.
3. Автоматизация кластеризации порождает индекс, позволяющий реализовать различные виды поиска, включая браузеринг по категориям. Это ускоряет поиск со стороны пользователей хранилища.
4. Хранилище само следит за актуальностью дерева категорий. При несоответствии дерева категорий поступающим ресурсам, то есть при невозможности правильно классифицировать ресурс, хранилище должно принять решение о перекластеризации.

Таким образом, от понятия «информационное хранилище» мы переходим к понятию *«интеллектуальное информационное хранилище»*. Набор его функций стал более широким и теперь включает в дополнение к предыдущим функциям:

- импорт уже имеющихся ресурсов;
- кластеризация имеющихся ресурсов с целью составления дерева категорий;
- обработка поступающих ресурсов, то есть выполнение классификации поступающих ресурсов и публикация их в соответствующих категориях;
- возможность полнотекстового и нечеткого поиска по текстам ресурсов, возможность поиска по дереву категорий;
- слежение за тем, чтобы дерево категорий всегда покрывало поступающие ресурсы (чтобы не было ресурсов, не относящихся ни к одной категории, или наоборот, относящихся ко всем категориям).

### **5.3. Функции автоматизированной системы проектирования нечетких контроллеров**

Современные нечеткие контроллеры можно разделить на собственно нечеткие традиционные контроллеры и нейро-нечеткие контроллеры. Нечеткие контроллеры представляют собой систему нечеткого вывода, причем функции принадлежности формируются на основе экспертных оценок. Реализация нечеткого контроллера представляет собой базу правил, базу лингвистических переменных (имен и функций принадлежности) и типовую процедуру логического вывода.

Аппаратная реализация нечеткого контроллера на основе микропроцессора включает в себя:

- табличное представление имен переменных, параметризованных функций принадлежности функций и нечетких правил;
- модули, реализующие нечеткий вывод по Мамдани, Сугено, Цукamoto, Ларсену и др.

Система автоматизированного проектирования нечетких контроллеров представляет собой набор следующих средств:

1. Диалоговые средства, позволяющие эксперту определить имена, функции принадлежности и правила.
2. Средства визуализации нечетких правил, управляющей поверхности решений.
3. Набор генераторов модулей табличного представления нечетких правил и логического вывода для разных систем команд на основе библиотеки модулей.

Описанная структура является наиболее распространенной. Основным недостатком такой структуры является использование субъективных представлений эксперта о лингвистических переменных и правилах. Известны методы извлечения функций принадлежности из паттернов образцов проблемной области с помощью нечетких нейронных сетей, позволяющих с помощью кластеризации объектов проблемной области идентифицировать параметры функций принадлежности. В качестве средства кластеризации можно использовать алгоритм FCM или нейронную сеть Кохонена. Для уточнения параметров функций принадлежности используют алгоритм обратного распространения ошибки. Таким образом, в структуре автоматизированного проектирования нечетких контроллеров появляются новые компоненты:

- модуль кластеризации объектов проблемной области;
- модуль идентификации параметров функций принадлежности.

## 5.4. Реализация системы проектирования НК

### 5.4.1. Анализ методов реализации нечетких контроллеров

Началом возникновения теории нечетких множеств (Fuzzy Sets Theory) можно считать 1965г., когда профессор Лотфи Заде (Lotfi Zadeh) из университета Беркли опубликовал основополагающую работу «Fuzzy Sets» в журнале «Information and Control» [1]. Эта работа заложила основы моделирования интеллектуальной деятельности человека и явилась начальным толчком к развитию новой математической теории.

Прилагательное «fuzzy» можно перевести на русский язык как нечеткий, размытый. Оно было введено в название новой теории с целью отдаления от традиционной четкой математики и булевой логики, оперирующих с четкими понятиями: «принадлежит - не принадлежит», «истина – ложь». Сама теория возникла, как «неудовлетворенность математическими методами классической теории систем, которая вынуждала добиваться искусственной точности, неуместной во многих системах реального мира, особенно в так называемых гуманистических системах, включающих людей» [2].

Математическая теория нечетких множеств, позволяет давать описание нечетким понятиям и знаниям, оперировать этими знаниями и делать нечеткие выводы. Нечеткая логика, в основном, обеспечивает эффективные средства отображения неопределенностей и неточностей реального мира. Наличие математических средств отражения нечеткости исходной информации позволяет построить модель, адекватную реальности. Основанные на этой теории методы построения компьютерных нечетких систем существенно расширяют области применения компьютеров.

В последнее время нечеткое управление является одной из самых активных и результативных областей исследований применения теории нечетких множеств.

Применение нечеткого управления оказывается особенно полезным, когда технологические процессы являются слишком сложными для анализа с помощью общепринятых количественных методов, или когда доступные источники информации интерпретируются качественно, неточно или неопределенно. Многие комплексные процессы представляют собой многопараметрические системы и являются существенно нелинейными, а в ряде случаев нелинейными во времени. Для применения более сложных методов управления часто не хватает информации о процессе и надежных математических моделей, описывающих процесс. Знания о ходе процесса, на которые опирается оператор, реализуются им в виде правил «если – то», имеющих нечеткое информационное содержание. Этот же принцип использован при автоматизации управления процессами на базе нечеткого контроллера. Например: «если X есть большое положительное число и Y есть малое положительное, то C есть положительное

среднее» (для контроллера с двумя входными сигналами  $X$  и  $Y$  и одной выходной переменной  $C$ ). Термы «положительное большое», «положительное среднее» и «положительное малое» представляют собой так называемые лингвистические переменные, и являются неопределенными описаниями конечных значений входных переменных  $X$  и  $Y$ , и выходной переменной  $C$ . Каждое лингвистическое правило интерпретируется начальным отношением, которое, в свою очередь, определяет в общем случае отношение между нечеткими входными и нечеткими выходными значениями.

Экспериментально было доказано, что нечеткое управление более эффективно и выдает лучшие результаты, по сравнению с результатами, получаемыми при традиционных алгоритмах управления.

