

**Ставропольский филиал
ГОУ ВПО «Московский государственный гуманитарный университет
имени М.А. Шолохова»**

С. И. МАКАРЕНКО

Интеллектуальные информационные системы

Учебное пособие

Ставрополь
СФ МГГУ им. М. А. Шолохова
2009

УДК 004.8
ББК 32.965
М15

Макаренко С. И. Интеллектуальные информационные системы: учебное пособие. – Ставрополь: СФ МГГУ им. М. А. Шолохова, 2009. – 206 с.: ил.

Рецензенты:

доцент кафедры прикладной информатики и математики Ставропольского филиала Московского государственного гуманитарного университета имени М. А. Шолохова кандидат технических наук, доцент Федосеев В. Е.,

доцент кафедры прикладной информатики и математики Ставропольского филиала Московского государственного гуманитарного университета имени М. А. Шолохова кандидат технических наук Дятлов Д. В.

Учебное пособие адресовано студентам, обучающимся по специальности 080801 (351400) «Прикладная информатика в экономике» изучающих дисциплину «Интеллектуальные информационные системы», а также может быть использовано специалистами в области проектирования и организации интеллектуальных информационных систем.

Утверждено на заседании кафедры прикладной информатики и математики Ставропольского филиала Московского государственного гуманитарного университета имени М. А. Шолохова в качестве методического пособия для студентов по специальности 080801 (351400) - «Прикладная информатика в экономике».

Оглавление

Список сокращений	8
Введение	9
ЧАСТЬ 1. ВВЕДЕНИЕ В ИНТЕЛЛЕКТУАЛЬНЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ.....	11
1. ПОНЯТИЕ ИНТЕЛЛЕКТУАЛЬНОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ	11
1.1 Роль интеллектуальных информационных систем в современном мире	11
1.2 История исследований в области искусственного интеллекта и основные понятия в данной области	12
1.3 Интеллектуальная информационная система и ее основные свойства	16
1.4 Классификация интеллектуальных информационных систем	21
1.5 Примеры интеллектуальных информационных систем	22
2. ОСОБЕННОСТИ ПОСТРОЕНИЯ СИСТЕМ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА	24
2.1 Формулировка концепции создания искусственного интеллекта.....	24
2.2 Определение систем искусственного интеллекта	25
2.3 Информационная модель реакции систем искусственного интеллекта на воздействия окружающей среды	28
2.4 Жизненный цикл системы искусственного интеллекта и критерии перехода между этапами этого цикла	33
3. КЛАССИФИКАЦИЯ СИСТЕМ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА	35
3.1 Понятие системы искусственного интеллекта и ее место в классификации информационных систем	35
3.2 Классификация систем искусственного интеллекта	37
ЧАСТЬ 2. ЭЛЕМЕНТЫ СИСТЕМНО-КОГНИТИВНОГО АНАЛИЗА	40
4. СИСТЕМНО-КОГНИТИВНЫЙ АНАЛИЗ	40
4.1 Основные понятия когнитивной теории.....	40
4.2 Концепция системно-когнитивного анализа	41
4.2.1 Базовая когнитивная концепция	41
4.2.2 Когнитивная концепция в свободном изложении	42
4.2.3 Когнитивная концепция в формальном изложении.....	47
4.3 Когнитивное моделирование.....	49
5. ПРЕДСТАВЛЕНИЕ И ОБРАБОТКА ДАННЫХ В РАМКАХ ТЕОРИИ СИСТЕМНО-КОГНИТИВНОГО АНАЛИЗА	51
5.1 Понятия «данные», «информация», «знания».....	51

5.2 Концепция смысла Шенка-Абельсона.....	52
5.3 Диалектика «Структура – свойство – отношение» в рамках когнитивной теории.....	53
5.4 Понятия «факт», «смысл», «мысль» в рамках когнитивной теории.....	53
5.5 Иерархия задач обработки данных: «мониторинг», «анализ», «прогнозирование», «управление» в рамках когнитивной теории.....	54
6. КОГНИТИВНАЯ СТРУКТУРИЗАЦИЯ ЗНАНИЙ ОБ ИССЛЕДУЕМОМ ОБЪЕКТЕ И ВНЕШНЕЙ СРЕДЫ	56
6.1 Когнитивная структуризация знаний об исследуемом объекте и внешней среды на основе PEST-анализа.....	56
6.2 Ситуационный анализ проблем на базе SWOT-анализа.....	57
6.3 Этапы когнитивной технологии.....	58
ЧАСТЬ 3. ПРЕДСТАВЛЕНИЕ И ВЫВОД ЗНАНИЙ	61
7. МОДЕЛИ ПРЕДСТАВЛЕНИЯ ЗНАНИЙ.....	61
7.1 Декларативные и процедурные знания.....	61
7.2 Логическая модель представления знаний.....	61
7.3 Псевдофизические модели представления знаний	65
7.4 Сетевая модель представления знаний.....	67
7.5 Фреймовая модель представления знаний	71
7.6 Продукционная форма представления знаний.....	73
8. МЕТОДЫ ПРИОБРЕТЕНИЯ И ИЗВЛЕЧЕНИЯ ЗНАНИЙ	76
8.1 Основные термины и определения в области приобретения знаний.....	76
8.2 Методы приобретения знаний.....	77
8.3 Методы извлечения знаний из данных.....	80
8.4 Методы получения экспертных знаний.....	81
8.5 Методы формирования знаний	82
9. НЕЧЕТКИЙ ВЫВОД ЗНАНИЙ	84
9.1 Основные положения нечеткого вывода знаний.....	84
9.2 Типы неточного вывода.....	85
10. ИЗВЛЕЧЕНИЕ ЗНАНИЙ ИЗ ДАННЫХ МЕТОДАМИ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА ДАННЫХ	87
10.1 Особенности систем интеллектуального анализа данных.....	87
10.2 Типы закономерностей, выявляемых методами интеллектуального анализа данных.....	90
10.3 Этапы функционирования типовой системы интеллектуального анализа данных	92
10.4 Пример функционирования системы интеллектуального анализа данных	93

ЧАСТЬ 4. ПРИНЦИПЫ ПОСТРОЕНИЯ И ФУНКЦИОНИРОВАНИЯ ПРИКЛАДНЫХ СИСТЕМ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА.....	96
11. ЭКСПЕРТНЫЕ СИСТЕМЫ	96
11.1 Экспертные системы: базовые понятия.....	96
11.2 Классификация экспертных систем	98
11.3 Составные части экспертной системы и порядок ее функционирования	99
11.4 Функционирование базы знаний экспертной системы	101
11.4.1 Обратный метод логического дедуктивного вывода.....	102
11.4.2 Прямой метод логического дедуктивного вывода.....	103
12. ЭТАПЫ ПРОЕКТИРОВАНИЯ ЭКСПЕРТНОЙ СИСТЕМЫ	105
12.1 Этап идентификации.....	105
12.2 Этап концептуализации	106
12.2.1 Атрибутивный подход к построению модели предметной области	107
12.2.2 Структурный (когнитивный) подход к построению модели предметной области	107
12.2.2.1 Понятия предметной области	107
12.2.2.2 Взаимосвязи между понятиями предметной области	108
12.2.2.3 Интерпретация предметной области	108
12.2.2.4 Установление семантических отношений между понятиями предметной области	109
12.3 Этап формализации.....	109
12.4 Этап выполнения.....	109
12.5 Этап тестирования	110
12.6 Этап опытной эксплуатации.....	111
13. ПРИМЕРЫ ПОСТРОЕНИЯ ЭКСПЕРТНЫХ СИСТЕМ	113
13.1 Пример построения экспертных диагностических систем.....	113
13.2 Пример ЭС, основанной на правилах логического вывода и действующую в обратном порядке	115
14. ОСНОВЫ ЯЗЫКА ПРОГРАММИРОВАНИЯ ПРОЛОГ	119
14.1 Синтаксис	119
14.1.1 Термы	119
14.1.2 Константы	120
14.1.2.1 Атомы.....	120
14.1.2.2 Числа	120
14.1.3 Переменные.....	120
14.1.3.1 Область действия переменных	121
14.1.4 Сложные термы, или структуры	121
14.1.5 Синтаксис операторов	122
14.1.6 Синтаксис списков.....	122

14.1.7 Синтаксис строк.....	122
14.2 Утверждения	122
14.3 Запросы.....	123
14.4 Ввод программ	124
14.5 Унификация.....	125
14.6 Выражения	127
14.6.1 Арифметические выражения.....	128
14.6.2 Арифметические операторы	128
14.6.3 Вычисление арифметических выражений.....	129
14.6.4 Сравнение результатов арифметических выражений.....	130
14.7 Структуры данных	131
14.7.1 Списки	131
14.7.2 Стандартные функции обработки списков.....	132
14.7.3 Сложение многочленов	135
15. НЕЙРОННЫЕ СЕТИ.....	139
15.1 Проблемы решаемые нейронными сетями.....	139
15.2 Биологический нейрон и формальная модель нейрона Маккалоки и Питтса	140
15.3 Активационная функция нейрона.....	142
15.4 Простейшая нейронная сеть	143
15.5 Однослойная нейронная сеть и персептрон Розенблата.....	143
15.6 Машинное обучение нейронной сети на примерах	145
15.6.1 Обучение на примерах	145
15.6.2 Решение задач классификации и линейного разделения множеств	149
15.6.3 Проблемы обучения нейронной сети	150
15.6.4 Пример решения задачи нейроном	151
15.7 Классификация нейронных сетей	152
16. МНОГОСЛОЙНЫЕ НЕЙРОННЫЕ СЕТИ	156
16.1 Многослойный персептрон	156
16.2 Модель Хопфилда.....	157
16.3 Когнитрон и неоккогнитрон Фукушимы.....	160
16.4 Модель нелокального нейрона.....	160
16.5 Динамические нейронные сети	162
16.6 Проблемы развития нейронных сетей	162
16.7 Нейрокомпьютеры, нейропроцессоры, нейропакеты	163
17. ТЕОРИЯ НЕЧЕТКИХ МНОЖЕСТВ.....	165
17.1 Понятие нечеткого множества	165
17.2 Возможности применения теории нечетких множеств для описания различных видов неопределенности	168
17.3 Операции над нечеткими множествами	173
17.3.1 Равные F-множества	173

17.3.2 F-подмножества	173
17.3.3 Объединение F-множеств.....	174
17.3.4 Пересечение F-множеств.....	175
17.3.5 Особенности операций пересечения и объединения F- множеств	175
17.3.6 Разность и дополнение F-множеств.....	177
17.3.7 Другие отношения F-множеств.....	178
17.4 Понятие о нечеткой логике	178
18. ГЕНЕТИЧЕСКИЕ АЛГОРИТМЫ	181
18.1 Основные понятия, принципы и предпосылки генетических алгоритмов	181
18.2 Принцип функционирования генетического алгоритма.....	183
18.2.1 Алгоритм функционирования простейшего генетического алгоритма	187
18.2.2 Репродуктивный план Холланда, как пример реализации генетического алгоритма	191
18.3 Достоинства и недостатки генетических алгоритмов.....	193
18.4 Примеры применения генетических алгоритмов.....	194
19. ОБРАБОТКА ЕСТЕСТВЕННОГО ЯЗЫКА В ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМАХ.....	195
19.1 Основные понятия о системах, использующих естественный язык	195
19.2 Технологии анализа естественного языка	197
19.2.1 Подбор шаблона.....	197
19.2.2 Синтаксический анализ	198
19.2.3 Семантические грамматики	200
19.2.4 Анализ с помощью падежных фреймов	201
19.2.5 Обработка предложений естественного языка с помощью нейронных сетей.....	204
Заключение	205
Список использованных источников	206

Список сокращений

ERP	- Enterprise Resource Planing – система промышленного планирования ресурсов
KLL	- knowledge level learning - обучение на уровне знаний
SLL	- symbol level learning - обучение на символьном уровне
БД	- база данных
БЗ	- база знаний
БНФ	- бэкусовская нормальная форма
ГА	- генетический алгоритм
ЕЯ	- естественный язык
ИАД	- интеллектуальный анализ данных
ИС	- информационная система
ИИС	- интеллектуальная информационная система
ИНС	- искусственная нейронная сеть
ИО	- информационный образ
НС	- нейронная сеть
ОД	- объект диагностирования
ПО	- предметная область
ПО	- программное обеспечение
СЕЯИ	- система естественно-языкового интерфейса
СИИ	- система искусственного интеллекта
СКНФ	- совершенная конъюнктивная нормальная форма
СППР	- система поддержки принятия решений
СУБД	- система управления базами данных
ТНМ	- теория нечетких множеств
ТНЛ	- теория нечеткой логики
ТРИЗ	- технология решения изобретательских задач
ЭДС	- экспертно-диагностическая система
ЭС	- экспертная система

Введение

Учебное пособие написано по опыту преподавания автором дисциплины «Интеллектуальные информационные системы» на кафедре прикладной информатики и математики Ставропольского филиала Московского государственного гуманитарного университета имени М. А. Шолохова и в первую очередь адресовано студентам, обучающимся по специальности «Прикладная информатика в экономике». Также учебное пособие может быть использовано специалистами в области проектирования и организации интеллектуальных информационных систем.

Учебная дисциплина «Интеллектуальные информационные системы» изучается студентами вузов, заинтересованных в применении интеллектуальных информационных систем в экономике. Предлагаемая дисциплина дает представление студентам о состоянии разработки и тенденциях развития информационных систем, их различных приложениях. Студенты смогут сориентироваться, какие именно модели и методы современных интеллектуальных информационных систем могут быть использованы при решении тех или иных задач экономического анализа и принятия решений. Цель изучения дисциплины — дать слушателям комплекс ориентирующих знаний по основным понятиям интеллектуальных информационных систем и возможностям их использования в различных областях применения в сфере экономики.

Учебное пособие учитывает требования государственного образовательного стандарта и структурно соответствует учебной программе и тематическому плану изучения дисциплины «Интеллектуальные информационные системы» [4]. Отдельные части пособия соответствуют темам дисциплины, а отдельные главы — лекционным занятиям. Дополнительно глава 14 может быть использована для проведения практических занятий по технологии проектирования экспертных систем и самостоятельно изучения студентами соответствующего материала с использованием ПК.

При написании пособия автор придерживался принципа необходимости дополнения общетеоретических и концептуальных основ интеллектуальных информационных систем, изучение которых предусмотрено государственным образовательным стандартом, дополнительными сведениями в данной сфере.

Кроме того, пособие может быть использовано в качестве конспекта лекций, так как в тексте пособия материал, рекомендуемый к конспектированию на лекциях, выделен *курсивом*.

При составлении учебного пособия автор ориентировался на известные учебные материалы в предметной области, а также использовал ресурсы сети Internet посвященные вопросам интеллектуальных информационных систем и систем искусственного интеллекта.

В основу глав учебного пособия был положен материал следующих источников:

- глава 1 – учебные пособия [1, 2, 3];
- главы 2-6 – учебное пособие [1];
- глава 7 – учебное пособие [3];
- глава 8 – учебные пособия [2, 3], справочное пособие [5];
- глава 9 – учебное пособие [2], справочное пособие [5];
- глава 10 – учебное пособие [2];
- главы 11-13 – учебное пособие [1], работа [6], учебные курсы [7, 8];
- глава 14 – учебный курс [7];
- главы 15-16 – учебные пособия [1, 3] и работа [9], дополненные материалом работ [10, 11, 12];
- глава 17 – монография [13], дополненная материалом работ [14, 15, 16, 17];
- глава 18 – обобщение работ [18, 19, 20, 21];
- глава 19 – учебное пособие [3].

Таким образом, литература [1-3] составляет основную литературу по дисциплине и рекомендуется к изучению при освоении материала дисциплины. Источники [5, 6-17] составляют дополнительную литературу по дисциплине и рекомендуются для более углубленного изучения соответствующих отдельных тем.

Автор выражает благодарность рецензентам за кропотливый труд по поиску ошибок и неточностей, а также ценные замечания, которые помогли сделать материал пособия лучше и доступнее.

Предложения и замечания по учебному пособию автор просит направлять на email: mak-serg@yandex.ru.

ЧАСТЬ 1. ВВЕДЕНИЕ В ИНТЕЛЛЕКТУАЛЬНЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ

1. ПОНЯТИЕ ИНТЕЛЛЕКТУАЛЬНОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ

*«Системы искусственного интеллекта позволяют с успехом решать
сложнейшие проблемы, которых до создания этих систем не возникало»
/Из компьютерного фольклора/*

1.1 Роль интеллектуальных информационных систем в современном мире

Прогресс в сфере экономики немыслим без применения современных информационных технологий, представляющих собой основу экономических *информационных систем* (ИС). Информационные системы в экономике имеют дело с организацией и эффективной обработкой больших массивов данных в компьютеризированных системах предприятий, обеспечивая информационную поддержку принятия решений менеджерами. Глобализация финансовых рынков, развитие средств электронной коммерции и формирование в Интернете доступных для анализа баз данных финансово-экономической информации, снижение стоимости программной реализации ИС, привели за последние два года к беспрецедентному росту их использования в экономике. ИС позволяют объективно оценить достигнутый уровень развития экономики, выявить резервы и обеспечить успех их деятельности на основе применения правильных решений.

Работы в области искусственного интеллекта в течение довольно длительного времени представлялись многим как причуды оторванных от реальности информантов-интеллектуалов, обучающих компьютер игре в шахматы или распознаванию сцен, или же пытающихся создать автономно ориентирующиеся в пространстве мобильные роботы.

Появление экспертных систем MYCIN, DENDRAL, PROSPECTOR, а так же обнадеживающие результаты их успешного применения в области медицины, технической диагностики, геофизики, управления непрерывными технологическими процессами решительно изменили ситуацию. Стало очевидным, что методы правдоподобных и дедуктивных выводов могут быть хорошим дополнением или частичной заменой специалиста, ставящего медицинский или технический диагноз и вообще принимающего решения в форме выбора одной из альтернативных гипотез на основании наблюдаемых данных.

Эти успехи стимулировали применение технологий и методов искусственного интеллекта в самых разных отраслях экономики, в первую очередь, для анализа и диагностирования эффективности экономической

деятельности предприятий, выбора эффективной стратегии поведения трейдера на рынке ценных бумаг, выбора оптимальных вариантов инвестиционных проектов в условиях неопределенности и при наличии трудно формализуемых факторов. Первые экспертные системы были оторваны от корпоративных информационных систем и строились как самостоятельные программы, имели собственную организацию хранения данных и знаний. Поэтому их применение к реальным проблемам в сфере экономики первоначально не дало ожидаемого результата. Возникли проблемы, связанные с высокой трудоемкостью создания и реорганизации базы знаний традиционными методами интервьюирования экспертов, а также с загрузкой, хранением и актуализацией больших объемов данных, на которые не были рассчитаны эти экспертные системы [1, 2].

Новая волна и значительный эффект от применения технологии искусственного интеллекта получены в результате разработки и применения интеллектуальных информационных систем, явившихся синтезом экспертных и информационных систем. Создание ИИС стало естественным продолжением широкого применения информационных систем классического типа. Системы реинжиниринга бизнес-процессов показали возможность упорядочения информационных потоков и совершенствования структуры предприятия при внедрении информационных технологий, помогли освоить методологию разработки информационной модели предприятия. Интегрированные ИС предприятия обеспечивают информационную поддержку всех производственных процессов и служб предприятия, включая проектирование, изготовление и сбыт продукции, финансово-экономический анализ, планирование, управление персоналом, маркетинг, сопровождение эксплуатации изделий, перспективное планирование. Внедрение информационных систем типа ERP (Enterprise Resource Planing) увеличивает эффективность работы предприятия на 20-30%. В результате появились полностью компьютеризованные информационно-технологические связи между корпорациями (системы B2B или бизнес-бизнес) и связи корпорации с клиентами (системы B2C или системы бизнес-клиент).

1.2 История исследований в области искусственного интеллекта и основные понятия в данной области

Зарождение исследований в области искусственного интеллекта (ИИ). Два направления: логическое и нейрокибернетическое. Ранние исследования в 50-60-е годы (Н. Винер, Тьюринг, Мак-Каллок, Розенблатт, Саймон, Маккарти, Слэйджл, Сэмюэль, Гелернер, Н. Амосов). Появление первого развитого языка программирования LISP для построения систем ИИ. Появление в конце 60-х годов интегральных (интеллектуальных) роботов и первых экспертных систем. Успехи экспертных систем и застой в нейрокибернетике в 70-е годы. Новый бум нейрокибернетики в начале 80-х годов (модель Хопфилда). Появление логического программирования и

языка PROLOG. Программа создания ЭВМ 5-го поколения. Стратегическая компьютерная инициатива США. Исследования по ИИ в СССР и России.

С самого начала исследований *в области моделирования процесса мышления* (конец 40-х годов) выделились два до недавнего времени *практически независимых направления* [3]:

- *логическое,*
- *нейрокибернетическое.*

Первое было основано на выявлении и применении в интеллектуальных системах различных логических и эмпирических приемов (эвристик), которые применяет человек для решения каких-либо задач. В дальнейшем с появлением концепций «экспертных систем» (ЭС) (в начале 80-х годов) это направление вылилось в научно-технологическое направление информатики «инженерия знаний», занимающееся созданием т. н. «систем, основанных на знаниях» (Knowledge Based Systems). Именно с этим направлением обычно ассоциируется термин «искусственный интеллект» (ИИ).

Второе направление – нейрокибернетическое – было основано на построении самоорганизующихся систем, состоящих из множества элементов, функционально подобных нейронам головного мозга. Это направление началось с концепции формального нейрона Мак-Каллока-Питтса и исследований Розенблатта с различными моделями перцептрона – системы, обучающейся распознаванию образов. В связи с относительными успехами в логическом направлении ИИ и низким технологическом уровнем в микроэлектронике нейрокибернетическое направление было почти забыто с конца 60-х годов до начала 80-х, когда появились новые удачные теоретические модели (например, «модель Хопфилда») и сверхбольшие интегральные схемы.

Логическое направление можно рассматривать как моделирование мышления на уровне сознания или вербального или логического (целенаправленного) мышления. Его достоинствами являются [3]:

- возможность относительно легкого понимания работы системы;
- легкость отображения процесса рассуждений системы на ее интерфейс с пользователем на естественном языке или каком-либо формальном языке;
- достижимость однозначности поведения системы в одинаковых ситуациях.