Началом практического применения теории нечетких множеств можно считать 1975г., когда Мамдани и Ассилиан (Mamdani and Assilian) построили первый нечеткий контролер для управления простым паровым двигателем [3]. В 1982г. Холмблад и Остергаард (Holmblad and Ostergaard) разработали первый промышленный нечеткий контроллер, который был внедрен в управление процессом обжига цемента на заводе в Дании [4].

В последствии, было разработано множество подобных устройств, например, контроллер управления движением (Saski Akiyama, 1988), контроллер руки робота (Tanscheit Scharf, 1988), контроллер температуры теплого воздуха (Ollero Garc ia Cerezo, 1988) и множество других.

Успех первого промышленного контролера, основанного на нечетких лингвистических правилах «Если – то» привел к всплеску интереса к теории нечетких множеств среди математиков и инженеров. Несколько позднее Бартом Коско (Bart Kosko) была доказана теорема о нечеткой аппроксимации (Fuzzy Approximation Theorem), согласно которой любая математическая система может быть аппроксимирована системой, основанной на нечеткой логике [5]. Другими словами, с помощью естественно-языковых высказываний - правил «Если – то», с последующей их формализацией средствами теории нечетких множеств, можно сколько угодно точно отразить произвольную взаимосвязь «входы – выход» без использования сложного аппарата дифференциального и интегрального исчислений, традиционно применяемого в управлении и идентификации.

Системы управления, основанные на нечеткой логике, разработаны и успешно внедрены в таких областях, как: управление технологическими процессами, управление транспортом, бытовая техника и электроника, медицинская диагностика, техническая диагностика, финансовый менеджмент, биржевое прогнозирование, распознавание образов. Область применения систем, основанных на нечеткой логике очень широка - от бытовой электроники и автоматики, до средств наведения ракет ПВО и управления боевыми вертолетами в военной технике.

Практический опыт разработки систем нечеткого логического вывода свидетельствует, что сроки и стоимость их проектирования значительно мень-

ше, чем при использовании традиционного математического аппарата, при этом обеспечивается требуемый уровень робастности и прозрачности моделей.

#### 5.4.2. Некоторые существующие аппаратные fuzzy процессоры

Приложения с нечеткой логикой развивались не только на логическом уровне. Разработчики систем управления начали встраивать нечеткую логику непосредственно в процессоры. Так, в 1986 году, компания *AT&T Bell Labs* начала создавать процессоры со встроенной нечеткой логикой обработки информации. В связи с популярностью и распространенностью традиционных микроконтроллеров, в Европе и США ведутся интенсивные работы по интеграции fuzzy команд в ассемблеры для программирования промышленных контроллеров (чипы *Motorola 68HC11*, *68HC12*, *68HC21*, *Intel MCS96*). А также, разрабатываются различные варианты fuzzy-сопроцессоров, работающих совместно с центральным процессором по общей шине данных, и помогающих в обработке информации и оптимизации использования правил (*Siemens Nixdorf*).

##### **Некоторые существующие аппаратные fuzzy процессоры:**

- **ST52 Duallogic (STMicroelectronics)** - семейство 8-битовых микроконтроллеров, содержащих в одном корпусе традиционное вычислительное ядро, ядро для fuzzy-вычислений и периферийные схемы. Поддерживает специальный набор инструкций для работы с нечеткой логикой и позволяет определять несколько независимых наборов правил для нескольких различных алгоритмов;

- **ST62 (STMicroelectronics)** - 8-битовый микроконтроллер со встроенной, однократно программируемой памятью для автомобильной промышленности, продолжение семейства Duallogic. Расширенный температурный диапазон (от -40° до +125°C), гарантированный срок хранения данных для памяти EPROM и EEPROM не менее 20 лет;

- **68HC12 (Motorola)** - fuzzy-микроконтроллер, базирующийся на ядре стандартного микроконтроллера 68HC11 и содержащий специальные функции для реализации нечеткой логики. Предназначен для использования с программным пакетом FuzzyTech и позволяет увеличить скорость выполнения приложений, созданных в этом пакете, до 15 раз и компактность кода до 6 раз по сравнению с реализацией на обычном ядре 68HC11;

- **VY86C570 (Togai InfraLogic)** - fuzzy-сопроцессор, 12 -битовое ядро FCA (Fuzzy Computational Acceleration[Акселерация Нечетких Вычислений]), 4Kx12 бит памяти OCTD (Observation, Conclusion, & Temporary Data[Наблюдение, Вывод и Временные Данные]), память RB (Rule Base[База Правил]), и интерфейсная логика в одном корпусе;

- **SAE 81C99 (Siemens)** - fuzzy-процессор, способный выполнять восемь программируемых алгоритмов, обрабатывать 256 входных переменных и формировать до 64 выходных значений максимум по 16384 правилам. Может использоваться как отдельное устройство или в качестве сопроцессора для 8 и 16-

разрядных микроконтроллеров. Скорость работы - 10 миллионов правил в секунду.

Но анализ стоимости продукции показывает, что гораздо дешевле эмулировать нечеткую логику, чем разрабатывать аппаратные устройства. Применение специализированных аппаратных решений оправданно для получения реальных преимуществ в быстродействии. Для эмулирования нечеткого управления, достаточно сложным вопросом является разработка системы, и последующая трансляция на традиционный язык программирования. После этого уже можно преобразовывать программу в код для конкретного микроконтроллера.

Компания Аптроникс(*Aptronix*) предлагает использовать язык Java, имеющий все необходимое для достаточно адекватного воспроизведения инструкций нечеткой логики реализуемого приложения методами языка. Кроме того, использование Java API открывает новые перспективы для исследования fuzzy-систем. Сеть Internet, как глобальная среда распространения Java-приложений, идеально подходит для интеграции прикладных устройств, созданных при помощи алгоритмов нечеткой логики.

В качестве примера можно привести устройство для управления душем [6], эмуляция работы которого, написана на языке Java.

#### 5.4.3. Пример проектирования нечеткого контроллера в среде Matlab

Существуют среды для разработки нечетких контроллеров, такие как FuzzyStudio(*STMicroelectronics*), Visual FIVE (*STMicroelectronics*), FuzzyTech(*INFORM GmbH*), позволяющие эмулировать нечеткое управление на ядре конкретного микроконтроллера, или сформировать программный код на конкретном языке программирования.