Недостатками логического подхода являются [3]:

- трудность и неестественность реализации нечетких знаков (образов);

- трудность (или даже невозможность) реализации адекватного поведения в условиях неопределенности (недостаточности знаний, зашумленности данных, не точно поставленной цели и т.п.);
- трудность и неэффективность распараллеливания процесса решения задач.

Нейрокибернетическое направление (или нейроинформатика) можно рассматривать как моделирование образного мышления и мышления на подсознательном уровне (моделирование интуиции, творческого воображения, инсайта). Его достоинства – это отсутствие недостатков, свойственных логическому направлению, а недостатки – отсутствие его достоинств. Кроме того, в нейрокибернетическом направлении привлекает возможность (быть может, иллюзорная), задав базовые весьма простые алгоритмы адаптации и особенности структуры искусственной нейронной сети, получить систему, настраивающуюся на поведение сколь угодно сложное и адекватное решаемой задаче. Причем его сложность зависит только от количественных факторов модели нейронной сети. Еще одним достоинством в случае аппаратной реализации нейронной сети является ее живучесть, т.е. способность сохранять приемлемую эффективность решения задачи при выходе из строя элементов сети. Это свойство нейронных сетей достигается за счет избыточности. В случае программной реализации структурная избыточность нейронных сетей позволяет им успешно работать в условиях неполной или зашумленной информации.

Чем отличается понятие «знание» от понятия «данные» или «информация»? В последнее время ученые приходят к выводу, что наряду с веществом и энергией информация является объективно существующей неотъемлемой частью материального мира, характеризующей его упорядоченность (неоднородность) или структуру. Способность живых существ сохранять свою структуру (упорядоченность) в мире, где, вероятно, превалирует стремление к увеличению энтропии (однородности), обусловлена их способностью распознавать структуру окружающего мира и использовать результат распознавания (т.е. знания о мире) для целей выживания.

Таким образом, **знания** – это воспринятая живым существом (субъектом) информация из внешнего мира и в отличие от «информации знание» субъективно. Оно зависит от особенностей жизненного опыта субъекта, его истории взаимоотношения с внешней средой, т.е. от особенностей процесса его обучения или самообучения. На этом уровне абстракции знание уникально и обмен знанием между индивидуумами не может происходить без потерь в отличие от данных, в которых закодирована информация (неоднородность), и которые могут передаваться от передатчика к приемнику без потерь (не учитывая возможность искажения вследствие помех).

Знание передается между субъектами посредством какого-либо языка представления знаний, наиболее типичным представителем которого является естественный язык. Создавая и используя естественный язык, человек с одной стороны стремился в нем формализовать и унифицировать знания для того, чтобы передавать их одинаковым образом наибольшему количеству людей с разным жизненным опытом, а с другой стороны, пытался дать возможность передавать все богатство личного знания.

Первая тенденция привела к появлению различных формализованных специальных диалектов языка в различных областях знаний (математике, физике, медицине и т.д.).

Вторая привела к появлению художественной литературы, в основе которой лежит стремление средствами языка вызвать ассоциации (переживания) в мозгу человека, т.е. заставить его думать и переживать на основе знаний, почерпнутых из прочтенного, и своих собственных знаний. По большому счету все разновидности искусства направлены на это – передачу знаний с использованием ассоциаций.

Если перейти от такого высокого уровня абстракции (философского) к более приземленному, то можно сравнивать знания и данные в их формализованном виде, что обычно и делается в литературе по искусственному интеллекту. Тогда можно сформулировать следующие отличия знаний от данных [3]:

- знания более структурированы;
- в знаниях наибольшее значение имеют не атомарные элементы знаний (как в данных), а взаимосвязи между ними;
- знания более самоинтерпретируемы, чем данные, т.е. в знаниях содержится информация о том, как их использовать;
- знания активны в отличие от пассивных данных, т.е. знания могут порождать действия системы, использующей их.

Следует иметь в виду, что резкой границы между данными и знаниями нет, т.к. в последние двадцать лет разработчики систем управления базами данных все более делают их похожими на знания. Примером может служить применение семантических сетей (формализма для представления знаний) для проектирования баз данных, появление объектно-ориентированных баз данных, хранимых процедур (это делает в какой-то мере данные активными) и т.п. Таким образом, отличия знаний от данных, перечисленные выше, с развитием средств информатики сглаживаются.

В инженерии знаний различают следующие основные понятия о знаниях, заимствованные из семиотики – науки о знаковых системах:

- *экстенциональные знания* – *поверхностные или конкретные знания,*

- **интенциональные знания** – глубинные или абстрактные знания (знания о закономерностях),
- **синтаксис** – структура знаковой системы (данных или знаний),
- **семантика** – смысл знаковой системы (знаний), т.е. эквивалентное ее представление в другой парадигме представления знаний (внутренней),
- **прагматика** – цели, связанные со знаковой системой (например, цели или назначение предложения на естественном языке – команда, вопрос, пояснение и т.п.).

1.3 Интеллектуальная информационная система и ее основные свойства

Интеллектуальные информационные системы (ИИС) — естественный результат развития обычных информационных систем, сосредоточили в себе наиболее наукоемкие технологии с высоким уровнем автоматизации не только процессов подготовки информации для принятия решений, но и самих процессов выработки вариантов решений, опирающихся на полученные информационной системой данные. ИИС способны диагностировать состояние предприятия, оказывать помощь в антикризисном управлении, обеспечивать выбор оптимальных решений по стратегии развития предприятия и его инвестиционной деятельности. Благодаря наличию средств естественно-языкового интерфейса появляется возможность непосредственного применения ИИС бизнес пользователем, не владеющим языками программирования, в качестве средств поддержки процессов анализа, оценки и принятия экономических решений. ИИС применяются для экономического анализа деятельности предприятия, стратегического планирования, инвестиционного анализа, оценки рисков и формирования портфеля ценных бумаг, финансового анализа, маркетинга и т.д.

В этом случае потребовался значительно больший объем информации как собственно о предприятии, так и о его окружении, т.е. природных, политических, экономических и других факторах, конкурентах, поставщиках и т.д., а также значительно более сложные вычисления, необходимость учета слабо формализуемых факторов, высокий уровень интерфейса. Поставленные задачи реализованы в системах поддержки принятия решений. Их отличительная черта — значительно более высокий уровень «интеллекта», чем у обычных интегрированных систем управления производством; наличие специальных процедур для отбора и ввода данных, в том числе и по расписанию из различных внешних систем. В системах поддержки принятия решений производится заблаговременное вычисление (в целях обеспечения уменьшения времени реакции) агрегированных данных, часто используемых в запросах; используется специальная организация хранения данных, обеспечивающая возможность многоаспектного поиска с изменяемой

глубиной агрегирования/деагрегирования данных. Эта технология получила название хранилищ и витрин данных в сочетании с оперативной аналитической обработкой данных.

Наиболее мощные фирмы, разрабатывающие системы управления базами данных (СУБД) — ORACLE, BASE, Microsoft, — поставляют на рынок системы, в которые модули поддержки принятия решений входят как компонента. В состав таких входят технологии искусственного интеллекта — нейронные сети, интеллектуальный анализ данных. Объектно-ориентированная структура этих баз данных сделала реальностью идеологию *фреймов*, разработанную в рамках искусственного интеллекта. Технические решения, необходимые для создания полномасштабных интеллектуальных информационных систем — средства ведения баз знаний на основе объектно-ориентированных баз данных, автоматизации формирования баз знаний на основе методов интеллектуального анализа данных, полнотекстовые системы поиска и семантические анализаторы естественного языка для естественно-языкового интерфейса — стали производиться как серийно выпускаемые программные изделия. Не реализованными в рамках таких СУБД пока остаются технологии реализации правдоподобных (вероятностных) и логических (дедуктивных) выводов.

По материалам кадровых агентств, в Интернете существует устойчивый высокий спрос на специалистов, владеющих современными технологиями проектирования и разработки ИИС. Поскольку технические и программные средства изменяются достаточно быстро (их полное обновление происходит в течение 2-3 лет), а принципы работы интеллектуальных систем изменяются относительно медленно (на протяжении 15-20 лет).

Первоначально ИИС использовали знания нескольких экспертов в каждой из областей инвестиций. В настоящее время базы знаний частично формируются посредством машинного обучения, используя методы индукции, генетические алгоритмы и некоторые другие методы извлечения знаний. Менеджер, используя такую схему, теоретически может принимать решения более эффективно и с меньшей стоимостью, чем это смог бы сделать любой индивидуальный эксперт в данной области. Наиболее очевидным преимуществом интеграции некоторых форм искусственного интеллекта в процессе принятия решений по сравнению с постоянным консультированием с группой экспертов обычно является более низкая стоимость и большее соответствие результатов задаче.

В отличие от обычных аналитических и статистических моделей, ИИС позволяют получить решение трудно формализуемых слабо структурированных задач.

Возможность ИИС работать со слабоструктурированными данными подразумевает наличие следующих качеств:

- решать задачи, описанные только в терминах «мягких» моделей, когда зависимости между основными показателями являются не вполне определенными или даже неизвестными в пределах некоторого класса;***
- способность к работе с неопределенными или динамичными данными, изменяющимися в процессе обработки, позволяет использовать ИИС в условиях, когда методы обработки данных могут изменяться и уточняться по мере поступления новых данных;***
- способность к развитию системы и извлечению знаний из накопленного опыта конкретных ситуаций увеличивает мобильность и гибкость системы, позволяя ей быстро осваивать новые области применения.***

Возможность использования информации, которая явно не хранится, а выводится из имеющихся в базе данных, позволяет уменьшить объемы хранимой актуальной информации при сохранении богатства доступной пользователю информации. Направленность ИИС на решение слабоструктурированных, плохо формализуемых задач расширяет область применения ИИС.

Наличие развитых коммуникативных способностей у ИИС дает возможность пользователю выдавать задания системе и получать от нее обработанные данные и комментарии на языке, близком к естественному. Система естественно-языкового интерфейса (СЕЯИ) транслирует естественно-языковые структуры на внутри машинный уровень представления знаний. Включает морфологический, синтаксический, семантический анализ и соответственно в обратном порядке синтез. Программа интеллектуального интерфейса воспринимает сообщения пользователя и преобразует их в форму представления базы знаний и, наоборот, переводит внутреннее представление результата обработки в формат пользователя и выдает сообщение на требуемый носитель. Важнейшее требование к организации диалога пользователя с ИИС — естественность, означающая формулирование потребностей пользователя с использованием профессиональных терминов конкретной области применения.

Для ИИС характерны следующие признаки [1, 2]:

- 1. Развитые коммуникативные способности: возможность обработки произвольных запросов в диалоге на языке максимально приближенном к естественному (система естественно-языкового интерфейса — СЕЯИ);***

2. *Направленность на решение слабоструктурированных, плохо формализуемых задач (реализация мягких моделей);*
3. *Способность работать с неопределенными и динамичными данными;*
4. *Способность к развитию системы и извлечению знаний из накопленного опыта конкретных ситуаций;*
5. *Возможность получения и использования информации, которая явно не хранится, а выводится из имеющихся в базе данных;*
6. *Система имеет не только модель предметной области, но и модель самой себя, что позволяет ей определять границы своей компетентности;*
7. *Способность к аддуктивным выводам, т.е. к выводам по аналогии;*
8. *Способность объяснять свои действия, неудачи пользователя, предупреждать пользователя о некоторых ситуациях, приводящих к нарушению целостности данных.*

Наибольшее распространение ИИС получили для экономического анализа деятельности предприятия, стратегического планирования, инвестиционного анализа, оценки рисков и формирования портфеля ценных бумаг, финансового анализа, маркетинга.

Традиционно считается, что ИИС содержит [1, 2]:

1. *Базу данных,*
2. *Базу знаний,*
3. *Интерпретатор правил или машину вывода,*
4. *Компоненту объяснения и естественно языкового интерфейса, обеспечивающих связный диалог пользователя и системы с попеременным переходом инициативы.*

Отличительные особенности ИИС по сравнению с обычными ИС состоят в следующем [1, 2]:

1. *Интерфейс с пользователем на естественном языке с использованием понятий, характерных для предметной области пользователя;*
2. *Способность объяснять свои действия и подсказывать пользователю, как правильно ввести экономические показатели и как выбрать подходящие к его задаче параметры экономической модели;*
3. *Представление модели экономического объекта и его окружения в виде базы знаний и средств дедуктивных и правдоподобных выводов в сочетании с возможностью работы с неполной или неточной информацией;*

4. Способность автоматического обнаружения закономерностей бизнеса в ранее накопленных фактах и включения их в базу знаний.

Применение ИИС совместно со стандартными методами исследования операций, динамического программирования, а также с методами нечеткой логики для планирования при комплексной автоматизации деятельности предприятия, приносит принципиальные выгоды: реально снижаются операционные издержки; повышается качество управленческих решений.

Интеллектуальные информационные системы особенно эффективны в применении к слабо структурированным задачам, в которых пока отсутствует строгая формализация, и для решения которых применяются эвристические процедуры, позволяющие в большинстве случаев получить решение. Отчасти этим объясняется то, что диапазон применения ИИС необычайно широк: от управления непрерывными технологическими процессами в реальном времени до оценки последствий от нарушения условий поставки товаров по импорту.

По мере совершенствования принципов логического и правдоподобного вывода, применяемых в ИИС за счет использования нечеткой, модальной, временной логики, байесовских сетей вывода, ИИС начинают проникать в высокоинтеллектуальные области, связанные с разработкой стратегических решений по совершенствованию деятельности предприятий. Этому способствуют более современные алгоритмы анализа и синтеза предложений естественного языка, облегчающие общение пользователя с системой.

Включение в состав ИИС классических экономико-математических моделей, методов линейного, квадратичного и динамического программирования позволяет сочетать анализ объекта на основе экономических показателей с учетом факторов и рисков политических и внеэкономических факторов, оценивать последствия полученных их ИИС решений.

Наличие в составе ИИС объектно-ориентированной базы данных позволяет однородными средствами обеспечить хранение и актуализацию как фактов, так и знаний.

1.4 Классификация интеллектуальных информационных систем

Интеллектуальные информационные системы можно классифицировать по разным основаниям. Базовые основания и классификация по ним приведены ниже и на рис. 1.1 [1, 2].

1. Предметная область применения:

- ИИС менеджмента,
- ИИС риск-менеджмента,
- ИИС инвестиций
- ИИС в военной сфере и др.

2. Степень автономности от корпоративной ИС или базы данных:

- автономные в виде самостоятельных программных продуктов с собственной базой данных;
- сопрягаемые с корпоративной;
- полностью интегрированные.

3. По способу и оперативности взаимодействия с объектом:

- статические ИИС,
- динамические ИИС:
 - ИИС реального времени;
 - советующие ИИС, в контур которых вовлечен пользователь.

4. По адаптивности:

- обучаемые ИИС, т.е. системы, параметры и структура которых, могут изменяться в процессе обучения или самообучения (нейронные сети, генетические алгоритмы и др.);
- ИИС, параметры которых изменяются администратором базы знаний (экспертные системы и др.).

5. По модели представления знаний:

- методы резолюций исчисления предикатов;
- Немонотонные, модальные и временные логики;
- Марковские и Баесовские сети вывода;
- Казуальные деревья и теория веры;
- Теория Демпстера-Шейфера;
- Нечеткие системы.

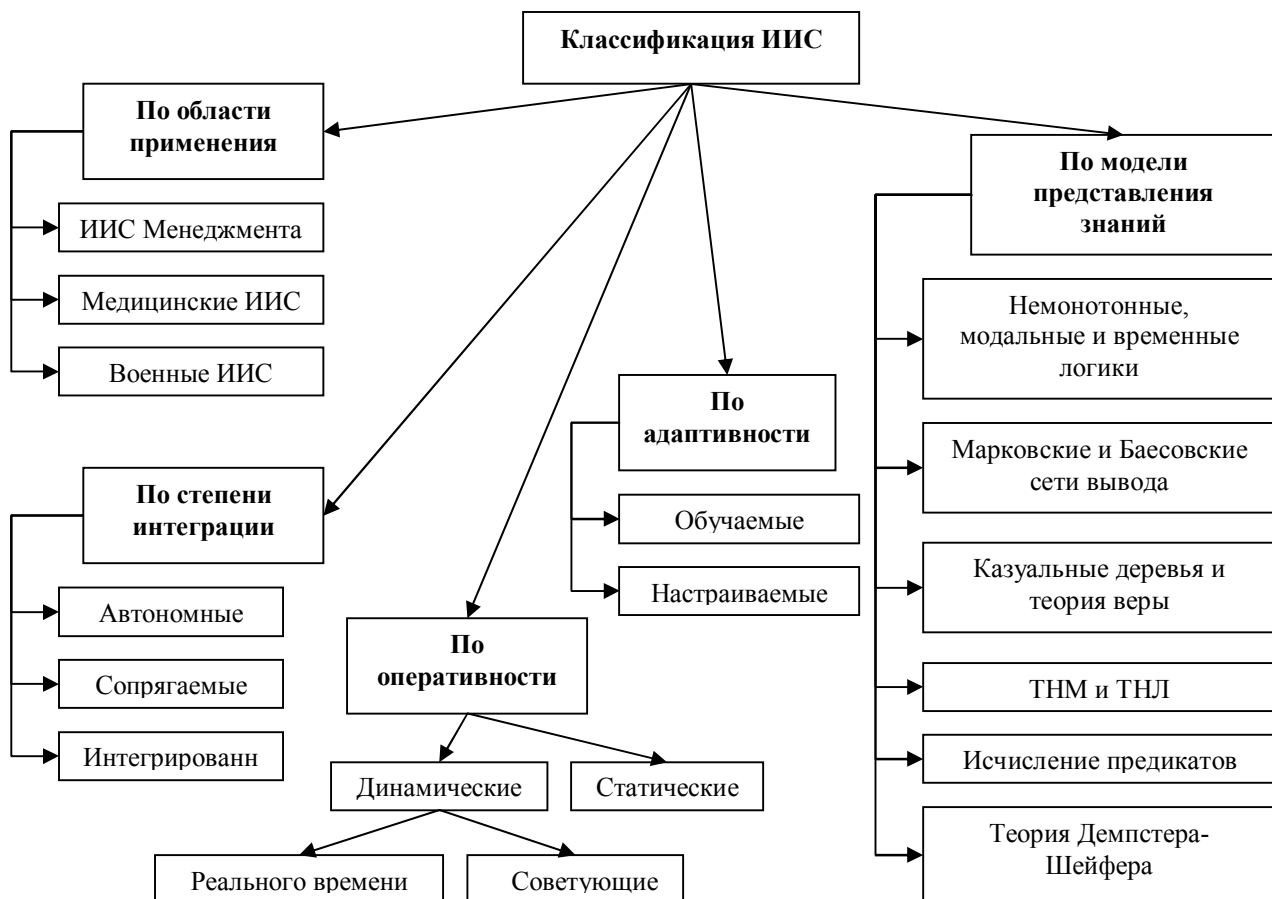


Рисунок 1.1 - Классификация интеллектуальных информационных систем

1.5 Примеры интеллектуальных информационных систем

Intelligent Hedger: основанный на знаниях подход в задачах страхования от риска от фирмы Information System Departm, New York University. Проблема огромного количества постоянно растущих альтернатив страхования от рисков, быстрое принятие решений менеджерами по рискам, а также недостаток машинной поддержки в процессе страхования от рисков предполагает различные оптимальные решения для менеджеров по риску. В данной системе разработка страхования от риска сформулирована как многоцелевая оптимизационная задача, которая включает несколько сложностей, с которыми существующие технические решения не справляются.

Система рассуждений в прогнозировании обмена валют. Фирма: Department of Computer Science City Polytechnic University of Hong Kong. Представляет новый подход в прогнозировании курсов валют, основанный на аккумуляции и рассуждениях с поддержкой признаков, присутствующих для фокусирования на наборе гипотез о движении обменных курсов. Представленный в прогнозирующей системе *набор признаков* — это

заданный набор экономических значений и различные наборы изменяющихся во времени параметров, используемых в модели прогнозирования. Краткие характеристики: математическая основа примененного подхода базируется на теории Демпстера—Шейфера.

Nereid: Система поддержки принятия решений для оптимизации работы с валютными опционами. Фирма: NTT Data, The Tokai Bank, Science University of Tokyo. Система облегчает дилерскую поддержку для оптимального ответа как один из возможных представленных вариантов. Краткие характеристики: система разработана с использованием фреймовой системы CLP, которая легко интегрирует финансовую область в приложение ИИ. Предложен смешанный тип оптимизации, сочетающий эвристические знания с техникой линейного программирования. Система работает на Sun-станциях.

PMIDSS: Система поддержки принятия решений при управлении портфелем. Разработчики: Финансовая группа Нью-Йоркского университета. Решаемые задачи: выбор портфеля ценных бумаг; долгосрочное планирование инвестиций. Краткие характеристики: смешанная система представления знаний, использование разнообразных механизмов вывода: логика, направленные семантические сети, фреймы, правила.

В основу материала главы 1 положены работы [1, 2, 3].

2. ОСОБЕННОСТИ ПОСТРОЕНИЯ СИСТЕМ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

2.1 Формулировка концепции создания искусственного интеллекта

Основной замысел по созданию искусственного интеллекта состоит в следующем [1]:

- во-первых, выявить основные моменты играющие существенную роль при создании **естественного** интеллекта;
- во-вторых, попробовать реализовать эти моменты на базе современных компьютерных технологий.

Наблюдения за системами естественного интеллекта позволяют сформулировать следующую гипотезу [1].

1. **Естественный** интеллект реально существует.
2. **Естественный** интеллект создается не мгновенно, а в течение довольно длительного времени по вполне определенной технологии, которая включает три основных этапа:
 - Создание материальной системы поддержки естественного интеллекта по сложной технологии в изолированных от среды условиях;
 - Создание активной информационной структуры, базирующейся на материальной системе поддержки, способной к развитию и саморазвитию в систему естественного интеллекта, т.е. создание системы потенциального естественного интеллекта (оболочки, инструментальной системы);
 - Формирование структуры и функций естественного интеллекта во взаимодействии системы его поддержки с другими подобными системами и с окружающей средой, как с природной, так и с «социальной», т.е. созданной другими подобными системами, в результате чего происходит трансформация системы потенциального естественного интеллекта в систему реального естественного интеллекта.
3. Системы искусственного интеллекта (СИИ) полностью функционально эквивалентные естественному интеллекту могут быть созданы на базе другой материальной структуры системы поддержки системы и другой системы потенциального искусственного интеллекта.

4. *Создание СИИ должно включать три этапа:*

- *Создание материальной системы поддержки (эта проблема в основном решена, т.к. СИИ могут создаваться даже на базе современных персональных компьютеров);*
- *Создание системы потенциального искусственного интеллекта, т.е. программной оболочки, инструментальной системы (таких систем в настоящее время существует пока еще очень мало);*
- *Обучение и самообучение системы потенциального искусственного интеллекта и преобразование ее в реальную СИИ.*

5. *Основную роль в создании системы потенциального искусственного интеллекта играет разработка научной концепции, отражающей способы реализации естественного интеллекта и пути его трансформации из потенциального в реальный.*

2.2 Определение систем искусственного интеллекта

В 1950 году в статье «Вычислительные машины и разум» (Computing machinery and intelligence) выдающийся английский математик и философ Алан Тьюринг предложил тест, чтобы заменить бессмысленный, по его мнению, вопрос «может ли машина мыслить?» на более определённый.

Вместо того, чтобы отвлеченно спорить о критериях, позволяющих отличить живое мыслящее существо от машины, выглядящей как живая и мыслящая, он предложил реализуемый на практике способ установить это.

Тест Тьюринга: *судья-человек ограниченное время (например, 5 минут) переписывается в чате (в оригинале – по телеграфу) на естественном языке с двумя собеседниками, один из которых – человек, а другой – компьютер. Если судья за предоставленное время не сможет надёжно определить, кто есть кто, то компьютер прошёл тест Тьюринга [1].*

Предполагается, что каждый из собеседников стремится, чтобы человеком признали его. С целью сделать тест простым и универсальным, переписка сводится к обмену текстовыми сообщениями.

Переписка должна производиться через контролируемые промежутки времени, чтобы судья не мог делать заключения исходя из скорости ответов. (Тьюринг ввел это правило потому, что в его времена компьютеры реагировали гораздо медленнее человека. Сегодня же это правило необходимо, наоборот, потому что они реагируют гораздо быстрее, чем человек).

Идею Тьюринга поддержал Джо Вайзенбаум, написавший в 1966 году первую «беседующую» программу «Элиза». Программа всего в 200 строк

лишь повторяла фразы собеседника в форме вопросов и составляла новые фразы из уже использованных в беседе слов. Тем ни менее этого оказалось достаточно, чтобы поразить воображение тысяч людей.

Тьюринг считал, что компьютеры в конечном счёте пройдут его тест, т.е. на вопрос: «Может ли машина мыслить?» он отвечал утвердительно, но в будущем времени: «Да, смогут!»

Алан Тьюринг был не только выдающимся ученым, но и настоящим пророком компьютерной эры. Достаточно сказать, что в 1950 году (!!!), когда он писал, что к 2000 году, на столе у миллионов людей будут стоять компьютеры, имеющие оперативную память 1 миллиард бит (около 119 Мб) и оказался в этом абсолютно прав. Когда он писал это, все компьютеры мира вместе взятые едва ли имели такую память. Он также предсказал, что обучение будет играть важную роль в создании мощных интеллектуальных систем, что сегодня совершенно очевидно для всех специалистов по СИИ. Вот его слова: «Пытаясь имитировать интеллект взрослого человека, мы вынуждены много размышлять о том процессе, в результате которого человеческий мозг достиг своего настоящего состояния... Почему бы нам вместо того, чтобы пытаться создать программу, имитирующую интеллект взрослого человека, не попытаться создать программу, которая имитировала бы интеллект ребенка? Ведь если интеллект ребенка получает соответствующее воспитание, он становится интеллектом взрослого человека... Наш расчет состоит в том, что устройство, ему подобное, может быть легко запрограммировано... Таким образом, мы расчленим нашу проблему на две части: на задачу построения «программы-ребенка» и задачу «воспитания» этой программы» [1].

Именно этот путь и используют практически все системы ИИ. Кроме того, именно на этом пути появляются и другие признаки интеллектуальной деятельности: накопление опыта, адаптация и т. д.

Против теста Тьюринга было выдвинуто несколько возражений [1].

1. Машина, прошедшая тест, может не быть разумной, а просто следовать какому-то хитроумному набору правил. На что Тьюринг не без юмора отвечал: "А откуда мы знаем, что человек, который искренне считает, что он мыслит, на самом деле не следует какому-то хитроумному набору правил?"
2. Машина может быть разумной и не умея разговаривать, как человек, ведь и не все люди, которым мы не отказываем в разумности, умеют писать. Могут быть разработаны варианты теста Тьюринга для неграмотных машин и судей.
3. Если тест Тьюринга и проверяет наличие разума, то он не проверяет сознание (consciousness) и свободу воли (intentionality), тем самым

не улавливая весьма существенных различий между разумными людьми и разумными машинами.

Сегодня уже существуют многочисленные варианты интеллектуальных систем, которые не имеют цели, но имеют критерии поведения: генетические алгоритмы и имитационное моделирование эволюции. Поведение этих систем выглядит таким образом, как будто они имеют различные цели и добиваются их.

Ежегодно производится соревнование между разговаривающими программами, и наиболее человекоподобной, по мнению судей, присуждается приз Лебнера (Loebner).

Существует также приз для программы, которая, по мнению судей, пройдёт тест Тьюринга. Этот приз ещё ни разу не присуждался.

В заключение отметим, что и сегодня тест Тьюринга не потерял своей фундаментальности и актуальности, более того – приобрел новое звучание в связи с возникновением Internet, общением людей в чатах и на форумах под условными никами и появлением почтовых и других программ-роботов, которые рассылают спам (некорректную навязчивую рекламу и другую не востребуемую информацию), взламывают пароли систем и пытаются выступать от имени их зарегистрированных пользователей и совершают другие неправомерные действия [1].

Таким образом, возникает задачи [1]:

- идентификации пола и других параметров собеседника (на эту возможность применения своего теста указывал и сам Тьюринг);
- выявления писем, написанных и посланных не людьми, а также такого автоматического написания писем, чтобы отличить их от написанных людьми было невозможно. Так что антиспамовый фильтр на электронной почте тоже представляет собой что-то вроде теста Тьюринга.

Не исключено, что скоро подобные проблемы (идентификации: человек или программа) могут возникнуть и в чатах. Что мешает сделать сетевых роботов типа программы «Элиза», но значительно более совершенных, которые будут сами регистрироваться в чатах и форумах участвовать в них с использованием слов и модифицированных предложений других участников? Простейший вариант – дублирование тем с других форумов и перенос их с форума на форум без изменений, что мы уже иногда наблюдаем в Internet (например: сквозная тема про «Чакра-муни»).

На практике чтобы на входе системы определить, кто в нее входит, человек или робот, достаточно при входе предъявить для решения простенькую для человека, но требующую огромных вычислительных ресурсов и системы типа неокогнитрона Фукушимы, задачку распознавания случайных наборов символов, представленных в нестандартных начертаниях,

масштабах и поворотах на фоне шума. Решил, – значит стучится человек-пользователь, не решил, – значит на входе робот, лазающий по мировой сети с неизвестными, чаще всего неблагоприятными целями.

Будем рассматривать следующие классы систем искусственного интеллекта [1].

- 1. Системы с интеллектуальной обратной связью и интеллектуальными интерфейсами.*
- 2. Автоматизированные системы распознавания образов.*
- 3. Автоматизированные системы поддержки принятия решений*
- 4. Экспертные системы (ЭС).*
- 5. Нейронные сети.*
- 6. Генетические алгоритмы и моделирование эволюции.*
- 7. Когнитивное моделирование.*
- 8. Выявление знаний из опыта (эмпирических фактов) и интеллектуальный анализ данных (data mining).*

Данная классификация не является исчерпывающей.

Система искусственного интеллекта в качестве существенной своей части включает: базу знаний, которая является результатом обобщения опыта эксплуатации данной системы в определенных конкретных условиях. Это значит, что программистом может быть разработана только «пустая оболочка» системы искусственного интеллекта, которая превращается в работоспособную систему в результате процесса обучения, который, таким образом, является необходимым технологическим этапом создания подобных систем. Можно провести аналогию между такой системой и ребенком: ребенок не может идти работать, т.к. ему для этого предварительно требуется длительное обучение в школе, а затем часто и в вузе, чтобы он смог выполнять определенные виды работ [1].

2.3 Информационная модель реакции систем искусственного интеллекта на воздействия окружающей среды

Модель реагирования системы на вызовы среды, предложенной в 1984 году В.Н. Лаптевым показана на рисунке 2.1. *На вход системы поступает задача или проблема.* Толкование различия между ними также дано В.Н. Лаптевым и состоит в следующем.

Ситуация, при которой фактическое состояние системы не совпадает с желаемым (целевым) называется **проблемной ситуацией** и представляет собой [1]:

- **задачу**, если способ перевода системы из фактического состояния в желаемое точно известен, и необходимо лишь применить его;

- *проблему*, если способ перевода системы из фактического состояния в желаемое не известен, и необходимо сначала его разработать и лишь затем применить его.

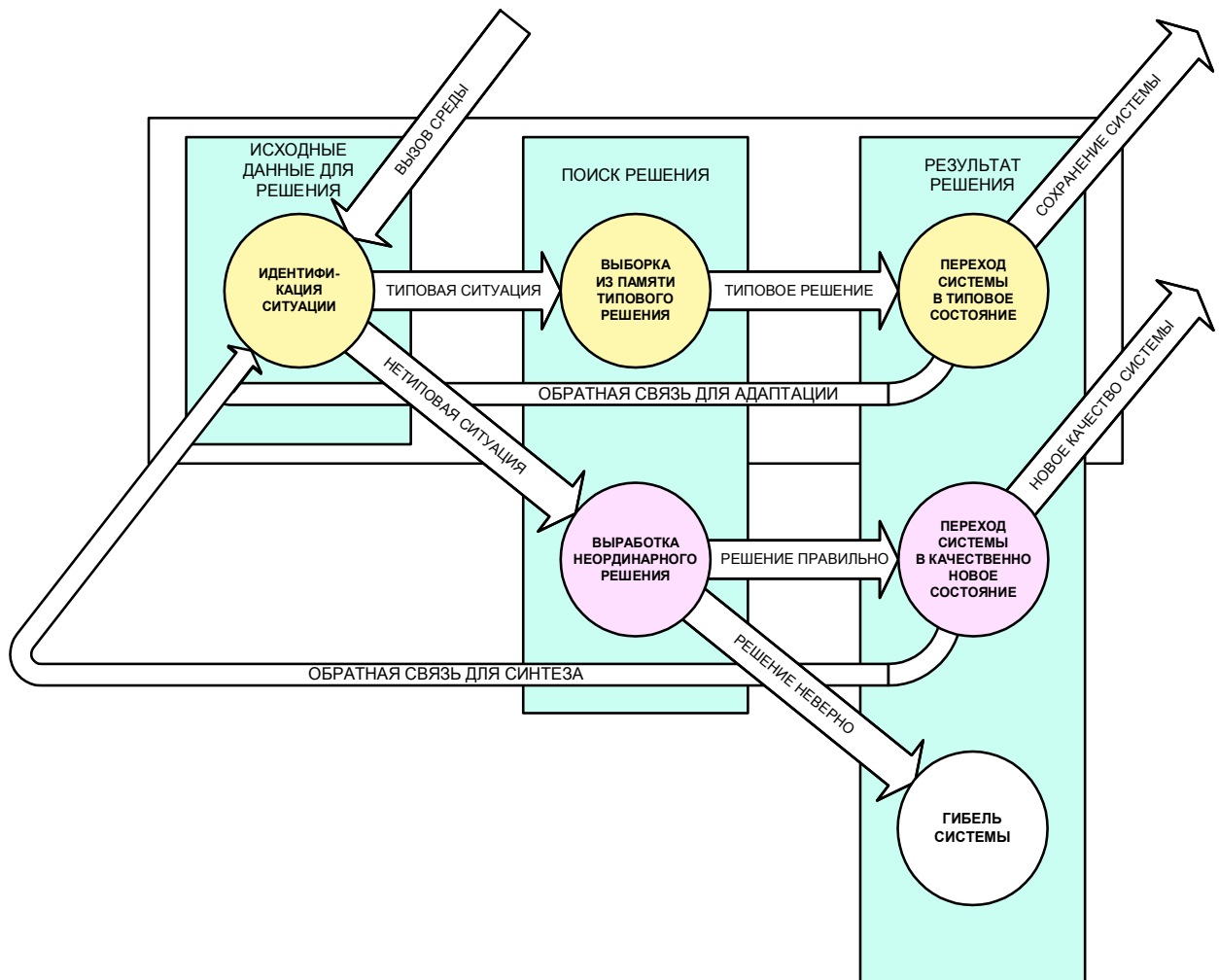


Рисунок 2.1 - Схема реагирования открытой системы на вызовы среды по В.Н. Лаптеву

Таким образом, можно считать, что *проблема* – это задача, способ решения которой неизвестен. Это означает, что если этот способ разработать, то этим самым проблема сводится к задаче, переводится в класс задач. Проще говоря, проблема – это сложная задача, а задача – это простая проблема.

Но и проблемы различаются по уровню сложности [1]:

- для решения одних достаточно автоматизированной системы поддержки принятия решений;
- для решения других – обязательным является творческое участие людей: специалистов, экспертов.

Информационная модель деятельности специалиста, представленная на рисунке 2.2 [1], разработана на основе модели, предложенной В.Н. Лаптевым.

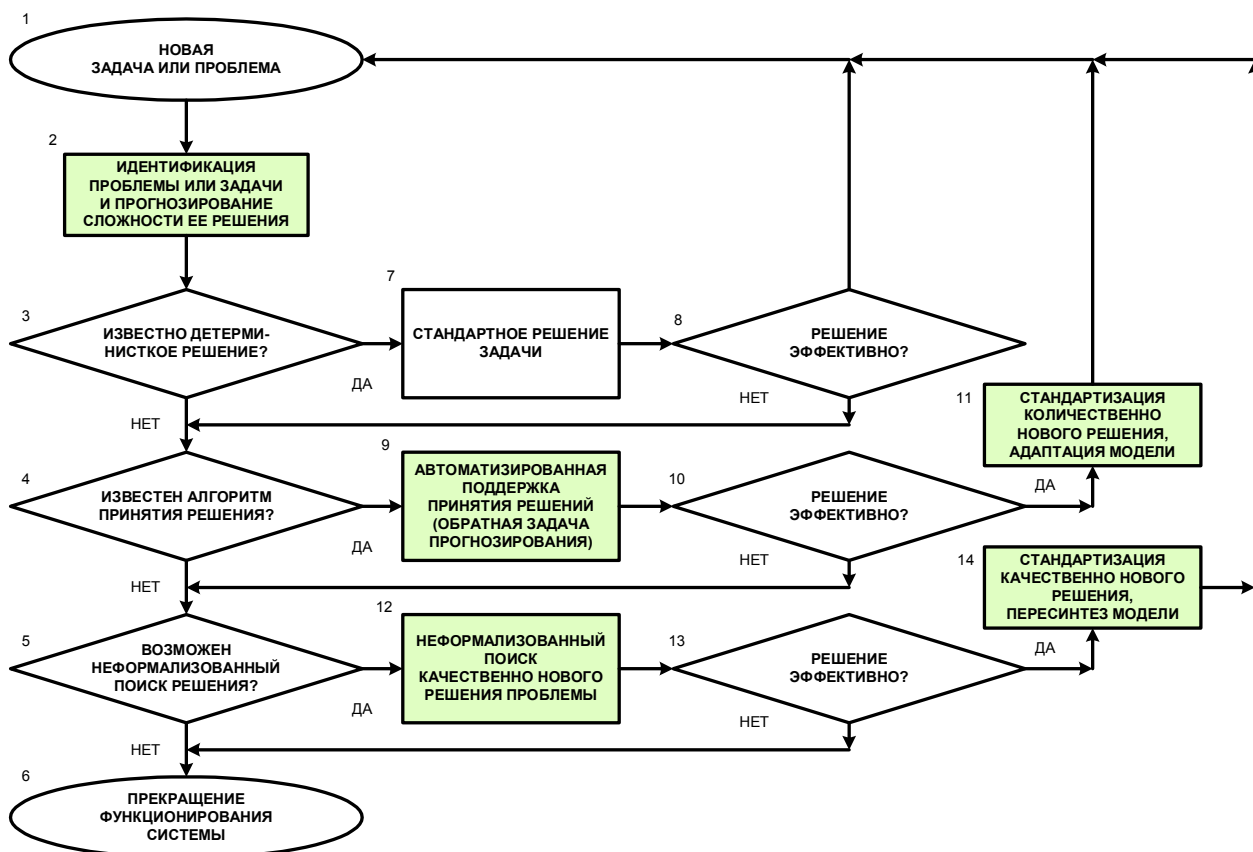


Рисунок 2.2 - Информационная модель деятельности специалиста и место систем искусственного интеллекта в этой деятельности

Рассмотрим информационную модель деятельности специалиста, представленную на рисунке 2.2 [1].

Блок 1. На вход системы поступает задача или проблема. Что именно неясно, т.к. чтобы это выяснить необходимо идентифицировать ситуацию и обратиться к базе данных стандартных решений с запросом, существует ли стандартное решение для данной ситуации.

Блок 2. Далее осуществляется идентификация проблемы или задачи и прогнозирование сложности ее решения. На этом этапе применяется интеллектуальная система, относящаяся к классу систем распознавания образов, идентификации и прогнозирования или эта функция реализуется специалистом самостоятельно "вручную".

Блок 3. Если в результате идентификации задачи или проблемы по ее признакам установлено, что точно имеется стандартное решение, то это означает, что на вход системы поступила точно такая же задача, как уже когда-то ранее встречалась. Для установления этого достаточно информационно-поисковой системы, осуществляющей поиск по точному совпадению параметров запроса и в применении интеллектуальных систем нет необходимости. Тогда происходит переход на блок 7, а иначе на блок 4.

Блок 4. Если установлено, что точно такой задачи не встречалось, но встречались сходные, аналогичные, которые могут быть найдены в результате обобщенного (нечеткого) поиска системой распознавания образов, то решение может быть найдено с помощью автоматизированной системы поддержки принятия решений путем решения обратной задачи прогнозирования. Это значит, что на вход системы поступила не задача, а проблема, имеющая количественную новизну по сравнению с решаемыми ранее (т.е. не очень сложная проблема). В этом случае осуществляется переход на блок 9, иначе – на блок 5.

Блок 5. Если установлено, что сходных проблем не встречалось, то необходимо качественно новое решение, поиск которого требует существенного творческого участия человека-эксперта. В этом случае происходит переход на блок 12, а иначе – на блок 6.

Блок 6. Переход на этот блок означает, что возможности поиска решения или выхода из проблемной ситуации системой исчерпаны и решения не найдено. В этом случае система обычно терпит ущерб целостности своей структуре и полноте функций, вплоть до разрушения и прекращения функционирования.

Блок 7. На этом этапе осуществляется реализация стандартного решения, соответствующего точно установленной задаче, а затем проверяется эффективность решения на блоке 8.

Блок 8. Если стандартное решение оказалось эффективным, это означает, что на этапах 2 и 3 идентификация задачи и способа решения осуществлены правильно и система может переходить к разрешению следующей проблемной ситуации (переход на блок 1). Если же стандартное решение оказалось неэффективным, то это означает, что проблемная ситуация идентифицирована как стандартная задача неверно и необходимо продолжить попытки ее разрешения с использованием более общих подходов, основанных на применении систем искусственного интеллекта (переход на блок 4), например, систем поддержки принятия решений.

Блок 9. Применяется автоматизированная система поддержки принятия решений, обеспечивающая решение обратной задачи прогнозирования. Отличие подобных систем от информационно-поисковых состоит в том, что они способны производить обобщение, выявлять силу и направление влияния различных факторов на поведение системы, и, на основе этого, по заданному целевому состоянию вырабатывать рекомендации по системе факторов, которые могли бы перевести систему в это состояние (обратная задача прогнозирования).

Блок 10. Если решение, полученное с помощью системы поддержки принятия решений, оказалось неэффективным, то это означает, что проблемная ситуация идентифицирована как аналогичная ранее встречавшимся неверно. Следовательно, что на вход системы поступила

качественно новая, по сравнению с решаемыми ранее, т.е. сложная проблема. В этом случае необходимо продолжить попытки разрешения проблемы с использованием творческих неформализованных подходов с участием человека-эксперта и перейти на блок 5, иначе – на блок 11.

Блок 11. Информация об условиях и результатах решения проблемы заносится в базу знаний, т.е. стандартизируется. После чего база знаний количественно (не принципиально) изменяется, т.е. осуществляется ее адаптация. В результате адаптации при встрече в будущем точно таких же проблемных ситуаций, как разрешенная, система уже будет разрешать ее не как проблему, а как стандартную задачу.

Блок 12. На этом этапе с использованием неформализованных творческих подходов осуществляется поиск качественно нового решения проблемы, не встречавшейся ранее, после чего управление передается блоку 13.

Блок 13. Если решение, полученное экспертами с помощью неформализованных подходов, оказалось неэффективным, то это означает, что система терпит крах (осуществляется переход на блок 6). Если же адекватное решение найдено, то происходит переход на блок 14.

Блок 14. Стандартизация качественно нового решения, проблемы и пересинтез модели. Информация об условиях и результатах творческого решения проблемы заносится в базу знаний, т.е. стандартизируется. После этого база знаний качественно, принципиально изменяется, т.е. фактически осуществляется ее пересоздание (пересинтез). В результате пересинтеза базы знаний при встрече в будущем проблемных ситуаций, аналогичных разрешенной, система уже будет реагировать на них как проблемы, решаемые автоматизированными системами поддержки принятия решений.

Блоки, в которых используются системы искусственного интеллекта, на рисунке 2.2 показаны затемненными:

- блоки 2 и 12: система распознавания образов, идентификации и прогнозирования;
- блоки 9, 11, 12 и 14: автоматизированная система поддержки принятия решений.

В заключение, приведем шуточный алгоритм решения проблем (рисунок 2.3) [1].