Рассмотрим процесс проектирования нечеткого контроллера.

Возьмем простой пример – вентилятор охлаждения процессора в компьютере. На зависимость скорости вращения вентилятора от температуры процессора влияют такие факторы, как площадь радиатора, температура окружающей среды, загруженность процессора, а также энергосбережение и уровень шума. Выберем в качестве оптимального рабочего диапазона температуру  $T$  в пределах  $65^{\circ}\text{C}$ . Мотор вентилятора работает в промежутке от 2 до 5.5 V.

Воспользуемся программой *Matlab* для построения модели.

Определяем входные и выходные переменные:



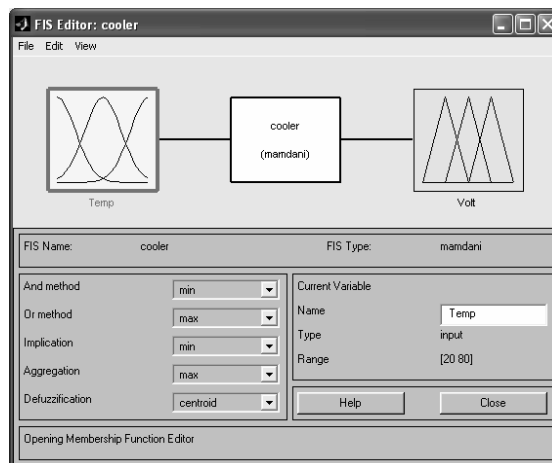


Рис. 5.1. Окно интерфейса определения структуры правил

Определяем функции принадлежности для входной переменной:

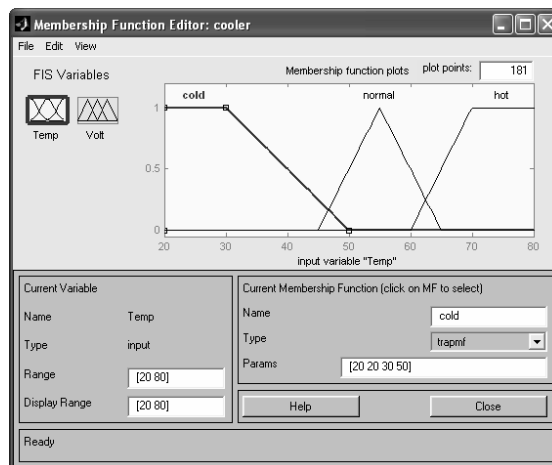


Рис. 5.2. Окно интерфейса определения входной переменной

И для выходной переменной:

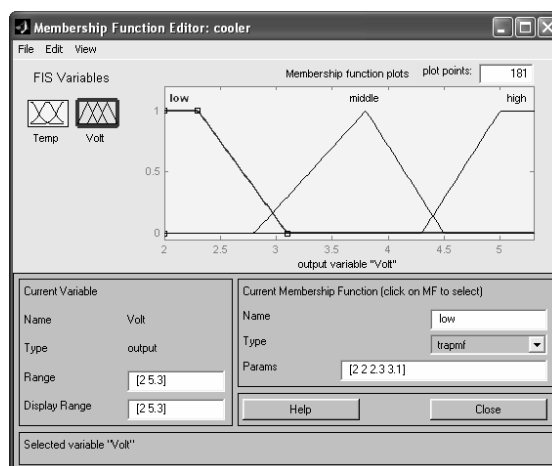


Рис. 5.3. Окно интерфейса для определения выходной переменной

Определим правила:

Если температура холодная, то напряжение низкое.

(If Temp is cold then Volt is low)

Если температура нормальная, то напряжение среднее.

(If Temp is normal then Volt is middle)

Если температура горячая, то напряжение высокое.

(If Temp is hot then Volt is high)

Посмотрим результат выполнения правил:

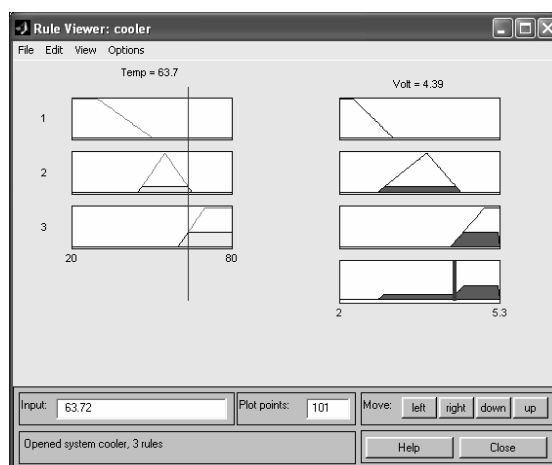


Рис. 5.4. Система нечетких правил управления вентилятором компьютера

В итоге получим следующую поверхность решений:

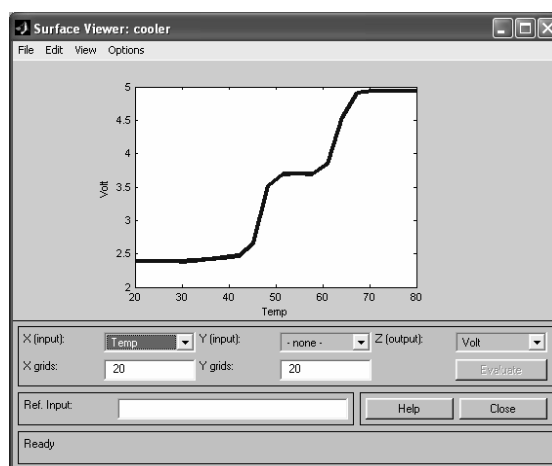


Рис. 5.5. Управляющая функция

Теперь необходимо реализовать полученный нечеткий контроллер на традиционном языке программирования. Определим основные этапы.

1. Определение входных и выходных переменных, и их диапазонов.

Переменная Temp, диапазон [20 80].

Переменная Volt, диапазон [2 5.3].

2. Функции принадлежности и правила можно записать в виде таблицы.

В столбцах будут находиться граничные значения функций принадлежности, в нашем случае каждую функцию можно представить в виде 4 значений.