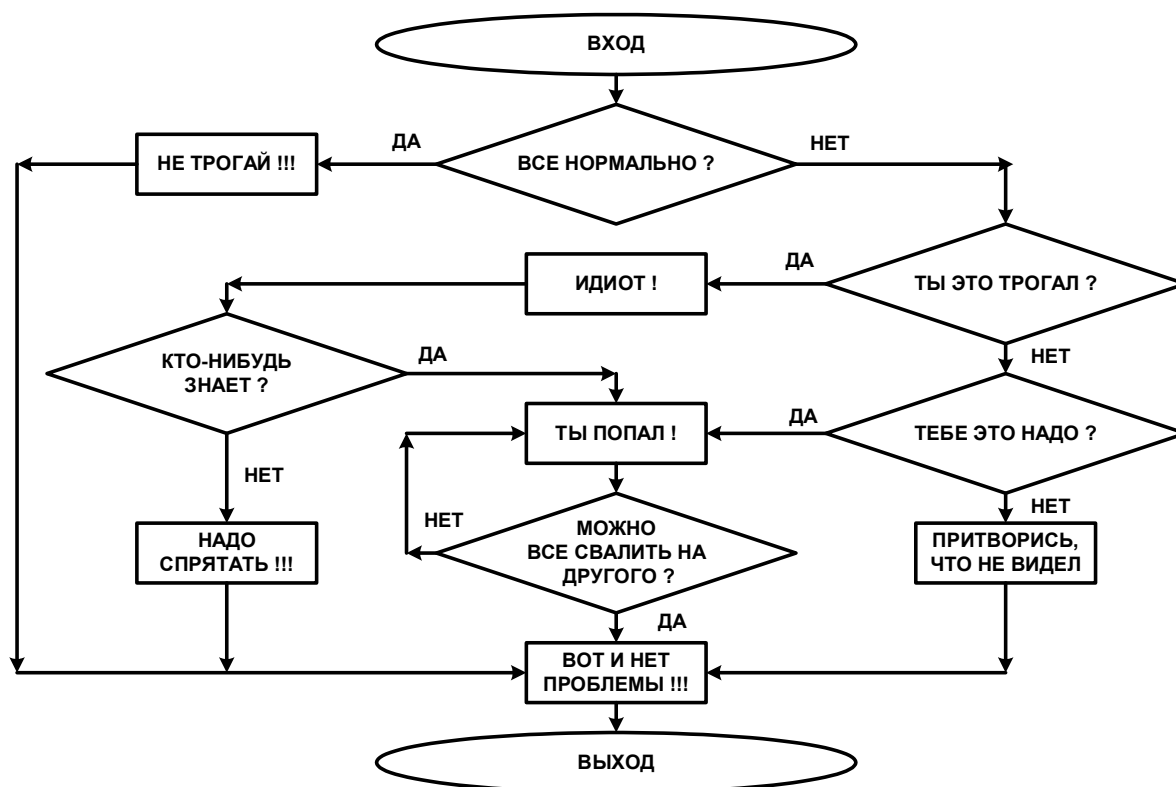


Рисунок 2.3 - Шуточный алгоритм решения проблем (Internet-фольклор)

2.4 Жизненный цикл системы искусственного интеллекта и критерии перехода между этапами этого цикла

Жизненный цикл систем искусственного интеллекта сходен с жизненным циклом другого программного обеспечения и включает этапы и критерии перехода между ними, представленные в таблице 2.1 [1].

Таблица 2.1 – Этапы жизненного цикла систем искусственного интеллекта и критерии перехода между ними, в соответствии с работой [1]

№	Наименование этапа	Критерии перехода к следующему этапу
1	Разработка идеи и концепции системы	Появление (в результате проведения маркетинговых и рекламных мероприятий) заказчика или спонсора, заинтересовавшегося системой
2	Разработка теоретических основ системы	Обоснование выбора математической модели по критериям или обоснование необходимости разработки новой модели
3	Разработка математической модели системы	Детальная разработка математической модели

№	Наименование этапа	Критерии перехода к следующему этапу
4	Разработка методики численных расчетов в системе:	
4.1	– разработка структур данных	детальная разработка структур входных, промежуточных и выходных данных
4.2	– разработка алгоритмов обработки данных	разработка обобщенных и детальных алгоритмов, реализующих на разработанных структурах данных математическую модель
5	Разработка структуры системы и экранных форм интерфейса	Разработка иерархической системы управления системой, структуры меню, экранных форм и средств управления на экранных формах
6	Разработка программной реализации системы	Разработка исходного текста программы системы, его компиляция и линковка. Исправление синтаксических ошибок в исходных текстах
7	Отладка системы	Поиск и исправление логических ошибок в исходных текстах на контрольных примерах. На контрольных примерах новые ошибки не обнаруживаются.
8	Экспериментальная эксплуатация	Поиск и исправление логических ошибок в исходных текстах на реальных данных без применения результатов работы системы на практике. На реальных данных новые ошибки практически не обнаруживаются, но считаются в принципе возможными.
9	Опытная эксплуатация	Поиск и исправление логических ошибок в исходных текстах на реальных данных с применением результатов работы системы на практике. На реальных данных новые ошибки не обнаруживаются и считаются недопустимыми.
10	Промышленная эксплуатация	Основной по длительности период, который продолжается до тех пор, пока система функционально устраивает Заказчика. У Заказчика появляется необходимость внесения количественных (косметических) изменений в систему на уровне п.5 (т.е. без изменения математической модели, структур данных и алгоритмов)
11	Заказные модификации системы	У Заказчика формируется потребность внесения качественных (принципиальных) изменений в систему на уровне п.3 и п.4, т.е. с изменениями в математической модели, структурах данных и алгоритмах
12	Разработка новых версий системы	Выясняется техническая невозможность или финансовая нецелесообразность разработки новых версий системы
13	Снятие системы с эксплуатации	

В основу главы 2 положен материал учебного пособия [1].

3. КЛАССИФИКАЦИЯ СИСТЕМ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

3.1 Понятие системы искусственного интеллекта и ее место в классификации информационных систем

Существует много различных подходов к классификации информационных систем. Вполне закономерно возникают вопросы о том [1]:

- Чем обусловлено различие этих классификаций?
- Какова классификация этих классификаций?
- Каким образом выбрать ту классификацию, которая больше всего подходит в конкретном случае?

Различия между классификациями определяются теми критериями, по которым производится классификация, например [1]:

- по степени структурированности решаемых задач;
- по автоматизируемым функциям;
- по степени автоматизации реализуемых функций;
- по сфере применения и характеру использования информации, в частности, по уровням управления.

Известно, что при обучении людей существуют различные уровни предметной обученности [1]:

- воспроизведение (память);
- решение стандартных задач (умения, навыки);
- решение нестандартных, творческих задач (знания, активное интеллектуальное понимание).

Интеллект может проявляться в различных областях, но мы рассмотрим его возможности в решении задач, т.к. эта область проявления является типичной для интеллекта.

*Задачи бывают стандартные и нестандартные. Для стандартных задач известны **алгоритмы** решения. Для нестандартных они неизвестны. Поэтому решение нестандартной задачи представляет собой **проблему** [1].*

Само понятие «стандартности» задачи **относительно**, относительна сама «неизвестность»: т.е. алгоритм может быть известен одним и неизвестен другим, или информация о нем может быть недоступной в определенный момент или период времени, и доступной – в другой. Поэтому для одних задача может быть стандартной, а для других нет.

Нахождение или разработка алгоритма решения переводит задачу из разряда нестандартных в стандартные [1].

В математике и кибернетике задача считается **решенной**, если известен алгоритм ее решения. Тогда процесс ее фактического решения превращается в рутинную работу, которую могут в точности выполнить человек, вычислительная машина или робот, под управлением программы реализующей данный алгоритм, не имеющие ни малейшего представления о смысле самой задачи [1].

Разработка алгоритма решения задачи связано с тонкими и сложными рассуждениями, требующими изобретательности, опыта, высокой квалификации. Считается, что эта работа является творческой, существенно неформализуемой и требует участия человека с его «естественным» опытом и интеллектом.

Необходимо отметить, что существует технология решения изобретательских задач (ТРИЗ), в которой сделана попытка, по мнению многих специалистов, довольно успешная, позволяющая в какой-то степени формализовать процедуру решения творческих задач.

Интеллектуальными считаются задачи, связанные с разработкой алгоритмов решения ранее нерешенных задач определенного типа [1].

Отличительной особенностью и одним из основных источников эффективности алгоритмов является то, что они сводят решение сложной задачи к определенной последовательности **достаточно простых** или даже элементарных для решения задач. В результате нерешаемая задача становится решаемой. Исходная информация поступает на вход алгоритма, на каждом шаге она преобразуется и в таком виде передается на следующий шаг, в результате чего на выходе алгоритма получается информация, представляющая собой решение задачи.

Алгоритм **может быть исполнен** такой системой, которая способна реализовать элементарные операции на различных шагах этого алгоритма.

Существует ряд задач, таких, как распознавание образов и идентификация, прогнозирование, принятие решений по управлению, для которых разбиение процесса поиска решения на отдельные элементарные шаги, а значит и разработка алгоритма, весьма затруднительны.

Интеллект - универсальный алгоритм, способный разрабатывать алгоритмы решения конкретных задач [1].

С этой точки зрения профессия программиста является одной из самых творческих и интеллектуальных, т. к. продуктом деятельности программиста являются алгоритмы реализованные на некотором языке программирования (программы).

Исходя из вышесказанного можно сделать вывод о том, что в нашем случае наиболее подходит классификацией ИС, основанная на критерии, позволяющем оценить «степень интеллектуальности ИС», т. е. на критерии «степени структурированности решаемых задач» (рисунок 3.1) [1].

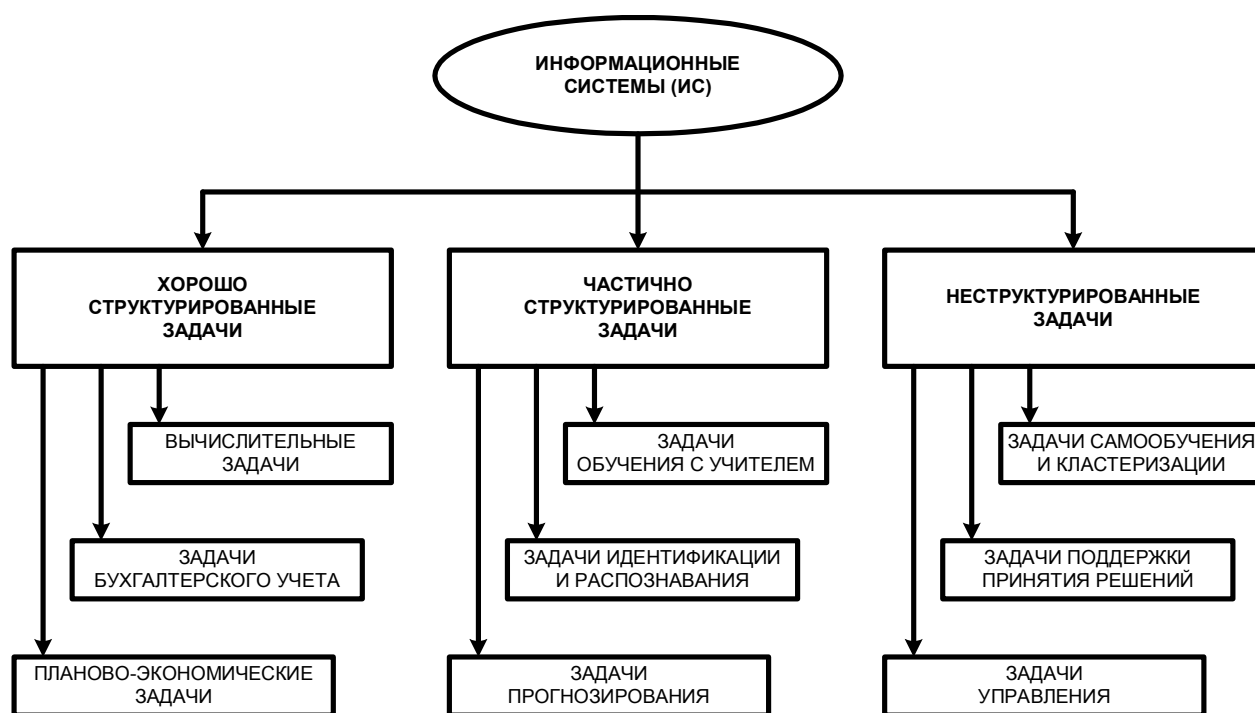


Рисунок 3.1 - Классификация информационных систем по степени структурированности решаемых задач

3.2 Классификация систем искусственного интеллекта

Классификация систем искусственного интеллекта приведена ниже.

Системы с интеллектуальной обратной связью и интеллектуальными интерфейсами. Интеллектуальный интерфейс (Intelligent interface) - интерфейс непосредственного взаимодействия ресурсов информационного комплекса и пользователя посредством программ обработки текстовых запросов пользователя. Примером может служить программа идентификация и аутентификация личности по почерку [1].

Автоматизированные системы распознавания образов [1]:

- формирование конкретных образов объектов и обобщенных образов классов;
- обучение, т.е. формирование обобщенных образов классов на основе ряда примеров объектов, классифицированных (т.е. отнесенных к тем или иным категориям – классам) учителем и составляющих обучающую выборку;
- самообучение, т.е. формирование кластеров объектов на основе анализа неклассифицированной обучающей выборки;

- распознавание, т.е. идентификацию (и прогнозирование) состояний объектов, описанных признаками, друг с другом и с обобщенными образами классов;
- измерение степени адекватности модели;
- решение обратной задачи идентификации и прогнозирования

Автоматизированные системы поддержки принятия решений.

Системы поддержки принятия решений (СППР) – это компьютерные системы, почти всегда интерактивные, разработанные, чтобы помочь менеджеру (или руководителю) в принятии решений управления, объединяя данные, сложные аналитические модели и удобное для пользователя программное обеспечение в единую мощную систему, которая может поддерживать слабоструктурированное и неструктурированное принятие решения. СППР находится под управлением пользователя от начала до реализации и используется ежедневно. Предназначена для автоматизации выбора рационального варианта из исходного множества альтернативных в условиях многокритериальности и неопределенности исходной информации [1].

Экспертные системы (ЭС). Это программа, которая в определенных отношениях заменяет эксперта или группу экспертов в той или иной предметной области. ЭС предназначены для решения практических задач, возникающих в слабо структурированных и трудно формализуемых предметных областях [1].

Исторически, ЭС были первыми системами искусственного интеллекта, которые привлекли внимание потребителей. Экспертные системы используются в маркетинге для сегментации рынка и выработке маркетинговых программ, а также в банковском деле для определения тенденции рынка, трейдинг для программирования котировок акций и валют, в аудите для подготовки заключений о финансовом состоянии предприятий [1].

Генетические алгоритмы и моделирование эволюции. Генетические Алгоритмы (ГА) – это адаптивные методы функциональной оптимизации, основанные на компьютерном имитационном моделировании биологической эволюции [1].

Когнитивное моделирование. Это способ анализа, обеспечивающий определение силы и направления влияния факторов на перевод объекта управления в целевое состояние с учетом сходства и различия в влиянии различных факторов на объект управления [1].

Основана на когнитивной структуризации предметной области, т.е. на выявление будущих целевых и нежелательных состояний объекта управления и наиболее существенных (базисных) факторов управления и внешней среды, влияющих на переход объекта в эти состояния, а также

установление на качественном уровне причинно-следственных связей между ними, с учетом взаимовлияния факторов друг на друга.

Выявление знаний из опыта и интеллектуальный анализ данных. Интеллектуальный анализ данных (ИАД или data mining) – это процесс обнаружения в «сырых» данных ранее неизвестных, нетривиальных, практически полезных и доступных интерпретации знаний, необходимых для принятия решений в различных сферах человеческой деятельности. Достижения технологии data mining активно используются в банковском деле для решения проблем Телекоммуникации, анализа биржевого рынка и др. [1].

Нейронные сети. Искусственная нейронная сеть (ИНС, нейросеть) - это набор нейронов, соединенных между собой. Как правило, передаточные функции всех нейронов в сети фиксированы, а веса являются параметрами сети и могут изменяться. Некоторые входы нейронов помечены как внешние входы сети, а некоторые выходы - как внешние выходы сети. Подавая любые числа на входы сети, мы получаем какой-то набор чисел на выходах сети. Практически любую задачу можно свести к задаче, решаемой нейросетью [1].

В основу главы 3 положен материал учебного пособия [1].

ЧАСТЬ 2. ЭЛЕМЕНТЫ СИСТЕМНО-КОГНИТИВНОГО АНАЛИЗА

4. СИСТЕМНО-КОГНИТИВНЫЙ АНАЛИЗ

4.1 Основные понятия когнитивной теории

Термин *«когнитивный»* происходит от *«cognition»* – «познание» (англ.) и используется для обозначения перспективного направления психологии (когнитивная психология), а также направления развития систем искусственного интеллекта (когнитивное моделирование и системно-когнитивный анализ), в которых ставится и решается задача автоматизации некоторых функций, реализуемых человеком, в процессе познания [1].

Исторически процессы познания первоначально изучались философами. В философии теория познания (сознания) называется *гносеологией*, от греч. *gnosis*, – знание, учение, познание, в отличие от *онтологии* – учения о бытие.

Однако, философский анализ процессов познания не касается исследования естественно-научными методами **конкретных форм сознания** и характерных для них **методов познания**, а также конкретных **способов их достижения** и реализации.

Когнитивная психология – это область психологии, непосредственно теоретически и экспериментально изучающая процессы познания у конкретных людей, различного пола, возраста, социального статуса и т.д.

Когнитивное моделирование – это способ анализа, обеспечивающий определение силы и направления влияния факторов на перевод объекта управления в целевое состояние с учетом сходства и различия в влиянии различных факторов на объект управления [1].

Как правило, когнитивное моделирование ведется путем построения и анализа когнитивных карт.

Классическая когнитивная карта – это ориентированный граф, в котором привилегированной вершиной является некоторое будущее (как правило, целевое) состояние объекта управления, остальные вершины соответствуют факторам, дуги, соединяющие факторы с вершиной состояния имеют толщину и знак, соответствующий силе и направлению влияния данного фактора на переход объекта управления в данное состояние, а дуги, соединяющие факторы показывают сходство и различие в влиянии этих факторов на объект управления [1].

4.2 Концепция системно-когнитивного анализа

4.2.1 Базовая когнитивная концепция

Автоматизировать процесс познания в целом безусловно значительно сложнее, чем отдельные операции процесса познания. Для этого необходимо [1]:

- выявить эти операции;
- найти место каждой из них в системе или последовательности процесса познания.

Когнитивной концепции, которая позволит автоматизировать процесс познания должна удовлетворять следующим требованиям [1]:

- адекватность, т.е. точное отражение сущности процессов познания, характерных для человека, в частности описание процессов вербализации, семантической адаптации и семантического синтеза (уточнения смысла слов и понятий и включения в словарь новых слов и понятий);
- высокая степень детализации и структурированности до уровня достаточно простых базовых когнитивных операций;
- возможность математического описания, формализации и автоматизации.

Положения когнитивной концепции приведены в определенном порядке, соответствующем реальному ходу процесса познания "от конкретных эмпирических исходных данных к содержательным информационным моделям, а затем к их верификации, адаптации и, в случае необходимости, к пересинтезу".

На базе выше сформулированных положений предложена целостная система взглядов на процесс познания, т.е. **формализуемая когнитивная концепция**, предназначенная для построения систем искусственного интеллекта.

Процесс познания рассматривается как многоуровневая иерархическая система обработки информации, в которой каждый последующий уровень является результатом интеграции элементов предыдущего уровня (рисунок 4.1) [1].

На 1-м уровне этой системы находятся дискретные элементы потока чувственного восприятия, которые на 2-м уровне интегрируются в чувственный образ конкретного объекта. Те, в свою очередь, на 3-м уровне интегрируются в обобщенные образы классов и факторов, образующие на 4-м уровне кластеры, а на 5-м конструкторы. Система конструкторов на 6-м уровне образуют текущую парадигму реальности (т.е. человек познает мир путем синтеза и применения конструкторов). На 7-м же уровне

обнаруживается, что текущая парадигма не является единственно-возможной [1].

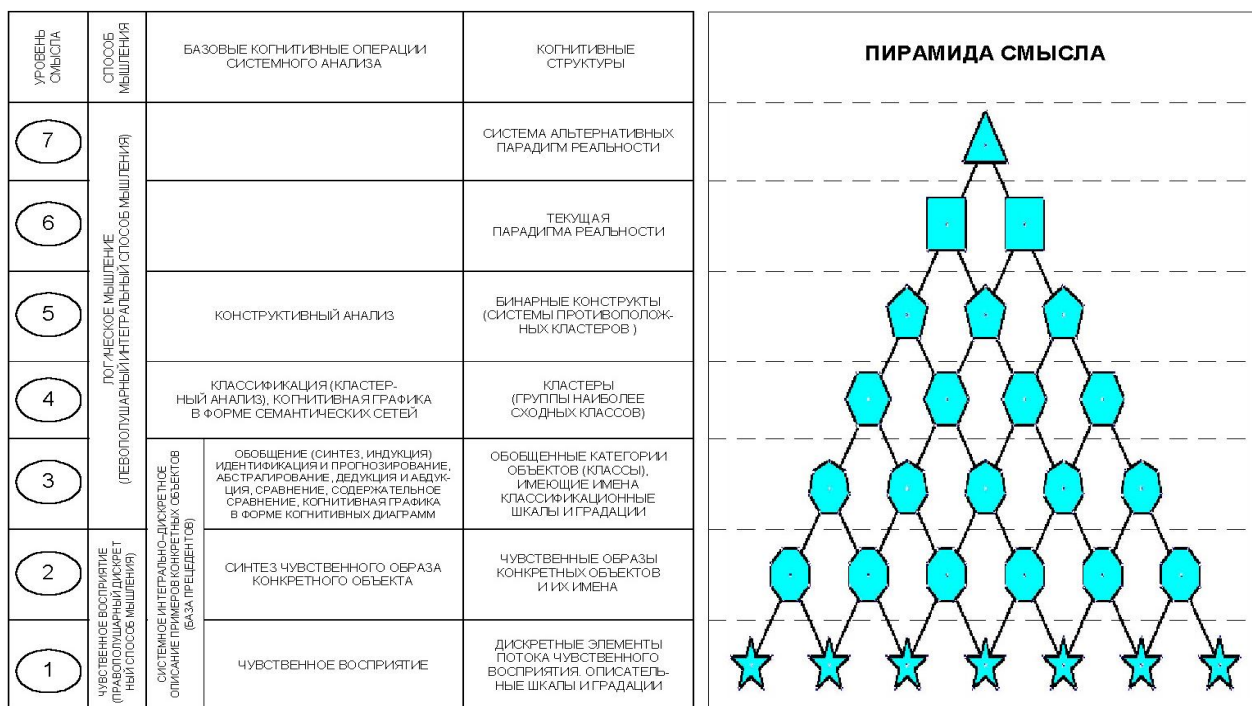


Рисунок 4.1 - Обобщенная схема когнитивной концепции

4.2.2 Когнитивная концепция в свободном изложении

Исходные данные для процесса познания поставляются из нескольких *независимых* информационных источников, имеющих *качественно* различную природу, которые мы будем условно называть «*органы чувств*». Данные от органов чувств также имеют *качественно* различную природу, обусловленную конкретным видом информационного источника. Для обозначения этих исходных данных будем использовать термин «*атрибут*». В результате выполнения когнитивной операции «присвоение имен» атрибутам могут быть присвоены уникальные имена, т.е. они могут быть отнесены к некоторым градациям номинальных шкал. Получение информации о предметной области в атрибутивной форме осуществляется когнитивной операцией «восприятие».

Исходные данные содержат внутренние закономерности, объединяющие *качественно* разнородные исходные данные от различных информационных источников.

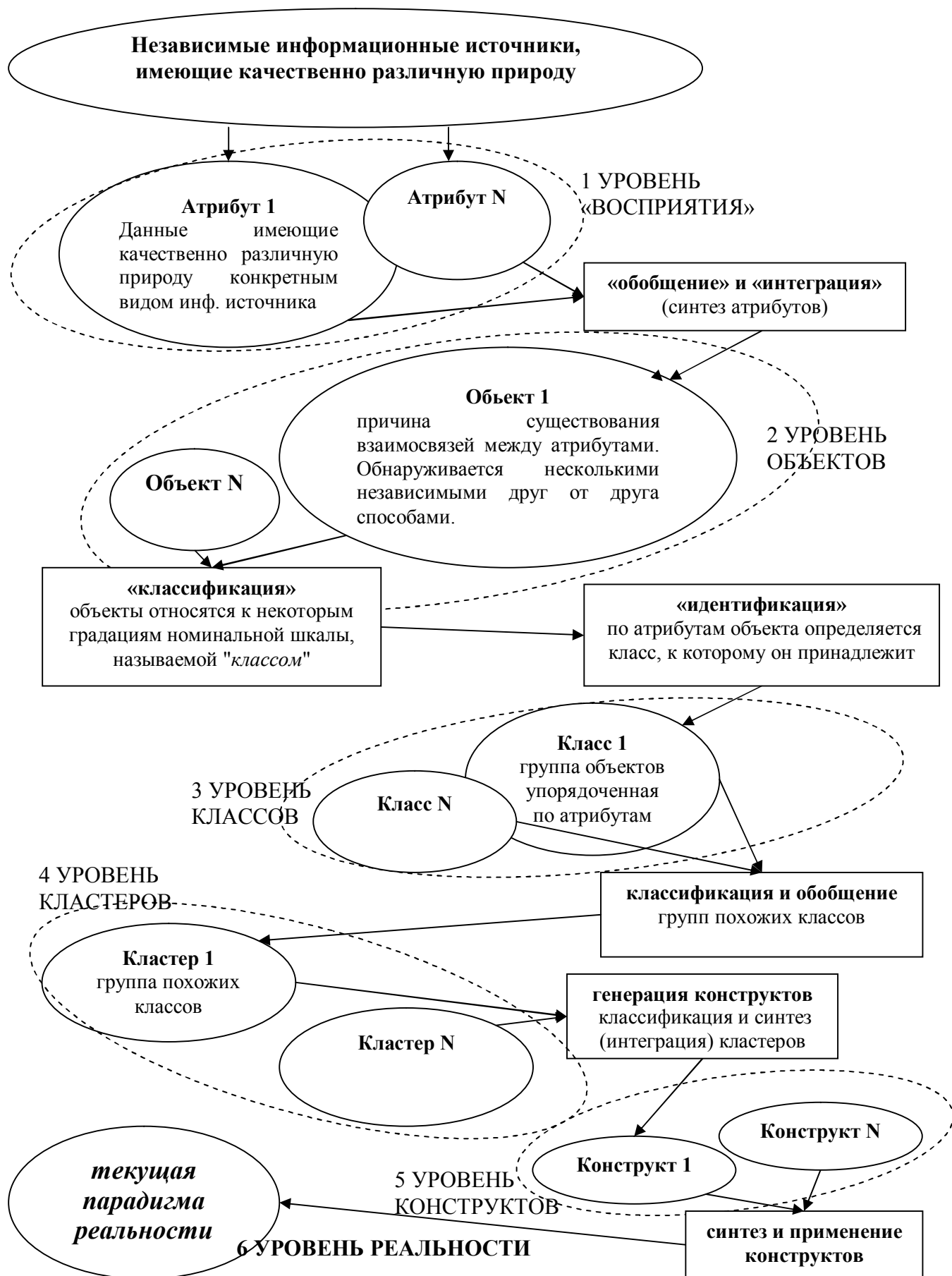


Рисунок 4.2 - Процесс познания как многоуровневая иерархическая система обработки информации

После восприятия предметной области может быть проведен ее первичный анализ путем выполнения когнитивной операции: «сопоставление опыта, воплощенного в модели, с общественным», т.е. с результатами восприятия той же предметной области другими. Это делается с целью исключения из дальнейшего анализа всех наиболее явных расхождений, как сомнительных.

Однако, закономерности в предметной области могут быть выявлены путем выполнения когнитивной операции *«обобщение»* только после накопления в результате мониторинга достаточно большого объема исходных данных в *памяти*.

Наличие этих закономерностей позволяют предположить, что:

- существуют некие интегративные структуры, не сводящиеся ни к одному из качественно-различных аспектов исходных данных и обладающие по отношению к ним системными, т.е. эмерджентными свойствами, которые не могут быть предметом прямого восприятия с помощью органов чувств, но могут являться предметом для других форм познания, например логической формы. Для обозначения этих структур будем использовать термин *«объект»*;
- «объекты» считаются причинами существования взаимосвязей между атрибутами.

Объектам приписывается объективное существование, в том смысле, что любой объект обнаруживается несколькими независимыми друг от друга способами с помощью различных органов чувств (этот критерий объективного существования в физике называется «принцип наблюдаемости»).

После обобщения возможны когнитивные операции: «определение значимости шкал и градаций атрибутов» и «определение степени сформированности шкал и градаций классов».

Путем выполнения когнитивной операции "присвоение имен" конкретным объектам могут быть присвоены уникальные имена, т.е. они могут быть отнесены к некоторым градациям номинальной шкалы, которые мы будем называть *«классами»*. После этого возможно выполнение когнитивной операции *«идентификация объектов»*, т.е. их «узнавание»: при этом по атрибутам объекта определяется класс, к которому принадлежит объект. *При этом все атрибуты, независимо от их качественно различной природы, рассматриваются с одной-единственной точки зрения: «Какое количество информации они несут о принадлежности данного объекта к каждому из классов»* [1].

Кроме того, возможно выполнение когнитивной операции: «дедукция и абдукция, обратная задача идентификации и прогнозирования», имеющей

очень важное значение для управления, т.е. вывод всех атрибутов в порядке убывания содержащегося в них количества информации о принадлежности к данному классу.

Аналогично, может быть выполнена когнитивная операция: «семантический анализ атрибута», представляющий собой список классов, в порядке убывания количества информации о принадлежности к ним, содержащейся в данном атрибуте.

Таким образом, возможно два взаимно-дополнительных способа отображения объекта [1]:

- в форме принадлежности к некоторому классу (целостное, интегральное, экстенциональное);
- в форме системы атрибутов (дискретное, интенциональное).

Дальнейшее изучение атрибутов позволяет ввести понятия «*порядковая шкала*» и «*градация*». Порядковая шкала представляет собой способ классификации атрибутов одного качества, обычно по степени выраженности (интенсивности).

Градация — это конкретное положение или диапазон на шкале, которому ставится в соответствие конкретный атрибут, соответствующее определенной степени интенсивности. Каждому виду атрибутов, информация о которых получается с помощью определенного «органа чувств», ставится в соответствие одна шкала. Таким образом, если при анализе в номинальных шкалах, можно было в принципе ввести одну шкалу для всех атрибутов, то в порядковых шкалах каждому атрибуту будет соответствовать своя шкала [1].

После идентификации уникальных объектов с классами возможна их классификация и присвоение *обобщающих имен* группам похожих классов. Для обозначения группы похожих классов используем понятие «**кластер**». Формирование кластеров осуществляется с помощью когнитивной операции «**классификация**». Кластер представляет собой своего рода «**объект, состоящий из объектов**», т.е. объект 2-го порядка. Если объект выполняет интегративную функцию по отношению к атрибутам, то кластер — по отношению к объектам.

Необходимо подчеркнуть, что термин «**класс**» используется не только для обозначения образов уникальных объектов, но и для обозначения их кластеров, т.е. *классу может соответствовать не уникальное, а обобщающее имя, в этом случае мы имеем дело с обобщенным классом*. Да и кластеры могут быть не только кластерами уникальных объектов, но и обобщенных классов.

Если объективное существование уникальных объектов мало у кого вызывает сомнение, то вопрос об объективном существовании интегративных структур 2-го и более высоких порядков остается открытым.

В некоторых философских системах подобным объектам приписывался даже более высокий статус существования, чем самим объектам, например обычные объекты рассматриваются лишь как «тени» «Эйдосов» (Платон). Известны и другие понятия для обозначения объектов высоких порядков, например «архетип» (Юм), «эгрегор» (Андреев) и др. Нельзя не отметить, что в современной физике (специальной и общей теории относительности) есть подобное понятие пространственно-временного интервала, который проявляется как *движение* объекта. По-видимому, статус существования структур реальности, отражаемых когнитивными структурами тем выше, чем выше интегративный уровень этих структур.

Являясь объектами 2-го порядка сами кластеры в результате выполнения когнитивной операции **«генерация конструктов»** могут быть классифицированы по степени сходства друг с другом. Для обозначения системы двух противоположных кластеров, с «спектром» промежуточных кластеров между ними, будем использовать термин **«бинарный конструкт»**, при этом сами противоположные кластеры будем называть **«полюса бинарного конструкта»**. Таким образом, конструкт представляет собой объект 3-го порядка.

Словом «бинарный» определяется, что в данном случае полюсов у конструкта всего два, но этим самым подчеркивается, что в принципе их может быть 3, 4 и больше. Бинарный конструкт можно формально представить в виде порядковой шкалы или даже шкалы отношений, на которой градациям соответствуют кластеры, а значит и сами классы и соответствующие объекты. Конструкты с количеством полюсов больше 2 могут быть представлены графически в форме семантических сетей в которых полюса являются вершинами, а дуги имеют цвет и толщину, соответствующие степени сходства-различия этих вершин. Семантические сети можно считать также просто графической формой представления результатов кластерного анализа.

Аналогично кластерам и конструктам классов формируются кластеры и конструкты атрибутов. В кластеры объединяются атрибуты, имеющие наиболее сходный смысл. В качестве полюсов конструктов выступают кластеры атрибутов, противоположных по смыслу.

В кластерном анализе определялась степень сходства или различия классов, а не то, чем конкретно сходны или отличаются. При выполнении когнитивной операции **«содержательное сравнение»** двух классов определяется вклад каждого атрибута в их сходство или различие. Результаты содержательного сравнения выводятся в наглядной графической форме когнитивных диаграмм, в которых изображаются информационные портреты классов с наиболее характерными и нехарактерными для них

атрибутами и атрибуты разных классов соединяются линиями, цвет и толщина которых соответствуют величине и знаку вклада этих атрибутов в сходство или различие данных классов.

Результаты идентификации и прогнозирования, осуществленные с помощью модели, путем выполнения когнитивной операции **«верификация»** сопоставляются с опытом, после чего определяется выполнять ли когнитивную операцию **«обучение»**, состоящую в том, что параметры модели могут изменяться количественно, и тогда мы имеем дело с адаптацией, или качественно, и тогда идет речь о переформировании модели.

4.2.3 Когнитивная концепция в формальном изложении

1. Процесс познания начинается с чувственного восприятия. Различные органы восприятия дают качественно-различную чувственную информацию в форме дискретного потока *элементов восприятия*. Эти элементы формализуются с помощью описательных шкал и градаций.

2. В процессе накопления опыта выявляются взаимосвязи между элементами чувственного восприятия: одни элементы часто наблюдаются с другими (имеет место их пространственно-временная корреляция), другие же вместе встречаются достаточно редко. Существование устойчивых связей между элементами восприятия говорит о том, что они отражают некую реальность, *интегральную* по отношению к этим элементам. Эту реальность будем называть объектами восприятия. Рассматриваемые в *единстве* с объектами элементы восприятия будем называть *признаками* объектов. Таким образом, органы восприятия дают чувственную информацию о признаках наблюдаемых объектов, процессов и явлений окружающего мира (**объектов**). Чувственный образ конкретного объекта представляет собой систему, возникающую как результат *процесса* синтеза признаков этого объекта. В условиях усложненного восприятия синтез чувственного образа объекта может быть существенно замедленным и даже не завершаться в реальном времени.

3. Человек присваивает конкретным объектам названия (имена) и сравнивает объекты друг с другом. При сравнении выясняется, что одни объекты в различных степенях сходны по их признакам, а другие отличаются. Сходные объекты объединяются в обобщенные категории (**классы**), которым присваиваются имена, производные от имен входящих в категорию конкретных объектов. Классы формализуются с помощью классификационных шкал и градаций и обеспечивают интегральный способ описания действительности. Путем обобщения (синтеза, индукции)

информации о признаках конкретных объектов, входящих в те или иные классы, формируются обобщенные образы классов. Накопление опыта и сравнение обобщенных образов классов друг с другом позволяет определить степень характерности признаков для классов, смысл признаков и ценность каждого признака для идентификации конкретных объектов с классами и сравнения классов, а также исключить наименее ценные признаки из дальнейшего анализа без существенного сокращения количества полезной информации о предметной области (**абстрагирование**). Абстрагирование позволяет существенно сократить затраты внутренних ресурсов системы на анализ информации. Идентификация представляет собой процесс узнавания, т.е. установление соответствия между чувственным описанием объекта, как совокупности дискретных признаков, и неделимым (целостным) именем класса, которое ассоциируется с местом и ролью воспринимаемого объекта в природе и обществе. Дискретное и целостное восприятие действительности поддерживаются, как правило, различными полушариями мозга: соответственно, правым и левым (доминантность полушарий). Таким образом, именно *системное* взаимодействие интегрального (целостного) и дискретного способов восприятия обеспечивает возможность установление *содержательного смысла* событий. При выполнении когнитивной операции «содержательное сравнение» двух классов определяется вклад каждого признака в их сходство или различие.

4. После идентификации уникальных объектов с классами возможна их классификация и присвоение *обобщающих имен* группам похожих классов. Для обозначения группы похожих классов используем понятие «*кластер*». Но и сами кластеры в результате выполнения когнитивной операции «генерация конструкторов» могут быть классифицированы по степени сходства друг с другом. Для обозначения системы двух противоположных кластеров, со «спектром» промежуточных кластеров между ними, будем использовать термин «*бинарный конструктор*», при этом сами противоположные кластеры будем называть «*полюса бинарного конструктора*». Бинарные конструкторы классов и атрибутов, т.е. конструкторы с двумя полюсами, наиболее типичны для человека и представляет собой когнитивные структуры, играющие огромную роль в процессах познания. Достаточно сказать, что *познание можно рассматривать как процесс генерации, совершенствования и применения конструкторов*. Качество конструктора тем выше, чем сильнее отличаются его полюса, т.е. чем больше диапазон его смысла.

Результаты идентификации и прогнозирования, осуществленные с помощью модели, путем выполнения когнитивной операции «верификация» сопоставляются с опытом, после чего определяется целесообразность

выполнения когнитивной операции «обучение». При этом может возникнуть три основных варианта:

1. Объект, входит в обучающую выборку и достоверно идентифицируется (внутренняя валидность, в адаптации нет необходимости).
2. Объект, не входит в обучающую выборку, но входит в исходную генеральную совокупность, по отношению к которой эта выборка репрезентативна, и достоверно идентифицируется (внешняя валидность, добавление объекта к обучающей выборке и адаптация модели приводит к количественному уточнению смысла признаков и образов классов).
3. Объект не входит в исходную генеральную совокупность и идентифицируется недостоверно (внешняя валидность, добавление объекта к обучающей выборке и синтез модели приводит к качественному уточнению смысла признаков и образов классов, исходная генеральная совокупность расширяется).

4.3 Когнитивное моделирование

В основе технологии когнитивного анализа и моделирования (рисунок 4.3) лежит когнитивная (познавательно-целевая) структуризация знаний об объекте и внешней для него среды.

Когнитивная структуризация предметной области – это выявление будущих целевых и нежелательных состояний объекта управления и наиболее существенных (базисных) факторов управления и внешней среды, влияющих на переход объекта в эти состояния, а также установление на качественном уровне причинно-следственных связей между ними, с учетом взаимовлияния факторов друг на друга [1].

Результаты когнитивной структуризации отображаются с помощью когнитивной карты (модели).

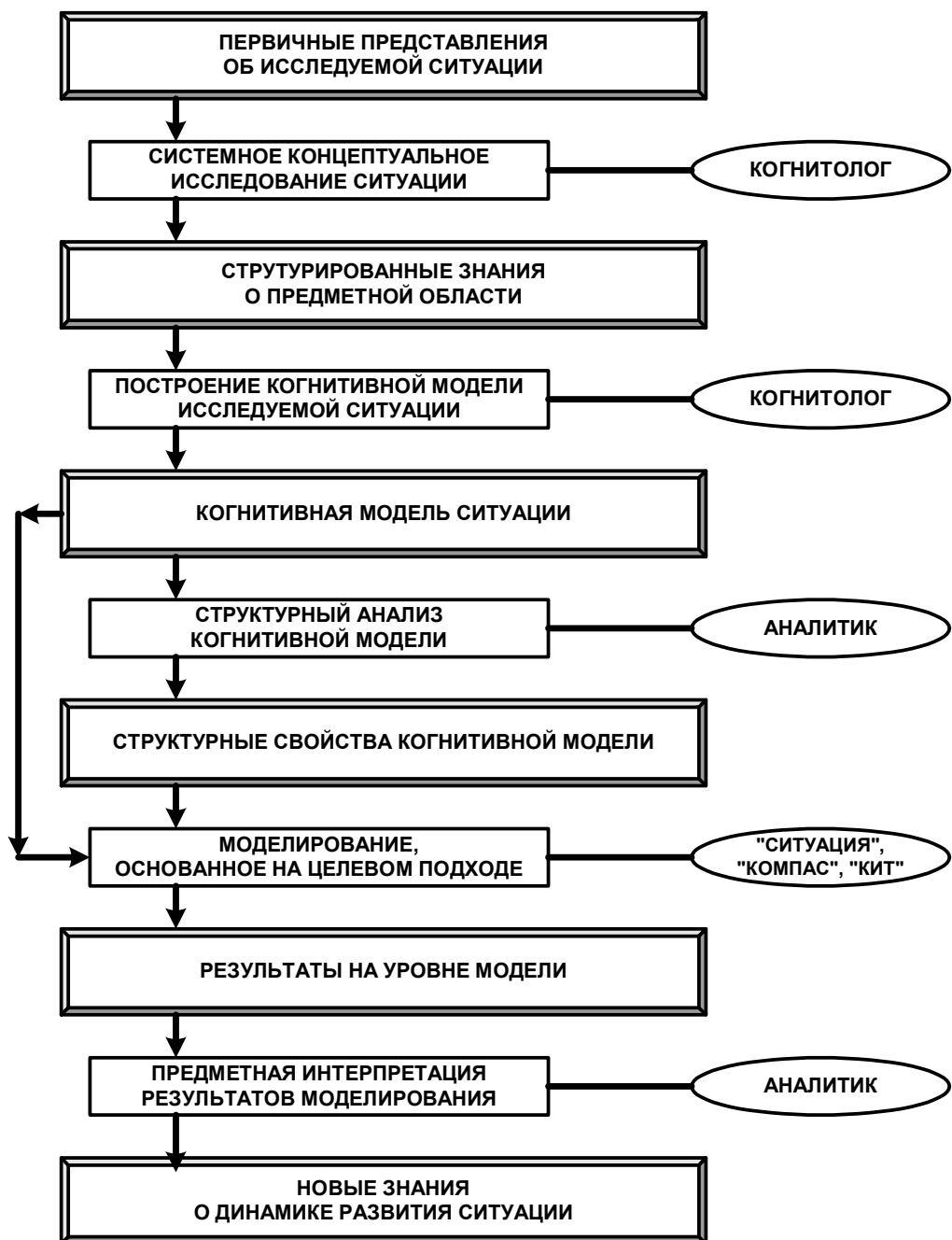


Рисунок 4.3 - Технология когнитивного анализа и моделирования

В основу главы 4 положен материал учебного пособия [1].

5. ПРЕДСТАВЛЕНИЕ И ОБРАБОТКА ДАННЫХ В РАМКАХ ТЕОРИИ СИСТЕМНО-КОГНИТИВНОГО АНАЛИЗА

5.1 Понятия «данные», «информация», «знания»

Существует неопределенность смыслового содержания «разночтения» терминов: «данные», «информация», «знания». Мы считаем целесообразным определить их следующим образом.

Данные представляют собой информацию, рассматриваемую в чисто синтаксическом аспекте, т.е. безотносительно к ее содержанию и использованию, т.е. семантике и телеологии (обычно на каком-либо носителе или в канале передачи) [1].

Информация – это данные, проинтерпретированные с использованием тезауруса, т.е. осмысленные данные, рассматриваемые в единстве синтаксического и семантического аспектов [1].