	1	1	1	1	2	2	2	2
Cold - low	0	0	0	0			.3	.1
Normal - middle	5	5	5	5	.8	.8	.8	.5
Hot - high	0	0	0	0	.3		.3	.3

3. Получив значение входной переменной, рассматриваем, какое из правил удовлетворяет данному значению, и с какой степенью уверенности. В случае срабатывания нескольких правил, производим суммирование функций принадлежности по максминному методу.

4. Находим результирующее значение по методу центра тяжести.

Перечисленные операции легко реализуются на любом из языков программирования, разница только в числе строк кода и определении формата хранения табличных данных. Удобнее реализовывать данный алгоритм на языке C++, поскольку большинство микроконтроллеров имеют программное обеспечение для конвертации программ на языке C++ в ассемблерный код микроконтроллера.

### Библиографический список

1. Zadeh, L. Fuzzy sets // Information and Control. — 1965. — №8. — P. 338-353.
2. Заде, Л. Понятие лингвистической переменной и ее применение к принятию приближенных решений. — М.: Мир, 1976. — 167 с.
3. Mamdami, E. H. and Assilian S., «An experiment in linguistic synthesis with a fuzzy logic controller», International Journal of Man-Machine Studies, vol. 7, pp. 1-13, 1975.
4. Holmblad, L. P., and Otergaard, J. J., « Control of a cement kiln by fuzzy logic», in Fuzzy Information and Decision Processes , Gupta, M. M. and Sanchez, E. E. New Work: North-Holland, pp. 389-399, 1982.
5. Kosko, B., Fuzzy Thinking, The New Science of Fuzzy Logic, Hyperion Books, 1994.
6. Fuzzy Shower Demo  
[http://www.iit.nrc.ca/IR\\_public/fuzzy/fuzzyShower.html](http://www.iit.nrc.ca/IR_public/fuzzy/fuzzyShower.html)
7. Аверкин, А. Н., Федосеева И.Н. Параметрические логики в интеллектуальных системах управления. - М.: ВЦ РАН, 2000. - 105 с.

8. Ярушкина, Н. Г. Основы теории нечетких и гибридных систем: Учеб. пособие.- М.: Финансы и статистика, 2004.- 320 с: ил.

## Глава 6 МЯГКИЕ ВЫЧИСЛЕНИЯ В ЭКОНОМИЧЕСКИХ И ГУМАНИТАРНЫХ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМАХ

### 6.1. Задача анализа терминосистемы нечеткой логики

Организация и анализ любой *терминосистемы*, а тем более относящейся к формирующимся, является достаточно длительным процессом, требующим больших затрат времени и участия многих людей. Важной задачей для прикладной лингвистики является моделирование терминосистемы при помощи новых информационных технологий, позволяющей специалистам в области лингвистики сократить время создания отраслевых словарей, обработки данных. Важным является также разработка принципа такого описания лексики отраслевого подъязыка, которое было бы достаточным для успешной профессиональной коммуникации.

Разрабатываемый словарь должен составляться с использованием как логико-понятийных, так и вероятностно-статистических методик. Первые из них (фреймовое представление) необходимы для представления когнитивного содержания терминов, а вторые – для описания лингво-прагматических характеристик единиц исследуемого подъязыка.

В связи с тем, что немецкоязычная терминосистема нечеткой логики прежде не была предметом специального исследования, на первом этапе анализа мы взяли за основу методику, предложенную С. В. Гриневым.

В 1993 году им был разработан и опубликован перечень критериев, с помощью которых в формализованном виде можно представлять наиболее значимые характеристики любой терминосистемы. В частности, по мнению С. В. Гринева, к *первой группе* критериев относятся:

- исторические характеристики, включающие возраст терминосистемы;
- данные о ее происхождении;
- замкнутость (соотношение собственных терминов и заимствованных из других терминологий этого же языка).

*Вторую группу* параметров представляют характеристики, названные С. В. Гриневым «формальными»:

- размер терминосистемы (до 100 терминов - микротерминология, 100 - 1000 - мезотерминология, свыше 1000 - макротерминология);

- структурный состав терминов (виды и соотношение их структурных типов);
- средняя длина терминов (лексическая - среднее число слов, входящих в термин; знаковая - среднее количество знаков в терминах);
- мотивированность, под которой автор подразумевает семантическую прозрачность;
- систематизированность.

В *третью группу* - семантические характеристики - включены такие показатели, как:

- предметная отнесенность;
- полнота терминологии (отсутствие в ней лакун);
- семантическая целостность (отсутствие автономных фрагментов, оторванных от основного корпуса терминологии);
- понятийная изоморфность терминологии (установление доли омонимии, полисемии, синонимии);
- абстрактность/конкретность и категориальные соотношения (соотношение терминов, называющих понятия разных категорий: объекты, процессы и т. д.);
- структурированность - глубина иерархии (процентное соотношение терминов, связанных иерархическими отношениями, и терминов, связанных чисто ассоциативными отношениями).

И, наконец, в качестве функциональных параметров автор предлагает учитывать нормированность, общепринятость и употребительность терминологии [1].

Одним из условий моделирования терминосистемы является отбор терминов отрасли из текстов. При этом используются следующие критерии терминологичности лексических единиц: 1) термин соотнесен со специальным (научным, или техническим, или другим профессиональным) понятием отрасли знания; 2) термин существует как член определенной терминосистемы; 3) термином может быть слово, словосочетание, аббревиатура, символ, сочетание слова и буквенных символов, сочетание слова и цифровых символов если данная единица соотносится в плане содержания с определенным понятием в логико-понятийной системе отрасли знания.

Отбор терминов из текстов осуществляется также с учетом вопросов об отношении терминов к другим пластам лексики, о частеречной принадлежности терминов, о терминологических словосочетаниях, о варьировании терминов в тексте. Общенаучная, общетехническая, межотраслевая лексика не представляет самостоятельного интереса при моделировании терминосистемы, поскольку не несет специальной отраслевой понятийной информации. Данная лексика рассматривается лишь в составе терминологических словосочетаний отрасли, в формировании которых она активно участвует. В плане частеречной принадлежности из текстов отбираются, в основном, существительные и образованные на их базе словосочетания, которые являются главным способом выражения

понятий в моделируемой терминосистеме. При отборе терминов из текстов в моделируемую терминосистему включаются все языковые варианты терминов, для которых характерно тождество семантики: они выражают одно понятие, при этом значение каждого варианта очевидно без обращения к контексту [3].