Знания, есть система информации, обеспечивающая увеличение вероятности достижения какой-либо **цели**, т.е. по сути знания – это «Ноу-хау» или технологии [1].

Вышесказанное резюмируем в следующей форме [1]:

знание = информация + цель

информация = данные + смысл;

знания = данные + смысл + цель.

От того, какое конкретное содержание вкладывается разработчиками в данные понятия, самым существенным образом зависят и подходы к созданию математических моделей, структур данных и алгоритмов функционирования СИИ.

Проблема состоит в том, что смысловое содержание этих понятий чаще всего не конкретизируется.

И это не случайно. Одной из основных причин этого положения дел, на наш взгляд, является то, что *конкретизировать смысловое содержание данных понятий представляется возможным лишь на основе интуитивно-ясной и хорошо обоснованной концепции смысла.*

Конечно, возникает вопрос о том, насколько вообще возможны, т.е. имеют смысл концепции смысла, не бессмысленны ли они? Может быть вопрос: «Какой смысл имеют концепции смысла?» – является одним из вариантов логического парадокса Рассела? Хотя эти вопросы имеют «несерьезный» оттенок, по сути, они сводятся к очень серьезному вопросу о том, насколько или в какой степени интеллект может познать сам себя, т.е. о

том, является интеллектуальная форма познания адекватным инструментом для познания интеллекта?

5.2 Концепция смысла Шенка-Абельсона

Анализ различных подходов к автоматизации процессов понимания смысла показал, что *все теории понимания смысла классифицируются на три группы* [1]:

- **Объектные теории** основаны на структурно-семантическом анализе, т.е. понимание приравнивается к самому объекту. Понять означает - установить значение языкового знака.
- **Субъект-объектные теории** синтактико-семантический анализа дополняют прагма-лингвистическим описанием или описанием реальных психологических процессов при создании и/или анализе языкового знака.
- **Субъектные теории** помимо собственно процедур понимания акцентируют внимание на оценке реципиентом **результатов** понимания (при помощи обратной связи в диалоге, рефлексии или монологе).

*Суть концепции смысла Шенка-Абельсона состоит в том, что факты рассматриваются как **причины** и их смысл считается известным, если известны **последствия** данного факта. Таким образом, понимание смысла определенных конкретных событий заключается в выявлении причинно-следственных взаимосвязей между этими событиями и другими* [1].

Данная концепция смысла является одной из наиболее интуитивно убедительных и хорошо обоснованных.

Ключевым моментом в концепции смысла Шенка-Абельсона является определение способа выявления силы и направленности влияния причинно-следственных взаимосвязей и их количественной оценки (меры) [1].

Слабым местом концепции смысла Шенка-Абельсона является сложность корректного и обоснованного выбора количественной меры силы и направленности причинно-следственных связей, а также конкретного способа определения численной величины этой меры в каждом конкретном случае (т.е. для каждого факта), причем непосредственно на основе эмпирических данных [1].

Проблема в том, что в общественном сознании продолжает господствовать упрощенческая точка зрения, состоящая в том, что корреляция является мерой причинно-следственных связей. И это имеет

место не смотря на многочисленные разъяснения в специальной литературе о том, что это не так, точнее не совсем так.

5.3 Диалектика «Структура – свойство – отношение» в рамках когнитивной теории

Рассмотрим простой пример. Два электрона определенным образом взаимодействуют друг с другом, находясь в определенных **отношениях**, а именно – отталкиваясь друг от друга с различной силой, зависящей от расстояния между ними. Но о каждом электроне можно сказать, что он обладает **свойством** отталкиваться от другого электрона. Чем поддерживается (обеспечивается) это свойство электронов, благодаря которому они могут быть в определенных отношениях друг с другом? В науке настоящего времени считается, что существует соответствующая материальная **структура**, которая называется «электрическое поле» и является одной из форм электромагнитного поля. Возникает вопрос – «а может ли эта структура существовать **независимо** от электрона?» Уже давно Герцем и Максвеллом получен однозначный положительный ответ на этот вопрос: «Да, может, и это – электромагнитные волны».

*Таким образом, **свойство** может существовать как некая материальная **структура** отдельно и независимо от объекта, свойством которого оно являлось и благодаря которому этот объект вступал в определенные **отношения** с другими объектами, обладавшими тем же свойством [1].*

Так нам светят звезды, которых, давно уже нет. Мы восхищаемся произведениями художников, поэтов и музыкантов давно прошедших времен.

*Итак, существуют различные точки зрения на одно и то же, которое, при различных условиях, может рассматриваться либо как определенные **отношения** объектов, либо как **свойства** этих объектов, на которых основаны их отношения, либо как самостоятельно существующая материальная **структура** [1].*

Каждая точка зрения имеет право на существование, но, по-видимому, каждая последующая из этих точек зрения является более глубокой, чем предыдущая.

5.4 Понятия «факт», «смысл», «мысль» в рамках когнитивной теории

Ключевым для когнитивной концепции является понятие факта.

***Под фактом** будем понимать соответствие элементов разных уровней интеграции-иерархии процессов познания, обнаруженное на опыте [1].*

***Факт** рассматривается как квант (частица, составная часть) смысла. Это является основой для формализации смысла [1].*

***Смысл** представляет собой «разность потенциалов» между смежными уровнями интеграции-иерархии в системе обработки информации в процессах познания [1].*

***Мысль** является операцией выявления смысла из фактов [1].*

***Мышление** есть процесс, состоящий из ряда взаимосвязанных по смыслу мыслей [1].*

Но существуют различные формы мыслей, которые перед разработкой методов формализации и программной реализации необходимо классифицировать и выявить среди них основные, т.е. такие, к которым сводятся все остальные или по крайней мере большинство из них. Как уже отмечалось выше, сделать это предлагается на основе базовой когнитивной концепции.

5.5 Иерархия задач обработки данных: «мониторинг», «анализ», «прогнозирование», «управление» в рамках когнитивной теории

Существует определенная иерархия задач обработки данных, информации и знаний [1]:

- ***Мониторинг** – накопление данных по ряду показателей об объекте управления с привязкой ко времени.*
- ***Анализ** – выявление смысла в данных, т.е. выявление в них причинно-следственных взаимосвязей.*
- ***Прогнозирование** – использование смысла причинно-следственных зависимостей в предметной области для предсказания поведения объекта управления в условиях действия определенных факторов.*
- ***Управление** – использования знаний для достижения определенных целей управления:*
 - *сохранение стабильного функционирования объекта управления;*
 - *перевод объекта управления в заранее заданное целевое состояние.*

Таким образом, управление – это высшая форма обработки и использования информации.



Рисунок 5.1 - Иерархическая структура обработки информации

В основу главы 5 положен материал учебного пособия [1].

6. КОГНИТИВНАЯ СТРУКТУРИЗАЦИЯ ЗНАНИЙ ОБ ИССЛЕДУЕМОМ ОБЪЕКТЕ И ВНЕШНЕЙ СРЕДЫ

6.1 Когнитивная структуризация знаний об исследуемом объекте и внешней среды на основе PEST-анализа

Отбор базисных факторов проводится путем применения *PEST-анализа*, выделяющего четыре основные группы факторов (аспекта), определяющих поведение исследуемого объекта (рисунок 6.1) [1]:

- *Policy* – политика;
- *Economy* – экономика;
- *Society* – общество (социокультурный аспект);
- *Technology* – технология.

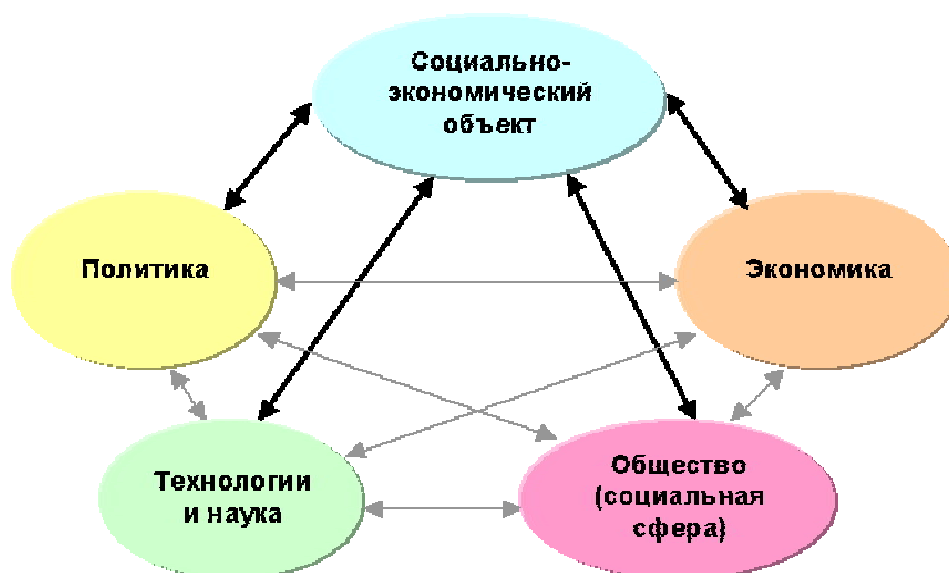


Рисунок 6.1 - Факторы PEST-анализа

Для каждого конкретного сложного объекта существует свой особый набор наиболее существенных факторов, определяющих его поведение и развитие.

PEST-анализ можно рассматривать как вариант системного анализа, т.к. факторы, относящиеся к перечисленным четырем аспектам, в общем случае тесно взаимосвязаны и характеризуют различные иерархические уровни общества, как системы.

В этой системе есть детерминирующие связи, направленные с нижних уровней иерархии системы к верхним (наука и технология влияет на экономику, экономика влияет на политику), а также обратные и

межуровневые связи. Изменение любого из факторов через эту систему связей может влиять на все остальные.

Эти изменения могут представлять угрозу развитию объекта, или, наоборот, предоставлять новые возможности для его успешного развития.

6.2 Ситуационный анализ проблем на базе SWOT-анализа

Проблемы и противоречия выделенные с помощью PEST – анализа можно разрешить на следующем шаге – SWOT-анализе. *Ситуационный анализ проблем, SWOT-анализ* (рисунок 6.2) [1]:

- *Strengths* – сильные стороны;
- *Weaknesses* – недостатки, слабые стороны;
- *Opportunities* – возможности;
- *Threats* – угрозы.

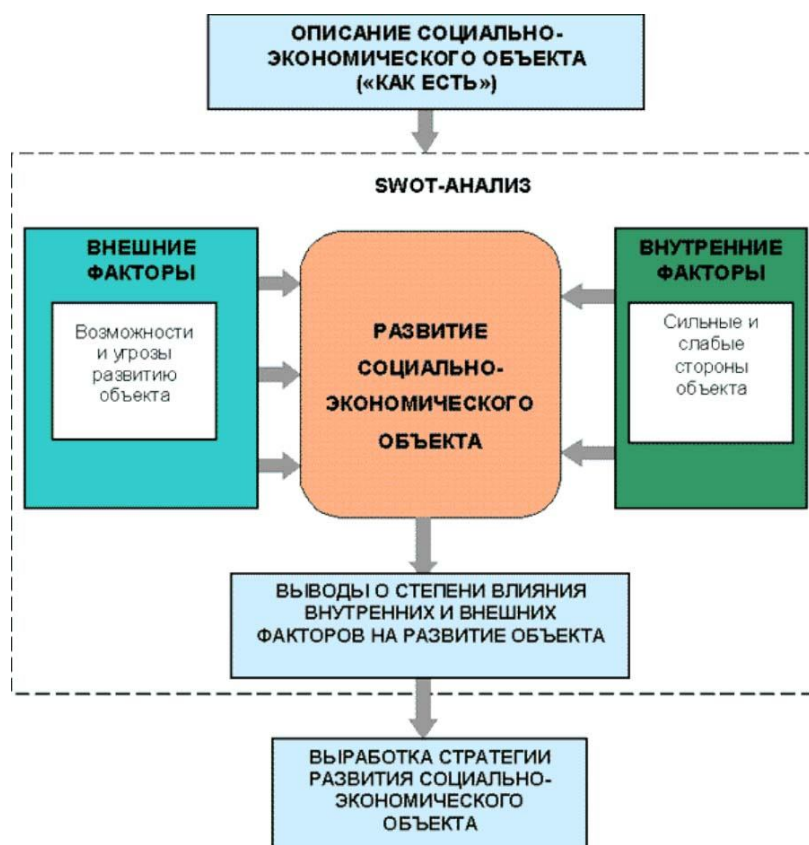


Рисунок 6.2 - Факторы SWOT-анализа

Он включает анализ сильных и слабых сторон развития исследуемого объекта в их взаимодействии с угрозами и возможностями и позволяет определить актуальные проблемные области, узкие места, шансы и опасности, с учетом факторов внешней среды.

Возможности определяются как обстоятельства, способствующее благоприятному развитию объекта [1].

Угрозы – это ситуации, в которых может быть нанесен ущерб объекту, например может быть нарушено его функционирование или он может лишиться имеющихся преимуществ [1].

На основании анализа различных возможных сочетаний сильных и слабых сторон с угрозами и возможностями формируется *проблемное поле* исследуемого объекта.

Проблемное поле – это совокупность проблем, существующих в моделируемом объекте и окружающей среде, в их взаимосвязи друг с другом [1].

Наличие такой информации – основа для определения целей (направлений) развития и путей их достижения, выработки стратегии развития.

Когнитивное моделирование на основе проведенного ситуационного анализа позволяет подготовить альтернативные варианты решений по снижению степени риска в выделенных проблемных зонах, прогнозировать возможные события, которые могут тяжелее всего отразиться на положении моделируемого объекта.

6.3 Этапы когнитивной технологии

Этапы когнитивной технологии и их результаты, представленные на рисунке 6.2, конкретизированы в таблице 6.1 [1].

Таблица 6.1 – Этапы когнитивной технологии и результаты ее применения

Наименование этапа	Форма представления результата
1. Когнитивная (познавательная-целевая) структуризация знаний об исследуемом объекте и внешней для него среды на основе PEST-анализа и SWOT-анализа. Анализ исходной ситуации вокруг исследуемого объекта с выделением базисных факторов, характеризующих экономические, политические и др. процессы, протекающие в объекте и в его макроокружении и влияющих на развитие объекта. - Выявление факторов, характеризующих сильные и слабые стороны исследуемого объекта. - Выявление факторов, характеризующих возможности и угрозы со стороны внешней среды объекта. - Построение проблемного поля исследуемого объекта.	Отчет о системном концептуальном исследовании объекта и его проблемной области
2. Построение когнитивной модели развития объекта – формализация знаний, полученных на этапе когнитивной структуризации. - Выделение и обоснование факторов. - Установление и обоснование взаимосвязей между факторами. - Построение графовой модели.	Компьютерная когнитивная модель объекта в виде ориентированного графа (и матрицы взаимосвязей факторов)

Наименование этапа	Форма представления результата
3. Сценарное исследование тенденций развития ситуации вокруг исследуемого объекта - Определение цели исследования. - Задание сценариев исследования и их моделирование. - Выявление тенденций развития объекта в его макроокружении. - Интерпретация результатов сценарного исследования.	Отчет о сценарном исследовании ситуации, с интерпретацией и выводами
4. Разработка стратегий управления ситуацией вокруг исследуемого объекта. - Определение и обоснование цели управления. - Решение обратной задачи. - Выбор стратегий управления и упорядочивание их по критериям: возможности достижения цели; риска потери управления ситуацией; риска возникновения чрезвычайных ситуаций.	Отчет о разработке стратегий управления с обоснованием стратегий по разным критериям качества управления
5. Поиск и обоснование стратегий достижения цели в стабильных или изменяющихся ситуациях. Для стабильных ситуаций: - выбор и обоснование цели управления; - выбор мероприятий (управлений) для достижения цели; - анализ принципиальной возможности достижения цели из текущего состояния ситуации с использованием выбранных мероприятий; - анализ реальных ограничений на реализацию выбранных мероприятий; - анализ и обоснование реальной возможности достижения цели; - выработка и сравнение стратегий достижения цели по: ○ близости результатов управления к намеченной цели; ○ затратам (финансовым, физическим и т.п.); ○ по характеру последствий (обратимые, необратимые) от реализации этих стратегий в реальной ситуации; ○ по риску возникновения чрезвычайных ситуаций. Для изменяющихся ситуаций: - выбор и обоснование текущей цели управления; - по отношению к текущей цели справедливы предыдущие п.п. для стабильных ситуаций; - анализ изменений, происходящих в ситуации, и их отображение в графовой модели ситуации. Переход к п. 1.	Отчет о разработке стратегий достижения цели в стабильных или изменяющихся ситуациях
6. Разработка программы реализации стратегии развития исследуемого объекта на основе динамического имитационного моделирования. - Распределение ресурсов по направлениям и во времени. - Координация. - Контроль за исполнением.	Программа реализации стратегии развития объекта. Компьютерная имитационная модель развития объекта

Применение когнитивных технологий открывает новые возможности прогнозирования и управления в различных областях [1]:

- *в экономической сфере* это позволяет в сжатые сроки разработать и обосновать стратегию экономического развития предприятия, банка, региона или даже целого государства с учетом влияния изменений во внешней среде;
- *в сфере финансов и фондового рынка* – учесть ожидания участников рынка;
- *в военной области и области информационной безопасности* – противостоять стратегическому информационному оружию, заблаговременно распознавая конфликтные структуры и вырабатывая адекватные меры реагирования на угрозы.

Когнитивные технологии автоматизируют часть функций процессов познания, поэтому они с успехом могут применяться во всех областях, в которых востребовано само познание. Вот лишь некоторые из этих областей [1]:

- Модели и методы интеллектуальных информационных технологий и систем для создания геополитических, национальных и региональных стратегий социально-экономического развития.
- Ситуационный анализ и управление развитием событий в кризисных средах и ситуациях.
- Информационный мониторинг социально-политических, социально-экономических и военно-политических ситуаций.
- Выработка аналитических сценариев развития проблемных ситуаций и управления ими.
- Подготовка рекомендаций по решению первоочередных стратегических проблем на основе компьютерной системы анализа проблемных ситуаций.
- Мониторинг проблем в социально-экономическом развитии корпорации, региона, города, государства.
- Анализ и управление развитием ситуации на потребительском рынке.

В основу главы 6 положен материал учебного пособия [1].

ЧАСТЬ 3. ПРЕДСТАВЛЕНИЕ И ВЫВОД ЗНАНИЙ

7. МОДЕЛИ ПРЕДСТАВЛЕНИЯ ЗНАНИЙ

7.1 Декларативные и процедурные знания

Любая предметная область характеризуется своим набором понятий и связей между ними, своими законами, связывающими между собой объекты данной предметной области, своими процессами, событиями. И конечно, каждая предметная область имеет свои, специфические методы решения задач. Знания о предметной области и способах решения в ней задач весьма разнообразны. Возможны различные классификации этих знаний.

В общем случае знания подразделяются на:

- *Процедурные знания* описывают последовательности действий, которые могут использоваться при решении задач. Это, например, программы для ЭВМ, словесные записи алгоритмов, инструкция по сборке некоторого изделия.
- *Декларативные знания* — это все знания, не являющиеся процедурными, например статьи в толковых словарях и энциклопедиях, формулировки законов в физике, химии и других науках и т.п. В отличие от процедурных знаний, отвечающих на вопрос: «Как сделать X?», декларативные знания отвечают, скорее, на вопросы: «Что есть X?» или «Какие связи имеются между X и Y?», «Почему X?» и т.д.

Языки представления знаний можно разделить на типы по формальным моделям представления знаний, которые лежат в их основе:

- логическая,
- сетевая,
- фреймовая,
- продукционная.

Ниже будут рассмотрены данные языки представления данных.

7.2 Логическая модель представления знаний

Логическая модель представляет собой формальную систему в которой все знания о предметной области описываются в виде формул этого исчисления или правил вывода. Описание в виде формул дает возможность представить декларативные знания, а правила вывода — процедурные знания.

Логическая модель знаний строится на базе предикатов.

Логика предикатов является развитием алгебры логики (или логики высказываний). В логике высказываний для обозначения фактов используются буквы (имена или идентификаторы или фразы), не имеющие структуры (используемые как атомарные объекты), и принимающие значения «1» или «0» («да» или «нет»). То, что фразы имеют атомарный характер, не позволяет обнаружить похожесть их смысла. Например, высказывания «расстояние от Земли до Солнца – 150 млрд. км» и «расстояние от Земли до Марса – 60 млн. км» имеют похожий смысл, но абсолютно разные в логике высказываний.

В логике предикатов факты обозначаются n -арными логическими функциями – предикатами $F(x_1, x_2, \dots, x_m)$, где F – имя предиката (функтор) и x_i – аргументы предиката [3].

***Предикатом** называется функция, принимающая два значения ИСТИНА и ЛОЖЬ – и предназначенная для выражения свойств объекта или связей между ними.*

Имена предикатов неделимы, т.е. являются так называемыми *атомами*. Аргументы могут быть атомами или функциями $f(x_1, x_2, \dots, x_m)$, где f – имя функции, а $x_1, x_2, \dots, x_m$, так же как и аргументы предикатов являются переменными или константами предметной области [3].

В результате интерпретации (по другому, конкретизации) предиката функторы и аргументы принимают значения констант из предметной области (строк, чисел, структур и т.д.). При этом следует различать интерпретацию на этапе описания предметной области (создания программ и баз знаний) и на этапе решения задач (выполнения программ с целью корректировки или пополнения баз знаний) [3].

Выше приведенные примеры высказываний в виде предикатов будут выглядеть как «расстояние (Земля, Солнце, 150000000000)» и «расстояние (Земля, Марс, 60000000)». Так как они имеют определенную структуру, их можно сравнивать по частям, моделируя работу с содержащимся в них смыслом.

Предикат с арностью $n > 1$ может использоваться в инженерии знаний для представления n -арного отношения, связывающего между собой n сущностей (объектов) – аргументов предиката [3].

Например, предикат «отец («Иван», «Петр Иванович»)» может означать, что сущности «Иван» и «Петр Иванович» связаны родственным отношением, а именно, последний является отцом Ивана или наоборот - уточнение семантики (смысла) этого предиката связано с тем, как он используется, т.е. в каких операциях или более сложных отношениях он участвует, и какую роль в них играют его 1-й и 2-й аргументы.