## **6.2. Реализация системы анализа терминосистемы нечеткой логики**

При отборе лексики используют метод экспертных оценок, когда специалист-эксперт в области нечеткой логики подтверждал или опровергал принадлежность выбранного термина к подязыку нечеткой логики или же к другим подязыкам нашего словаря.

Материалом конкретного описанного ниже исследования являются научные тексты, статьи, монографии, учебники по нечеткой логике на немецком языке.

### **6.2.1. Модель образования терминосистемы нечеткой логики**

Терминосистема нечеткой логики достаточно молода - Л.А.Заде издал первую научную работу и придумал термин “fuzzy logic” лишь в 1965 году, а исследования в этой области на немецком языке появились еще позже. Но из-за перспектив использования достижений этой науки в области современных технологий, ее терминология развивается быстрыми темпами.

В основе образования анализируемой терминосистемы лежит гетерогенная модель (ее возникновение стало результатом взаимодействия нескольких исходных терминосистем, номинирующих концепты тех отраслей знаний, на основе которых развилась новая наука).

В связи с этим в терминосистеме нечеткой логики можно выделить следующие группы терминов:

- 1) базовые термины, например, «Neuron», «Steuer», заимствованные из терминологии других терминосистем с сохранением первоначального значения;
- 2) термины, являющиеся производными от этих лексических единиц, например, «fuzzifiziert» (производное от “Fuzzy”), а также словосочетания, в которых один и более компонентов являются базовыми терминами, например, «diskrete unscharfe Zahl» (“diskret” – базовый математический термин, “unscharf” – базовый термин подязыка нечеткой логики);
- 3) термины, заимствованные из других областей знания, к которым добавлен базовый термин Fuzzy, например «Fuzzy-Differenz» (“Differenz” – мат. «разность»).

Структурные параметры модели:

#### ***а) Размер терминосистемы.***

Терминология в настоящее время содержит 660 единиц, что дает основание отнести ее к группе *мезотерминосистем*. Общий словарь мы подразделили

при помощи экспертов на шесть подсловарей, термины которых используются в научных текстах по нечеткой логике: «Нечеткая логика» (269 терминов), «Логика» (224), «Математика» (95), «Управляющие системы» (64), «Искусственный интеллект» (5), «Компьютер» (2).

**б) Структурный состав терминов (соотношение однословных, бинарных и многокомпонентных терминов).**

Соотношение однословных и сверхсловных единиц в изучаемой терминосистеме (подсловарь Fuzzy) достаточно типично. Преобладают бинарные (двухсловные) термины - 122, из которых большинство образовано по модели  $A + N$  («beschränkte Summe»), и по модели  $N + N$  («Fuzzy-Set»). На втором месте по распространенности - однословные термины - 74. Трехсловные термины представлены 68 терминологическими единицами; четырехсловные – 4; термин, состоящий более чем из четырех слов - 1.

**в) Основные способы терминообразования.**

Единицы, вошедшие в исследуемую терминосистему, образованы различными способами: семантическим (например, для общелитературного слова «Ausgabe» основным является значение «издание», тогда как в исследуемой терминосистеме - это выход, выходное устройство), синтаксическим (например, «LR-Grundverknüpfung», «linguistische Variable»), морфологическим (за счет префиксации, например, «Defuzzyfizierung»; суффиксации, например, «Fuzzyfizierung»). Большинство исследуемых терминов были образованы за счет присоединения к существительному слова «Fuzzy». Есть термины, заимствованные из английского, например «Overfitting», «Support».

#### 6.2.2. Синонимичность

Что касается синонимичных терминов, то тут следует сказать, что немецкий язык не предполагает такого большого количества синонимов, как, например, английский язык, но все же они есть. Чаще всего синонимы образуются тогда, когда в языке науки приживаются английский термин и его немецкое соответствие, например «Fuzzy-Zahl» и «unscharfe Zahl». Для базового термина «нечеткость» в немецком языке существуют даже пять синонимов – «Fuzzy», «Unschärfigkeit», «Vagheit», «Ungenauigkeit», «Impräzision».

Наличие синонимов дает основание, с одной стороны, характеризовать исследуемую терминосистему как формирующуюся, то есть располагающую неокончательно устоявшимся терминологическим аппаратом, с другой стороны, несмотря на это, анализируемая терминосистема обладает достаточно широким диапазоном номинативных средств.

Доказательством самостоятельности отрасли знания и сферы деятельности человека, по мнению многих авторов, является «наличие сложившейся специальной терминосистемы, обслуживающей их». У лингвистов нет единого подхода к определению самостоятельности терминосистемы. По С. В. Гриневу,



степень самостоятельности терминосистемы определяется соотношением собственных терминов и заимствованных из других терминосистем или из общелитературного языка [1]. В соответствии с этими критериями немецкоязычная терминосистема нечеткой логики может быть признана самостоятельной, потому что она номинирует понятия уже сложившейся новой отрасли науки.

Доказательством самостоятельности анализируемой терминосистемы является также значительное количество терминов, заимствованных из других областей, но подвергшихся существенной модификации семантического содержания в рамках терминосистемы нечеткой логики (например, «Semantik» заимствован из лингвистики, «Neuron», «Gehirn» - из биологии, «Kern» - из физики).

Таким образом, рассмотренные экстралингвистические и лингвистические характеристики терминосистемы нечеткой логики, давая общее представление о времени ее возникновения, структуре, основных способах и средствах номинации важнейших понятий, взаимоотношениях с другими терминосистемами, семантических особенностях, свидетельствуют о том, что анализируемая терминосистема является сложным образованием, находящимся в процессе формирования.