Предикат «компьютер (память, клавиатура, процессор, монитор)» может обозначать понятие «компьютер» как отношение, связывающее между собой составные части компьютера, предикат «внутри (процессор_Pentium, компьютер)» – то, что внутри компьютера находится процессор Pentium.

Предикат с арностью $n=1$ может представлять свойство сущности (объекта), обозначенного аргументом или характеристику объекта, обозначенного именем предиката [3]. Например:

кирпичный (дом),
оценка (5),
улица («Красный проспект»),
дата_рождения («1 апреля 1965 г.»),
быстродействие («1 Мфлопс»).

Предикат с арностью $n=0$ (без аргументов) может обозначать событие, признак или свойство, относящееся ко всей предметной области. Например: «конец работы» [3].

При записи формул (выражений) помимо логических связок «конъюнкция» ($\&$), «дизъюнкция» ($\vee$), «отрицание» ($\neg$), «следование» («импликация») ($\rightarrow$), заимствованных из логики высказываний, в логике предикатов используются кванторы всеобщности ($\forall$) и существования ($\exists$) [3].

Например, выражение:

$\forall (x, y, z) (\text{отец}(x, y) \& \text{мать}(x, z)) \rightarrow \text{родители}(x, y, z)$,

означает, что для всех значений x, y, z из предметной области справедливо утверждение «если y – отец и z – мать x , то y и z – родители x ».

Выражение

$(\exists x) (\text{студент}(x) \& \text{должность}(x, \text{«инженер»}))$,

означает, что существует хотя бы один студент, который работает в должности инженера.

Переменные при кванторах называются связанными переменными в отличие от свободных переменных [3]. Например, в выражении

$(\forall x) (\text{владелец}(x, y) \rightarrow \text{частная_собственность}(y))$,

x – связанная переменная, y – свободная переменная.

Логика предикатов 1-го порядка отличается от логик высших порядков тем, что в ней запрещено использовать выражения (формулы) в качестве аргументов предикатов.

Примером логического знания формализованного в виде логики предикатов высших порядков является знание: «Когда температура в печи достигает 120^0 и прошло менее 30 мин с момента включения печи, давление не может превосходить критическое. Если с момента включения печи прошло более 30 мин, то необходимо открыть вентиль №2».

Логическая модель представления этого знания имеет вид:

$$P(p = 120) T(t < 30) \rightarrow (D < D_{кр});$$

$$P(p = 120) T(t > 30) \rightarrow F(№2).$$

В этой записи использованы следующие обозначения для предикатов:

$P(p = 120)$ — предикат, становящийся истинным, когда температура достигает 120 градусов,

$T(t < 30)$ — предикат, остающийся истинным в течение 30 мин с начала процесса;

$T(t > 30)$ — предикат, становящийся истинным по истечении 30 мин с начала процесса;

$(D < D_{кр})$ — утверждение о том, что давление ниже критического;

$F(№2)$ — команда открыть вентиль №2.

Кроме того, в записях использованы типовые логические связки конъюнкции ($\vee$), импликации ($\wedge$) и логического следования ($\rightarrow$).

Первая строчка в записи представляет декларативные знания, а вторая — процедурные.

Решение задач в логике предикатов сводится к доказательству целевого утверждения в виде формулы или предиката (теоремы), используя известные утверждения (формулы) или аксиомы.

В конце 60-х годов Робинсоном для доказательства теорем в логике предикатов был предложен метод резолюции, основанный на доказательстве «от противного». Целевое утверждение инвертируется, добавляется к множеству аксиом и доказывается, что полученное таким образом множество утверждений является несовместным (противоречивым). Для выполнения доказательства методом резолюции необходимо провести определенные преобразования над множеством утверждений, а именно, привести их к совершенной конъюнктивной нормальной форме (СКНФ). СКНФ представляет собой набор (конъюнкцию) дизъюнктов без кванторов. Кванторы всеобщности подразумеваются, а кванторы существования заменяются на перечисление формул (или предикатов) со всеми константами из предметной области, для которых формула истинна. Например:

«отец (Иван, Петр)», «отец (Иван, Степан)» и т.д.

Языки представлений знаний логического типа широко использовались на ранних стадиях развития интеллектуальных систем, но вскоре были вытеснены (или во всяком случае сильно потеснены) языками других типов. Объясняется это громоздкостью записей, опирающихся на классические логические исчисления. При формировании таких записей легко допустить ошибки, а поиск их очень сложен. Отсутствие наглядности, удобочитаемости (особенно для тех, чья деятельность не связана с точными науками) затрудняло распространение языков такого типа.

7.3 Псевдофизические модели представления знаний

Недостатки классической логики и основанной на ней логики предикатов первого порядка как метода представления знаний об окружающем мире привели к появлению *псевдофизических логик*. В их основе лежит представление нечетких или размытых понятий в виде так называемых лингвистических переменных, придуманных Заде для того, чтобы приблизить семантику (смысл) знака к семантике, которая вырабатывается в мозгу человека в процессе его обучения (опыта) [3].

Для этого множество образов (десигнатов), с которыми должна оперировать интеллектуальная система, представляется в виде точек на шкалах. Например, можно рассматривать шкалы «возраст» (в годах), «расстояние до объекта» (в м или км) и т.п. С каждой шкалой связано множество знаковых значений лингвистической переменной. Например, со шкалой «возраст» могут быть связаны следующие значения одноименной лингвистической переменной: «юный», «молодой», «зрелый», «пожилой», «старый», «дряхлый». Со шкалой «расстояние» – «вплотную», «очень близко», «близко», «рядом», «недалеко», «далеко», «очень далеко». Взаимосвязь между этими двумя представлениями (множеством точек на шкале и множеством знаковых значений) задается с помощью функции принадлежности $\mu_x(t)$, где x – значение лингвистической переменной, t – значение на шкале.

Значение функции принадлежности интерпретируется как вероятность того, что значение t на шкале можно заменить знаком x или наоборот. Очевидно, что можно пронормировать значения функции принадлежности в соответствии с формулой

$$\sum_x \mu_x(t) = 1,$$

или в соответствии с

$$\sum_t \mu_x(t) = 1.$$

На рисунке 7.1 приведен пример описания лингвистической переменной возраст. Здесь каждая кривая описывает ее одно символьное значение.

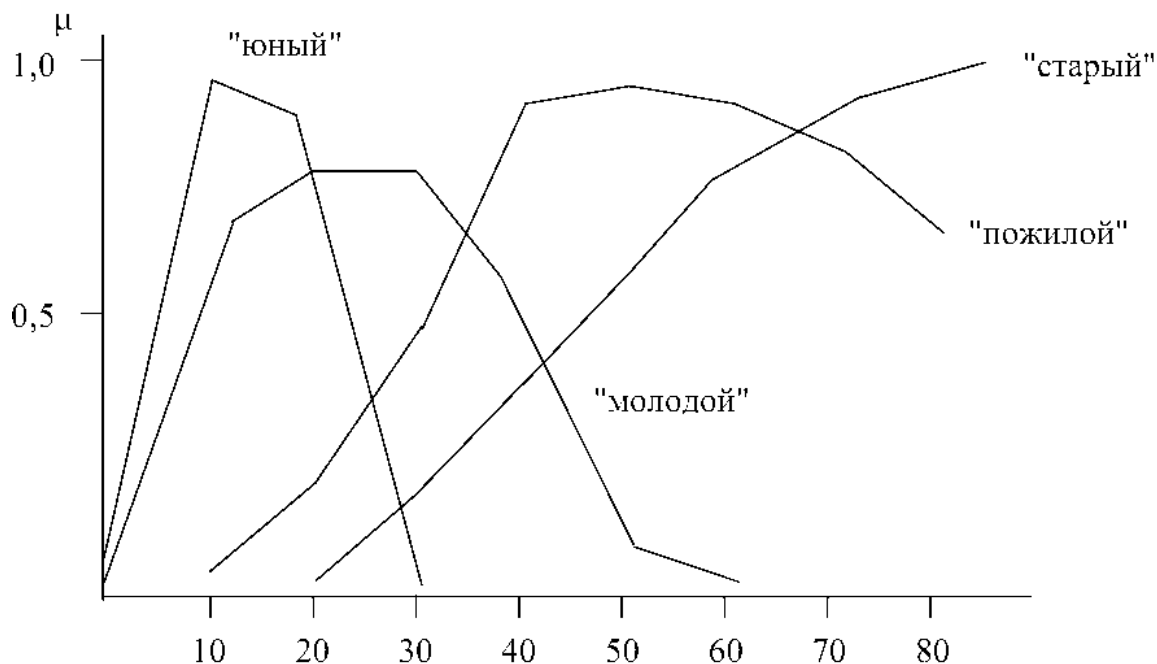


Рисунок 7.1 - Пример описания лингвистической переменной «возраст»

Наиболее используемыми псевдофизическими логиками являются пространственная, временная и каузальная (причинно-следственная).

Пространственная логика может быть разбита на логики статическая и динамическая, взаимного расположения объектов, расположения объектов в пространстве (расстояний и направлений).

Логика взаимного расположения объектов, расстояний и направлений делится на метрическую и топологическую логики. В отличие от метрической топологическая логика не связана с метрической шкалой.

Метрические шкалы подразделяются на экзоцентрические и эндоцентрические, относительные и абсолютные. Экзоцентрические шкалы имеют началом координат точку, связанную с самой интеллектуальной системой. Примером такой шкалы является шкала для описания лингвистической переменной «Расстояние до объекта» в логике расстояний. Ее символическими значениями могут быть следующие: «рядом», «близко», «недалеко», «далеко», «очень далеко» и т.п.

Эндоцентрическая шкала имеет началом координат точку вне системы. Примером такой шкалы является шкала для описания лингвистической переменной «расстояние между двумя объектами» в той же логике расстояний. Относительные шкалы имеют изменяемую точку отсчета (начало координат), а абсолютные — неизменную (обычно, подразумеваемую, т.е. явно не заданную).

Логика направлений оперирует с понятиями «справа», «слева», «впереди», «сзади» или «на восток», «на запад» и т.п.

В логике взаимного расположения объектов описываются следующие базовые отношения: унарные – «иметь горизонтальное положение», «иметь вертикальное положение», бинарные – «находиться внутри», «находиться вне».

Из базовых отношений с помощью логических связок строятся производные отношения, такие как «не соприкасаться» (отрицание «соприкасаться»), «быть вместе..» (следствие от «находиться там же..»), «висеть» (конъюнкция «иметь вертикальное положение» и «висеть на...»), «стоять» (конъюнкция «иметь вертикальное положение» и «иметь точку опоры на..») и т.п. Эти отношения описываются в виде правил, определяющих с помощью импликации сложное отношение через базовые. Кроме того, в псевдофизической логике в виде правил описываются свойства отношений и взаимосвязи между ними.

Например, свойства рефлексивности (например, рядом (x, x)), симметричности (например, рядом $(x, y) \rightarrow$ рядом (y, x)) и транзитивности (например, выше $(x, y) \rightarrow$ выше $(x, z) \&$ выше (z, y)).

Временная псевдофизическая логика имеет дело с отношениями «происходить одновременно», «пересекаться во времени» (n -арные), «быть раньше», «быть позже» (бинарные), «давно», «недавно», «скоро» (унарные) и т.п.

Более подробно особенности использования нечеткой логики при анализе и представлении данных и знаний представлены в главах 9 и 17.

7.4 Сетевая модель представления знаний

В основе сетевой модели лежит идея о том, что любые знания можно представить в виде совокупности объектов (понятий) и связей (отношений) между ними.

*Известно, что любую конкретную ситуацию в реальном мире, всегда можно представить в виде совокупности взаимосвязанных **понятий**. Причем число базовых **отношений** не может быть бесконечным (оно заведомо меньше 300); все остальные отношения выражаются через базовые в виде их комбинаций. Эта гипотеза служит основой утверждения о том, что семантические сети являются универсальным средством для представления знаний в интеллектуальных системах.*

Семантической сетью называется ориентированный граф с помеченными вершинами и дугами, где вершинам соответствуют конкретные объекты, дугам - отношения между ними.

Семантические сети являются весьма мощным средством представления знаний. Однако для них характерны неоднозначность представлений знаний и неоднородность связей.

В семантических сетях используются три основных типа объектов:

1. **Понятия** представляют собой сведения об абстрактных или конкретных (физических) объектах предметной области.
2. **События** - это действия, которые могут внести изменения в предметную область, т.е. изменить состояние предметной области.
3. **Свойства** используются для уточнения понятий и событий. Применительно к понятиям свойства описывают их особенности или характеристики, например - цвет, размер, качество. Применительно к событиям свойства - продолжительность, место, время и т.д.

Рассмотрим, например, текст, содержащий некоторые декларативные знания: «Слева от станка расположен приемный бункер. Расстояние до него равно двум метрам. Справа от станка — бункер готовой продукции. Он находится рядом со станком. Робот перемещается параллельно станку и бункерам на расстоянии 1 м».

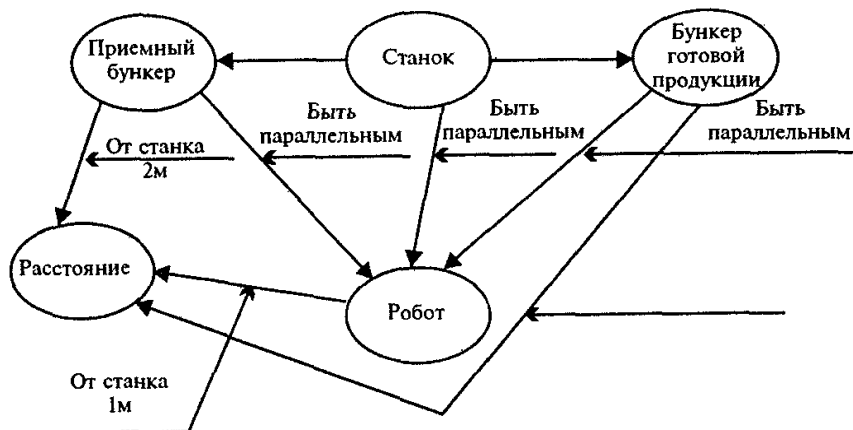


Рисунок 7.2 - Пример сетевого представления совокупности знаний

На рисунке 7.2 показано сетевое представление совокупности знаний в виде семантической сети. **Понятия и объекты**, встречающиеся в тексте, представлены в виде **сети**, а **отношения** — в виде **дуг**, связывающих соответствующие вершины.

В семантической сети возможно ввести различные виды отношений между объектами.

Атрибутивные отношения - это отношения между объектом и свойством, например, цвет, размер, форма, модификация и т.д. На

рисунке 7.3 приведен пример семантической сети с использованием атрибутивных отношений.

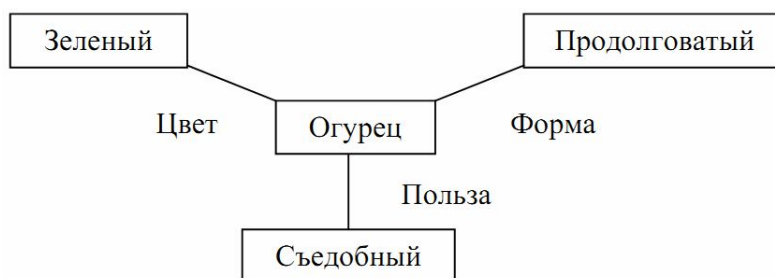


Рисунок 7.3 - Пример атрибутивных отношений

Теоретико-множественные (иерархические) отношения - это отношения между элементом множества (подмножества) и множеством, отношение части и целого, отношение между элементом класса и классом и т.п. Данный тип отношений используется для хранения в базе знаний сложных (составных или иерархических) понятий. Этот тип отношений иллюстрируется на рисунке 7.4.

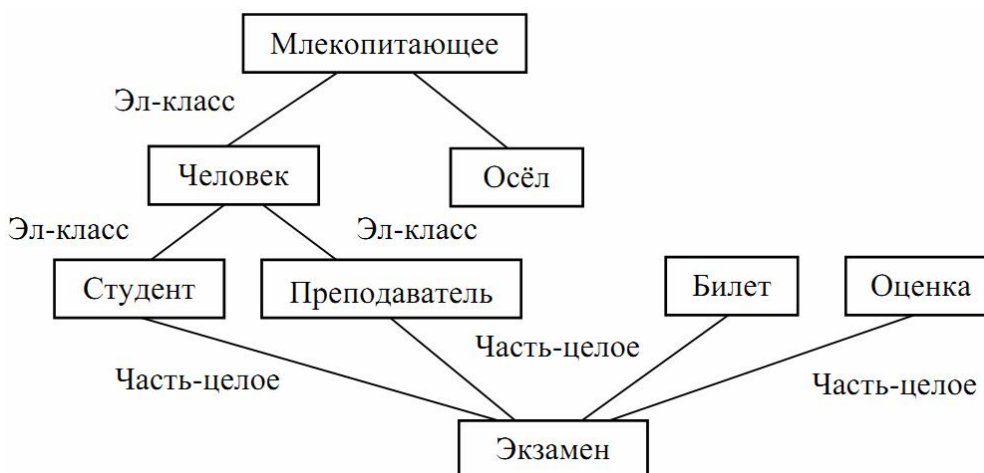


Рисунок 7.4 - Пример теоретико-множественных отношений

Квантифицированные отношения – это логические кванторы общности и существования. Они используются для представления знаний типа: «любой студент должен посещать лабораторные занятия», «существует хотя бы один язык программирования, который должен знать любой выпускник НГТУ».

К наиболее распространенным лингвистическим отношениям относятся

- падежные,
- атрибутивные.

Падежными (или ролевыми) отношениями могут являться следующие:

– агент, отношение между событием и тем, что (или кто) его вызывает, например, отношение между «завинчиванием» (гайки) и рукой;

– объект, отношение между событием и тем, над чем производится действие, например, между «завинчиванием» и «гайкой»;

– условие, отношение, указывающее логическую зависимость между событиями, например, отношение между «завинчиванием» (гайки) и «сборкой» (узла);

– инструмент, отношение между событием и объектом, с помощью которого оно совершается, например, между «завинчиванием» и «верстаком».

К базе знаний представленной семантической сетью, возможны следующие основные типы запросов:

- запрос на существование;
- запрос на перечисление.

При построении интеллектуальных банков знаний обычно используют разделение знаний на:

- интенциональные,
- экстенциональные.

***Экстенциональная семантическая сеть** (или **К-сеть**) содержит информацию о фактах, о конкретных объектах, событиях, действиях.*

***Интенциональная семантическая сеть** (или **А-сеть**) содержит информацию о закономерностях, потенциальных взаимосвязях между объектами, неизменяемую информацию об объектах, т.е. модель мира.*

Экстенциональные (конкретные) знания создаются и обновляются в процессе работы с банком данных, а интенциональные (абстрактные) изменяются редко. Первые можно назвать экземпляром, а последние – моделью (схемой) базы данных.

Запрос к банку знаний, обрабатываемый системой управления базой знаний, представляет собой набор фактов (ситуацию), при описании которого допускается использование переменных вместо значений атрибутов, имен понятий, событий и отношений. Запрос можно представить в виде графа, в котором вершины, соответствующие переменным, не определены. Поиск ответа сводится к задаче изоморфного вложения графа запроса (или его подграфа) в семантическую сеть.

Запрос на существование не содержит переменных и требует ответа типа ДА, если изоморфное вложение графа запроса в семантическую сеть удалось, и НЕТ – в противоположном случае. При обработке запроса на перечисление происходит поиск всех возможных изоморфных графу запроса подграфов в

семантической сети, а также присваивание переменным в запросе значений из найденных подграфов.

Кроме того, на семантических сетях можно использовать методы доказательства, используемые в логике предикатов, т.к. семантическая сеть легко преобразуется в логику предикатов 1-го порядка (каждое ребро можно представить в виде бинарного предиката).

Достоинством семантических сетей является их универсальность, достигаемая за счет выбора соответствующего применению набора отношений. В принципе с помощью семантической сети можно описать сколь угодно сложную ситуацию, факт или предметную область.

В виде семантической сети можно представить псевдофизическую логику.

С другой стороны, семантическую сеть можно записать в виде набора предикатов 1-го порядка. В этом случае именами предикатов будут являться имена отношений (ребер графа), а атрибутами – связываемые отношением узлы семантической сети.

Недостатком семантических сетей является их практическая необозримость при описании модели мира реального уровня сложности. При этом появляется проблема размещения семантической сети в памяти ЭВМ. Если ее размещать всю в оперативной (виртуальной) памяти, на ее сложность накладываются жесткие ограничения. Если размещать во внешней памяти, появляется проблема, как подгружать необходимые для работы участки.

7.5 Фреймовая модель представления знаний

При автоматизации процесса использования и представления знаний неоднозначность и неоднородность заметно усложняют процессы, протекающие в интеллектуальных системах. Поэтому вполне естественно желание как-то унифицировать форму представления знаний, сделать ее максимально однородной. Одним из способов решения этой задачи в искусственном интеллекте послужил переход к специальному представлению вершин в сети и унификация связей между вершинами (фреймами).

Фреймы используются в системах искусственного интеллекта (например, в экспертных системах) как одна из распространенных форм представления знаний.

Фрейм — это минимально возможное описание сущности какого-либо явления, события, ситуации, процесса или объекта. Минимально возможное означает, что при дальнейшем упрощении описания теряется его полнота, оно перестает определять ту единицу знаний, для которой оно предназначено.

Другими словами, **фрейм** – это структура, описывающая фрагмент базы знаний, который в какой-то степени рассматривается и обрабатывается обособленно от других фрагментов. Другие фрагменты, с которыми он связан, во фрейме представлены только их именами (идентификаторами) так же как и он в них.

В виде фрейма может описываться некоторый объект, ситуация, абстрактной понятие, формула, закон, правило, визуальная сцена и т.п. Понятие фрейма неразрывно связано с абстрагированием и построением иерархии понятий.

Фрейм имеет почти однородную структуру и состоит из стандартных единиц, называемых слотами. Каждая такая единица — **слот** — содержит название и свое значение. Изображается фрейм в виде цепочки:

$$\text{Фрейм} = \langle \text{слот } 1 \rangle, \langle \text{слот } 2 \rangle, \dots, \langle \text{слот } N \rangle.$$

Фреймы подразделяются на два типа:

- фреймы-прототипы (или классы),
- фреймы-примеры (или экземпляры).