### 6.2.3. Реализация программы обработки терминосистемы

Рассмотрим программу Fuzzy-Base, среда разработки которой Delphi 7. База данных (где содержится словарь и данные анализа) написана на языке InterBase 6.5, т. к. он поддерживает многоязычную среду и позволяет создать таблицы с полями разных кодовых таблиц, операционные системы и среда разработки которой:

**Windows 98, Windows XP**  
**INTERBASE 6.5**  
**DELPHI 7**

Для создания базы данных, содержащей немецкие и русские слова, потребуется поддерживать хранение данных в разных кодировках. Были рассмотрены варианты использования следующих баз данных:

**FOXPRO, PARADOX 7, ORACLE 7.3.4.5.1, INTERBASE 6.5**

Самым распространенным способом решения такого вопроса служит организация двух таблиц в разных кодировках со связью между собой по ключевому полю (FOXPRO и PARADOX). Рассмотренная версия ORACLE является одной из самых стабильных версий, однако она не имеет возможности организации таблиц в разных кодировках. Наиболее интересной в этой связи кажется СУБД INTERBASE, которая, кроме традиционного способа организации отдельной базы данных в выбранной кодировке, позволяет создать отдельные таблицы с полями в разных кодировках. Мощный SQL (Structured Query Language) язык, организация хранения отдельной базы данных в одном файле, лег-

кость установки как на локальные ресурсы, так и на сетевые делают эту СУБД самой привлекательной для организации хранилища мультязычных данных в нашем проекте.

Первоначально определены:

1. Формат таблиц словаря и обработки данных для анализа.
2. Синтаксис для ввода первоначальных данных словаря.

Словарь, в котором пока около 300 терминов, составлялся следующим образом:

1. Немецкое слово и все его формы (которые невозможно задать автоматически из-за большого количества исключений), разделенные между собой точкой с запятой).
2. Русский перевод (близкие значения разделены запятой, более общие – точкой с запятой, полисемические – цифрами, абсолютно несоотносительных слов (омонимов) пока не встречалось).
3. Английское (оригинальное) соответствие (если есть).

Эти три части словаря разделены следующим образом – пробел, тире, пробел. Необязательные данные (в нашем случае это артикли) заключены в квадратные скобки.

Примеры словарных статей:

[die] Fuzzy-Menge; [die] Fuzzy-Mengen – нечеткое множество - fuzzy set; fuzzy sets

[der] linguistische Operator-Modifikator; linguistischer Operator-Modifikator; [des] linguistischen Operator-Modifikators; linguistischem Operator-Modifikator; [den] linguistischen Operator-Modifikator; [die] linguistischen Operatoren-Modifikatoren; linguistische Operatoren-Modifikatoren; linguistischer Operatoren-Modifikatoren – лингвистический оператор-модификатор – linguistic operator-modifier

[die] L-Referenzfunktion; [die] L-Referenzfunktionen – эталонная функция левого фронта

[die] linguistische Variable; linguistischer Variable; linguistische Variablen; [die] linguistischen Variablen, LV; LVs – лингвистическая переменная – linguistic variable

Полезным модулем системы является программа по синтаксическому разбору формата MS Word. Связь Word с программой осуществляется через OLE технологии (MS Office XP). Минус этой системы в том, что она медленно работает.

Программа обработки терминосистемы должна удовлетворять двум ограничениям. Первое – ориентация на использование системы Word, так как этот

текстовый редактор поддерживает многоязычность (Unicode). DOS-кодировки не позволяют поддерживать немецкие и русские символы (т.к. они находятся в расширенной части таблицы кодировки). Второе ограничение – нам необходимо, чтобы сама программа обработки и введения словаря и анализа текста смогла понимать мультязычную кодировку. Для хранения одного символа может потребоваться не один байт, а два, три или четыре, в зависимости от версии кодировки Unicode.

1. *Создание базы данных (WISQL)*

```
CREATE DATABASE "EXAMPLE.GDB"  
USER "SYSDBA" PASSWORD "masterkey"  
LENGTH=100  
DEFAULT CHARACTER SET NONE
```

В данном случае никакой кодовой таблицы по умолчанию не указывается.

2. *Создание таблицы (WISQL)*

```
CREATE TABLE EX001  
(  
SLOVO_D VARCHAR(100) CHARACTER SET WIN1252 NOT NULL,  
  
SLOVO_R VARCHAR(100) CHARACTER SET WIN1251,  
SLOVO_U VARCHAR(100) CHARACTER SET UNICODE_FSS  
)
```

Таким скриптом создается таблица с полями в разной кодировке:

WIN1252 – для немецкого языка;

WIN1251 – для русского языка;

UNICODE\_FSS – универсальная кодировка.

3. *Добавление записи в таблицу (WISQL)*

а)

```
INSERT INTO EX001 (SLOVO_D, SLOVO_R, SLOVO_U)  
VALUES  
(  
_WIN1252 'Unschärfe',  
_WIN1251 'нечеткость',  
_WIN1251 'нечеткость'  
)
```

б)

```
INSERT INTO EX001 (SLOVO_D, SLOVO_R, SLOVO_U)  
VALUES
```

```
(
  _WIN1252 'Unschärfe',
  _WIN1251 'нечеткость',
  _WIN1252 'Unschärfe'
)
```

В случае а) в поле SLOVO\_U добавляется значение, подготовленное в кодировке \_WIN1251, а в случае б) значение подготовлено в кодировке \_WIN1252. Такой пример характерен для поля таблицы в кодировке UNICODE\_FSS.

Однако в общем случае, если кодировка поля не совпадает с кодировкой подготовленных данных для записи или изменения система выдаст сообщение об ошибке:

*-Cannot transliterate character between character sets*

При необходимости можно поддерживать специальные поля UNICODE:

```
INSERT INTO EX001 (SLOVO_D ,SLOVO_U)
VALUES ('ein',_UNICODE_FSS 'Unschärfe - нечеткость')
```

#### 4. Выборка данных:

Подход к выборке данных полностью аналогичен методу добавления и редактирования записей в таблице:

```
SELECT * FROM EX001 WHERE
SLOVO_D=_WIN1252 'Unschärfe';
```

```
SELECT SLOVO_U FROM EX001 WHERE
SLOVO_U LIKE _UNICODE_FSS '%ärfe%';
```

#### 5.Способ работы с многокодowymi таблицами в DELPHI

В данном конкретном примере предлагается работать без использования алиасов. Такой способ предполагает минимальные настройки BDE на станции клиента и простоту переноса локальных приложений. Однако способ работы с использованием алиаса базы данных, тоже возможен.

Основной проблемой при работе в DELPHI с многокодowymi и уникальными полями таблиц InterBase является непонимание стандартными SQL компонентами синтаксиса `where slovo_d like _WIN1252 :PAR` при формировании конечного SQL запроса подсистемой динамического языка SQL.

Один из способов решить эту проблему – формировать программно SQL запрос в компоненте TIBQuery. Однако и этот способ имеет ограничение на работу, т. к. property SQL: TString.

Это означает, что записать значение поля в кодировке UNICODE\_FSS таким способом тоже нельзя.

Ниже схематично приводится пример простой работы с полями таблиц InterBase.

### *1. Подключение к базе данных*

TIBDatabase.DatabaseName	EXAMPLE.GDB
TIBDatabase.params	user_name=sysdba
password=masterkey	
lc_ctype=NONE	
TIBTransaction	
TIBTable.TableName	EX001

### *2. Добавление записей*

TIBQuery

SL: WideString;

With IBQuery do begin

SQL.Clear;

SQL.Add('INSERT INTO EX001 (SLOVO\_D );

SQL.Add('VALUES (');

SQL.Add('\_WIN1252 '"+WideStrToWIN1252(SL)+'");

SQL.Add(')');

ExecSql;

End;

### *3. Выборка данных*

With IBQuery do begin

Close;

SQL.Clear;

SQL.Add('select slovo\_d, count(slovo) as CNT from EX001');

SQL.Add('where slovo\_d like \_WIN1252 '"+SL+'%");

SQL.Add('group by slovo');

Open;

End;

Для облегчения написания процедур перекодировки из одной кодовой таблицы в другую можно воспользоваться следующей ссылкой:

The Unicode units provide - <http://fundamentals.sourceforge.net/unicode.html>

А по этой ссылке можно получить специализированные Unicode компоненты:

Tnt Delphi Unicode Controls -

[http://home.ccci.org/wolbrink/tnt/delphi\\_unicode\\_controls.htm](http://home.ccci.org/wolbrink/tnt/delphi_unicode_controls.htm)

Использование Unicode значительно упрощает создание многоязычных приложений. Поэтому, создавая программы с прицелом на этот стандарт, можно заложить хорошую базу для локализации программного продукта.

Программа выполняет следующие статистические исследования:

1. обрабатывает тексты и подсчитывает общее количество слов;
2. подсчитывает общее количество терминов, разделяя их по словарям;
3. рисует диаграмму разделения терминов по словарям;
4. высчитывает процентное соотношение терминов разных подсловарей с коэффициентом и без;
5. выстраивает ранговый словарь найденных терминов;
6. переводит термины на русский и на английский языки;
7. показывает все грамматические формы немецкого термина;
8. выявляет новые термины;
9. отслеживает в подсловарях дубликаты и ошибки.

На первом этапе нами было обработано 18 научных текстов по нечеткой логике, общим объемом 80 015 слов, из которых 4646 являются терминами. Термины разделились по словарям следующим образом:

1. Логика – 2345;
2. Нечеткая логика – 1589;
3. Математика – 510;
4. Управляющие системы – 167;
5. Искусственный интеллект – 35.

Программа выявила также 63 новых термина.

В результате исследований мы получили следующие данные (приводится начало словаря):

Ранг	словарь	термин	частота
1	Logik	Modell	442
2	Logik	Ergebnis	295
3	Fuzzy	Fuzzy-Menge	162
4	Fuzzy	Fuzzy-Logic	160
5	Logik	Menge	145
6	Fuzzy	Zugehörigkeitsfunktion	121
7	Logik	Erfüllungsgrad	107
8	Fuzzy	neuronale Netz	97
9	Mathematik	Begriff	92
10	Fuzzy	unscharf	85
11	Logik	Folgerung	75

12	Logik	Berechnung	73
13	Fuzzy	Neuron	72
14	Logik	definieren	71
15	Logik	Gewichtung	71
16	Logik	Operator	69
17	Logik	Verknüpfung	65
18	Mathematik	Istwert	62
19	Logik	Element	61
20	Logik	Prämisse	56
21	Fuzzy	linguistische Variable	54
22	Mathematik	Parameter	52
23	Logik	Ausgangsgröße	51
24	LT	Stellgröße	51
25	Logik	linguistisch	50
26	LT	Regelung	50
27	Logik	Eingangsgröße	49
28	Fuzzy	gewichten	49
29	Logik	Ausprägung	48
30	Fuzzy	Gewicht	48
31	Logik	Intervall	45
32	Fuzzy	Unschärfe	44
33	Mathematik	Modellierung	42
34	Logik	UND-Verknüpfung	41
35	LT	Steuerung	38

**Отчет по анализу каждого обработанного текста** выглядит следующим образом:

C:\IVP\FUZZY.BASE\DOC\MATERIAL.2004\STATISTIC\06. TTilli Fuzzy Shell fuer Windows.doc

---

1	Fuzzy	52
2	LT	4
3	Logik	144
4	Mathematik	73

---

**Итого: 273 из 4163**

**Отчет по анализу каждого обработанного текста в Log-файле**

28/01/2004 17:11:45-----

C:\IVP\FUZZY.BASE\DOC\MATERIAL.2004\STATISTIC\01. Grundlagen der Fuzzy-Logik.doc

•добавь выражение: Fuzzy-Regelung

- добавь выражение: Fuzzy-Regelung
- добавь выражение: Maximum-und
- добавь выражение: Maximum-Verknüpfung
- добавь выражение: Minimum-Verknüpfung

К сожалению, пока в Log-файле не читаются умляуты.

28/01/2004 16:46:11-----

C:\IVP\FUZZY.BASE\DOC\SLOVAR.2004.w\Fuzzy.doc

•Дубликат

[die] Größenbeziehung, [die] Größenbeziehungen

•- (RUS) отношения величин

•Неправильная кодировка

[die]  $\alpha$ -Niveau-Menge, [die]  $\alpha$ -Niveaumenge, [die]  $\alpha$ -Niveau-Mengen, [die]

$\alpha$ -Niveaumengen

- (RUS) множество уровня альфа

•- (ENG)  $\alpha$ - level set

#### Отчет по загруженным словарям:

№ п/п	Раздел	Словарные записи		Словосочетания
1	Fuzzy	268	952	
2	Logik	225	723	
3	Mathematik	96	384	
4	KI	5	15	
5	LT	64	207	
6	Computer	2	6	
<b>ИТОГО: 6</b>		<b>660</b>	<b>2287</b>	

Проведенный анализ позволяет выявить наиболее актуальные и перспективные в научном плане направления дальнейших исследований, которые могли бы способствовать более углубленному изучению терминологии в целом.

Следующей задачей является создание переводчика-подстрочника, который можно подключать к уже имеющимся электронным словарям (программа уже работает как хороший отраслевой словарь). А на основе проведенных нами статистических исследований подязыка нечеткой логики, мы можем создать программу, которая определяла бы принадлежность переводимого текста именно к данной научной области исследования. Т.е. определенное количество слов подязыка нечеткой логики, обнаруженных электронным переводчиком в тексте, служит ему сигналом, что текст относится к определенной научной области и следует подключать словарь именно этого подязыка.



### **6.3. Задача моделирования доходов населения региона**

В связи с экономическими и социальными трансформациями, происходящими в российском обществе, большой практический и научный интерес представляет исследование доходов населения как фактора, который наиболее полно характеризует масштабы, эффективность и направленность происходящих преобразований. В данной работе разработана модель оценки уровня жизни населения через призму домохозяйств, как основных структурных ячеек общества.

Рассмотрим эффективность экономико-математической модели на примере конкретной задачи – определения количества необходимых финансовых средств на дотации домохозяйствам Ульяновской области для доведения их уровня жизни до уровня прожиточного минимума. Задача распределения населения по уровню доходов решается через структуризацию населения. Общая схема решения задачи выглядит следующим образом: население распределяется по домохозяйствам, для каждого домохозяйства рассчитываются бюджет прожиточного минимума и доходы. На основе разности между доходами и бюджетом прожиточного минимума строится вариационный ряд дефицитов бюджетов домохозяйств. Анализ дефицита бюджетов, анализ доходов в зависимости от входных параметров модели позволяет оценивать изменения в социальной политике, моделировать последствия изменений в тарифной политике монополистов.

С начала девяностых годов в российскую практику было введено понятие бюджета прожиточного минимума. Тем самым государство попыталось определить минимально допустимый уровень потребления для базовых групп населения. Следующим шагом была попытка обеспечить с помощью социальных дотаций уровень прожиточного минимума. В феврале 2003 года Государственной Думой во втором чтении был принят закон о социальных дотациях. Его суть заключается в доведении с помощью дотаций уровня жизни населения до уровня прожиточного минимума.

На настоящий момент остро стоит вопрос о реформировании жилищно-коммунального хозяйства, т.е. фактически в повышении оплаты услуг ЖКХ населением. Законодательно предпринимаются попытки выделения средств для возмещения низкодходной части населения стоимости услуг ЖКХ. Частичных компенсаций роста стоимости услуг других монополистов для населения не предусматривается. Тем не менее, государство должно поддерживать минимальный уровень жизни населения, а для этого необходимо оценивать уровень доходов и расходов домохозяйств. Социальные дотации необходимо увязать с доходами домохозяйств. Т.е. возникает задача: кого дотировать (какие ячейки общества) и в каком объеме. На этот вопрос и дает ответ данная работа.

Задача формирования статей бюджетов для социальной поддержки дефицитных домохозяйств требует комплексного подхода к решению, который за-

ключается в том, чтобы связать в единое целое демографическую структуру общества, бюджет прожиточного минимума и доходы населения. Именно построение такой модели делает данную тематику актуальной.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Проанализировать существующие модели классификации населения по группам.
2. Адаптировать существующие или создать собственную модель классификации населения.
3. Распределить население по группам в зависимости от классифицирующих признаков.
4. Исследовать существующие модели распределения населения по доходам.
5. Адаптировать существующие модели для решения задачи распределения населения по уровню доходов.
6. Построить вариационный ряд дефицитов бюджетов с целью определения социальных дотаций.

Разработать программное средство для моделирования распределения населения на типы домохозяйств, прогнозирования дефицита бюджета и расчета прожиточного минимума домохозяйств региона.

### **Библиографический список**

1. С. В. Гринев Введение в терминоведение. М., 1993. -309 с.
2. Ивина Л.В. Лингво-когнитивные основы анализа отраслевых терминосистем (на примере англоязычной терминологии венчурного финансирования): Учебно-методическое пособие – М.: Академический Проект, 2003. – 304 с.
3. Уткина Ю.Э. Лексико-семантическое моделирование английской терминосистемы «Очистка природных и сточных вод» и вопросы разработки англо-русского словаря отрасли: Автореф. дис. канд. филол. наук. – Л., 1998 – 16 с.

### **ЗАКЛЮЧЕНИЕ**

В сегодняшние дни рассмотренные технологии "вычислительного интеллекта" являются успешными прикладными технологиями. С их помощью решаются задачи распознавания образов и сцен, управления сложными объектами, в том числе мобильными роботами. Прикладные исследования смещаются от исследовательских образцов интеллектуальных программных систем в область промышленных применений. Основные активно исследуемыми направлениями становятся три области:

- Нечеткий контроль и роботика, в том числе мобильные роботы;

- Интернет-приложения для кластеризации сайтов и организации поисковых машин;

- Нечеткие базы данных, в том числе построенные на основе грубых множеств Ж. Павлака или мультимножествах (бэгах).

Многие прикладные исследования представлены правительственными лабораториями (мобильные роботы) или крупными фирмами и выходят за рамки возможностей университетских исследовательских групп.

Научное издание

**Прикладные интеллектуальные системы,  
основанные на мягких вычислениях**

Редактор: Н. А. Евдокимова

Подписано в печать: 30.11.2005

Формат 60×84/16. Бумага писчая. Печать трафаретная. Усл. печ. л. 8,00

Уч.-изд. л. 8,84. Гарнитура Computer Modern.

Тираж 200 экз. Заказ ...

Ульяновский государственный технический университет  
432027, Ульяновск, Сев. Венец, 32.

Типография УлГТУ, 432027, Ульяновск, Сев. Венец, 32.