Фреймы-прототипы используются для порождения фреймов-примеров.

В качестве примера рассмотрим фрейм для понятия «взятие»:

фрейм «Взятие»:	(Субъект, X_1);
	(Объект, X_2);
	(Место, X_3);
	(Время, X_4);
	(Условие, X_5).

В этом фрейме указаны имена слотов (субъект, объект и т.д.), но вместо их значений стоят переменные (X_1 , X_2 и т.д.). Такой фрейм называется фреймом-прототипом, или **протофреймом**.

Протофреймы хранят знания о самом понятии. Например, понятие «взять» связано с наличием слотов с указанными именами. Взятие осуществляет X_1 в месте X_3 во время X_4 , если выполнено условие X_5 . Берет X_1 нечто, обозначенное как X_2 . Подставляя вместо всех переменных конкретные значения, получим конкретный факт-описание:

фрейм «Взятие»:	(Субъект, Робот);
	(Объект, Деталь);
	(Место, Приемный бункер)
	(Время, X_4);
	(Условие, В бункере есть деталь, а у робота ее нет).

В нашем примере, наверное, основными для фрейма «взятие» можно считать слоты с именами «субъект» и «объект».

Фреймы, в которых обозначены все основные слоты (они каким-либо образом помечаются в описании фрейма), называются **фреймами-экземплярами**, или **экзофреймами**.

Поскольку в состав фрейма могут входить слоты с именами действий, фреймы годятся для представления как декларативных, так и процедурных знаний.

Чтобы представить семантическую сеть в виде совокупности фреймов, надо уметь представлять отношения между вершинами сети. Для этого также используются слоты фреймов. Эти слоты могут иметь имена вида «Связь Y », где Y — имя того отношения (его тип), которое устанавливает данный фрейм-вершина с другим фреймом-вершиной.

В качестве значения слота может выступать новый фрейм, что позволяет на множестве фреймов осуществлять иерархическую классификацию. Это очень удобное свойство фреймов, так как человеческие знания, как правило, упорядочены по общности.

7.6 Продукционная форма представления знаний

Основу продукционной модели составляют системы productions. Каждая продукция в наиболее общем виде записывается как стандартное выражение следующего вида:

«Имя продукции»:	Идентификатор правила;
	Область применимости (имя сферы);
	Условие применимости ядра;
	Ядро: если A , то B ;
	Постусловие.

Основная часть продукции — ее ядро имеет вид:

«Если A , то B »,

где A — посылка правила, B — заключение правила, имеющие разные значения.

Остальные элементы, образующие продукцию, носят вспомогательный характер.

В наиболее простом виде продукция может состоять только из имени (например, ее порядкового номера в системе productions) и ядра.

Наиболее часто используемая форма интерпретации продукции — логическая, при которой A является множеством элементарных условий, связанных логическими связками «И», «ИЛИ» и «НЕТ», B — множеством элементарных заключений. При этом правило считается работавшим

(выполняется B), если посылка A истинна. Другой формой интерпретации ядра является вероятностная интерпретация, при которой правило срабатывает с некоторой вероятностью, зависящей от истинности посылки [3].

В качестве заключения B обычно применяется операция добавления факта в базу данных интеллектуальной системы с указанием меры достоверности получаемого факта. В качестве постусловия могут использоваться какие-либо дополнительные действия или комментарии, сопровождающие правило [3].

Имя сферы указывает ту предметную область, к которой относятся знания, зафиксированные в данной продукции. В интеллектуальной системе может храниться совокупность знаний (ее называют базой знаний), относящихся к разным областям (например, знания о различных заболеваниях человека или знания из различных разделов математики).

Случай, когда ядро продукции описывает причинно-следственную связь явлений A и B :

***«Если сверкнет молния,
то гремит гром».***

Пример когда A и B представляют собой некоторые действия:

***«Если в доме вспыхнул пожар,
то вызывайте пожарную команду».***

Пример когда A — это некоторые знания, а B — действие:

***«Если в путеводителе указано, что в городе есть театр,
то надо пойти туда».***

Возможны и другие варианты ядра продукции. Таким образом, при помощи ядер можно представлять весьма разнообразные знания.

Обычно при описании баз знаний или экспертных систем правила представляются в более наглядном виде, например [3]:

ПРАВИЛО 1:

ЕСЛИ Образование = Высшее

И

Возраст = Молодой

И

Коммуникабельность = Высокая

ТО

Шансы найти работу = Высокие КД = 0.9.

При срабатывании этого правила в базу данных интеллектуальной системы (например, экспертной системы) добавляется факт, означающий, что шансы найти работу высоки с достоверностью 0.9 или 90 % (значение коэффициента достоверности КД). Понятия «Образование», «Возраст», «Коммуникабельность» служат для задания условия (в данном случае, конъюнкции), при котором срабатывает правило [3].

Когда речь шла о различных A и B в ядрах продукций, то практически было показано, что в такой форме можно представлять как декларативные знания, так и процедурные, хотя сама форма продукций весьма удобна для задания именно процедурных знаний. Пример метаправила для гипотетической базы знаний, пример из которой был приведен ранее [3]:

```
ЕСЛИ
    Экономика = развивается
ТО
    Увеличить приоритет правила 1
```

Для представления нечетких знаний факты и правила в продукционных системах снабжаются коэффициентами достоверности (или уверенности), которые могут принимать значения из разных интервалов в разных системах (например, $\langle 0,1 \rangle$, $\langle 0, 100 \rangle$, $\langle -1,+1 \rangle$). Во втором случае можно говорить об уверенности в процентах, а в последнем случае – о задании коэффициентом уверенности меры ложности или истинности факта [3].

Рассмотренные модели представления знаний широко используются в современных интеллектуальных системах и прежде всего в экспертных системах. Каждая из форм представлений знаний может служить основой для создания языка программирования, ориентированного на работу со знаниями. В конце 80-х годов наметилась тенденция создавать комбинированные языки представления знаний. Чаще всего комбинируются фреймовые и продукционные модели.

В основу главы 3 положен материал учебного пособия [3].

8. МЕТОДЫ ПРИОБРЕТЕНИЯ И ИЗВЛЕЧЕНИЯ ЗНАНИЙ

8.1 Основные термины и определения в области приобретения знаний

Рассматривая методы приобретения знаний, будем использовать следующие термины: извлечение, получение, формирование, приобретение знаний и обучение БЗ. Определим сущность указанных терминов.

*Под **извлечением знаний** будем понимать процесс приобретения материализованных знаний из текстологических источников информации с помощью некоторой совокупности методов и процедур, позволяющих переходить от знаний в текстовой форме к их аналогам для ввода в базу знаний СИИ.*

***Получение знаний** — это процесс приобретения вербализуемых и невербализуемых знаний эксперта, основанный на использовании непосредственно им самим или инженером по знаниям соответствующих приемов, процедур, методов и инструментальных средств.*

***Формирование знаний** — это процесс автоматического приобретения (порождения) системой искусственного интеллекта или инструментальным средством нового и полезного знания из исходной и текущей информации, которое в явном виде не формируют эксперты, в целях освоения новых процедур решения прикладных задач на основе использования различных моделей машинного обучения.*

*Под **приобретением знаний** будем понимать процесс, основанный на переносе знаний из различных источников в базу знаний путем использования различных методов, моделей, алгоритмов и инструментальных средств.*

*Понятие **получение знаний** соотносится с понятиями извлечение, приобретение, формирование знаний как часть — целое. То есть включает в себя процессы извлечения, приобретения и формирования знаний.*

***Обучение базы знаний (БЗ)** — это процесс ввода (переноса) приобретенных знаний в систему искусственного интеллекта (СИИ) на основе применения совокупности методов, приемов и процедур в целях ее заполнения, расширения и модификации.*

*Термин **обучение** рассматривается как свойство базы данных, как совокупность методов, приемов и процедур ввода знаний в БЗ и как процесс переноса знаний в СИИ.*

Большинство методов извлечения и получения знаний основано на прямом диалоге с экспертом.

8.2 Методы приобретения знаний

Основной проблемой при разработке современных интеллектуальных систем является проблема приобретения знаний, т.е. преобразование разного вида информации (данных) из внешнего представления в представление в виде знаний, пригодное для решения задач, для которых создается интеллектуальная система. Эту проблему часто называют **проблемой извлечения знаний из данных** (в более общем виде, из внешнего мира), которая сводится к задаче обучения интеллектуальной системы [3].

Примерами задач приобретения знаний являются [3]:

- выявление причинно-следственных связей между атрибутами реляционной базы данных и формирование их в виде правил в продукционной экспертной системе;
- формирование программы (или правил) решения задачи (например, планирования производственного процесса или поведение робота) на основе примеров удачного планирования, вводимых в компьютер;
- выявление информативных признаков для классификации объектов, существенных с точки зрения решаемой задачи.

Обучающиеся системы можно классифицировать по двум основным признакам [3]:

1. *уровень, на котором происходит обучение:*

- 1.1. *обучение на символьном уровне (SLL – symbol level learning), при котором происходит улучшение представления знаний на основе опыта, полученного при решении задач,*
- 1.2. *обучение на уровне знаний (KLL – knowledge level learning), при котором происходит формирование новых знаний из существующих знаний и данных.*

2. *применяемый метод обучения:*

2.1. *аналитические методы обучения:*

- 2.1.1. *использующие глубинные (knowledge-rich) знания,*
- 2.1.2. *использующие поверхностные (knowledge-driven) знания;*

2.2. *эмпирические методы обучения:*

- 2.2.1. *использующие знания (knowledge-learning),*
- 2.2.2. *использующие данные (data-driven).*

На символьном уровне обучение сводится к манипулированию уже существующими структурами, представляющими знание, например, корректировка коэффициентов достоверности правил-продукций, изменение порядка расположения (просмотра) правил-продукций в базе знаний вводимого

пользователем описания решения задачи на достаточно формализованном языке, не сильно отличающимся от языка, на котором представляются знания в системе.

На уровне знаний обучение сводится к выявлению и формализации новых знаний. Например, из фактов

журавль умеет летать,
воробей умеет летать,
синица умеет летать,
журавль есть птица,
воробей есть птица,
синица есть птица,

система может сформулировать правило-продукцию:

Если X есть птица, то X умеет летать.

Помимо вышеуказанной классификации *в инженерии знаний известны три основных подхода к приобретению знаний* [3]:

1. *индуктивный вывод,*
2. *вывод по аналогии*
3. *обучение на примерах.*

В основе **индуктивного вывода** лежит процесс получения знаний из данных и/или других знаний (в продукционных системах – правил из фактов и/или других правил) [3].

Вывод по аналогии основан на задании и обнаружении аналогий между объектами (ситуациями, образами, постановками задачи, фрагментами знаний) и применением известных методов (процедур) к аналогичным объектам [3].

В основе **обучения на примерах** лежит демонстрация системе и запоминание ей примеров решения задач. Резкой границы между этими методами не существует, т.к. все они базируются на обобщении, реализованной в той или иной форме, т.е. реализуют переход от более конкретного знания (фактов) к более абстрактному знанию [3].

На рисунке 8.1 показана классификация обучающихся систем и взаимосвязи между понятиями, связанными с приобретением знаний приведенная в работе [3].

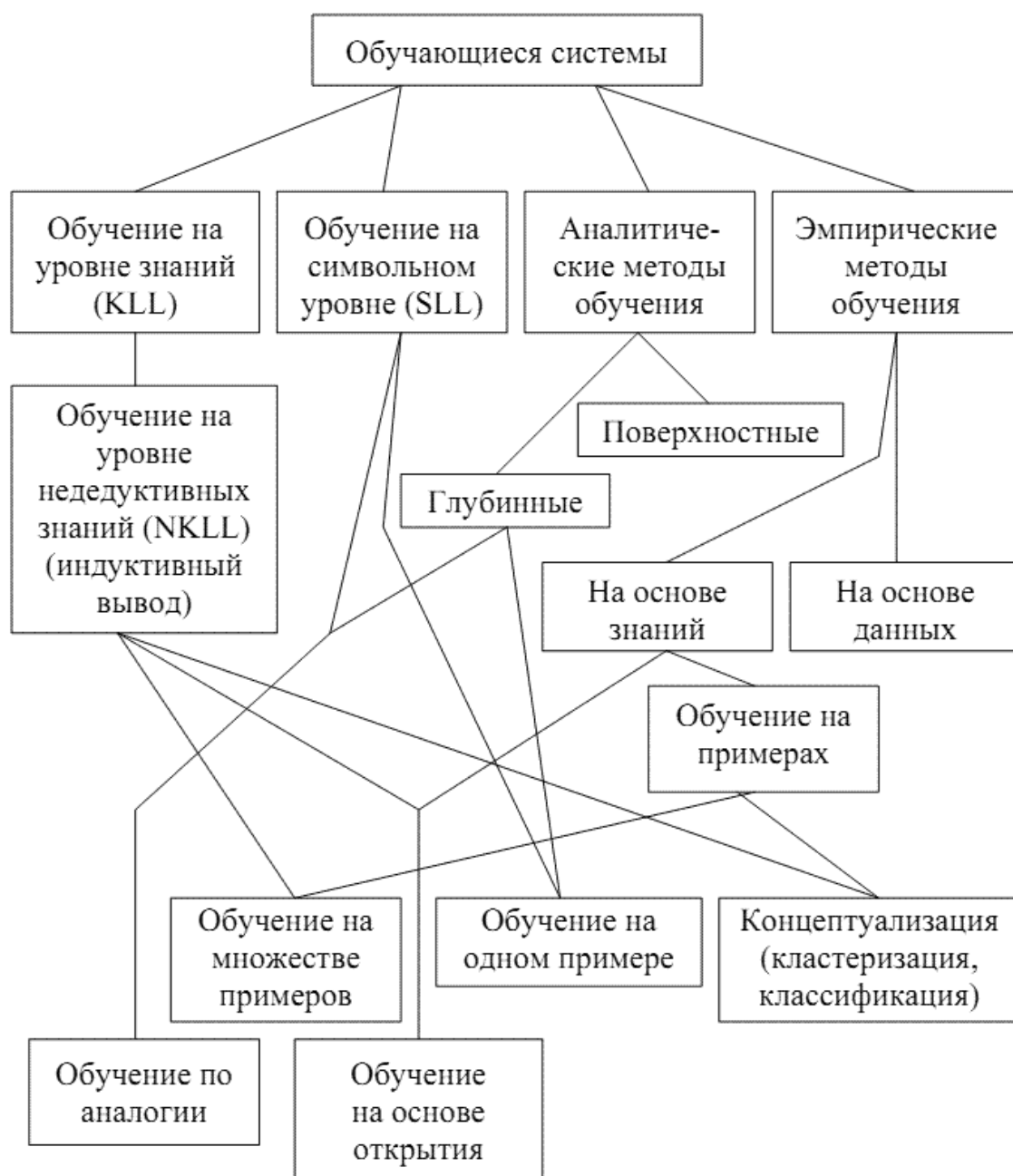


Рисунок 8.1 - Классификация обучающихся систем

Наиболее известными методами приобретения знаний являются [3]:

- ДСМ-метод (обычно относится к индуктивным методам),
- нейронные сети (в них реализовано в наиболее явном виде обучение на примерах).

В ДСМ-методе используется представление знаний об экспериментах (наблюдениях), подтверждающих причинно-следственные связи между факторами, в виде матрицы гипотез [3]:

$$M^+ = \begin{matrix} & a \backslash b & b_1 \dots & b_j & b_m \\ a_i & q_{i1} & & q_{ij} & q_{im} \\ \dots & & & & \\ a_i & q_{i1} & & q_{ij} & q_{im} \\ \dots & & & & \\ a_n & q_{n1} & & q_{nj} & q_{nm} \end{matrix}$$

где a_i – факторы-причины, b_j – факторы-следствия, q_{ij} – оценки истинности (силы) причинно-следственной связи между соответствующими факторами.

Оценка истинности определяется в процессе обучения (экспериментов или наблюдений) как k^+/k , k – общее количество экспериментов (примеров), а k^+ – количество экспериментов, подтверждающих причинно-следственную связь (положительных примеров).

Таким образом, после обучения мы имеем матрицу со значениями q_{ij} в пределах интервала $(0, 1)$. Если значение $q_{ij} \approx 1$, это означает, что между факторами a_i и b_j есть причинно-следственная связь и ее можно записать в виде правила. Иногда в ДСМ-методе используется и матрица отрицательных примеров M^- .

8.3 Методы извлечения знаний из данных

Методы извлечения знаний состоят из:

- текстологических методов,
- методов автоматической обработки текстов.

Текстологические методы предназначены для получения инженером по знаниям знаний из материализованных источников (монографии, учебники, статьи методики, инструкции и другие носители профессиональных знаний). Эти методы основываются не только на выявлении и понимании смысла текста, но и на выделении базовых понятий и отношений, т. е. формировании семантической (понятийной) структуры.

В инженерии знаний разработана методика анализа текстов в целях извлечения и структурирования знаний.

Методика анализа текстов в целях извлечения и структурирования знаний предусматривает:

- анализ микроструктуры текста,
- вычленение ключевых слов (компрессия или сжатие текста),
- формирование поля знаний на базе одного из языков представления знаний.

Сжатие текста служит методологической основой для использования текстологических процедур извлечения знаний. Текстологические методы самые трудоемкие, они применяются, как правило, на начальном этапе создания СИИ.

Существенное развитие получили методы извлечения знаний при использовании современных информационных технологий, в частности гипертекстовой технологии.

Гипертекст — это организация нелинейной последовательности записи и чтения информации, объединенной на основе ассоциативной связи. Синтез этой концепции и полиморфизма приводит к новой концепции, в рамках которой между информацией, представленной в различной форме (текстовой, графической и других), организуются ассоциативные связи.

Эти новые концепции работы со знаниями создают предпосылки для решения проблемы эффективности процесса приобретения знаний. Усилия исследователей в области инженерии знаний направлены на создание формальных методов извлечения знаний. К их числу можно отнести метод автоматической обработки текстов на основе статистической обработки семантических единиц.

Метод автоматической обработки текстов на основе статистической обработки семантических единиц. При использовании данного метода семантические единицы получаются путем статистической обработки текстов, в основе которой лежат универсальные механизмы определения частотных характеристик терминов.

Задача извлечения знаний решается в два этапа:

1. сначала формируется терминологическая сеть (поле знаний),
2. определяется ассоциативная близость терминов на основе статистически определенной меры ассоциации.

Достоинство рассмотренного метода состоит в автоматическом выявлении значимых слов и связей с учетом статистической информации о гипертексте в целом.

Указанные новые подходы к автоматизации извлечения знаний пока находятся на стадии исследований и не нашли применения в практике создания СИИ. Однако результаты исследований позволяют надеяться на создание эффективных методов и систем искусственного интеллекта, позволяющих снизить трудозатраты при извлечении знаний на начальном этапе синтеза баз знаний СИИ.

8.4 Методы получения экспертных знаний

К методам получения экспертных знаний относятся коммуникативные методы (пассивные и активные), основанные на прямом диалоге экспертов и инженеров по знаниям как без использования СИИ, так и с применением СИИ (технологии окон, меню).

Коммуникативные методы получения знаний рассматриваются как разновидности интервьюирования.

Основными особенностями коммуникативных методов является следующие:

- *Не имеют формального определения и носят качественный характер.*
- *Полученные с их помощью знания несут отпечаток самонаблюдений эксперта и субъективную интерпретацию инженера по знаниям.*
- *Требуют словесного выражения экспертом своих знаний, что является непростой задачей. Неточность и неадекватность словесных описаний мыслительных процессов и применяемых эвристических приемов, используемых при решении задач, ведут к серьезным последствиям.*
- *Сложность выражения процедурных знаний при их словесном описании.*
- *Крайняя сложность явного описания знаний, которые являются результатом компиляции и автоматизма процессов мышления, а также интуиции эксперта. В психологии доказано, что интуиция на самом деле является способностью распознавать образы. Однако словесное описание способности к распознаванию образов дать крайне трудно.*
- *Трудоемкость организации и неэффективность взаимодействия инженера по знаниям и эксперта. На них приходится большие интеллектуальные нагрузки, связанные с вербализацией знаний, управлением процессом коммуникации и необходимостью освоения анализа и документирования больших объемов новых знаний.*

Коммуникативные методы получения знаний отличаются своей низкой эффективностью. Так, при непосредственном взаимодействии инженера по знаниям и эксперта теряется до 76% информации. Один из путей совершенствования процесса приобретения знаний состоит в разработке методов, позволяющих передать часть функций, выполняемых инженером по знаниям, самому эксперту или СИИ.

8.5 Методы формирования знаний

Трудности извлечения знаний из текстовых источников и получения их от экспертов стимулировали развитие методов формирования знаний, известных, как методы машинного обучения.

Для развитых СИИ способность обучаться, т. е. самостоятельно формировать новые знания на основе текущих знаний, собственного опыта решения прикладных задач, является их существенной характеристикой. Методы формирования знаний лежат в основе автоматических систем

приобретения знаний. Автоматические системы формирования знаний более предпочтительны, так как при этом уменьшается вероятность ошибок в приобретаемых знаниях и снижается время их приобретения.

Главный вопрос, на который должны ответить методы формирования знаний, состоит в следующем: как от частного (примера) перейти к общему (обобщениям)? Базисом всех методов формирования знаний является индукция, которая лежит в основе получения общих выводов из совокупности частных утверждений.

В основу главы 8 положен материал учебных пособий [2, 3], справочного пособия [5].

9. НЕЧЕТКИЙ ВЫВОД ЗНАНИЙ

9.1 Основные положения нечеткого вывода знаний

Данные и знания, с которыми приходится иметь дело в ИС, редко бывают абсолютно точными и достоверными. Присущая знаниям неопределенность может иметь разнообразный характер, и для ее описания используется широкий спектр формализмов. Рассмотрим один из типов неопределенности в данных и знаниях — их неточность.

Будем называть **высказывание неточным**, если его истинность (или ложность) не может быть установлена с определенностью.

Основопологающим понятием при построении моделей неточного вывода является понятие вероятности, поэтому все описываемые далее методы связаны с вероятностной концепцией.

Модель оперирования с неточными данными и знаниями включает две составляющие:

- язык представления неточности;
- механизм вывода на неточных знаниях.

Для построения языка необходимо выбрать форму представления неточности (например, скаляр, интервал, распределение, лингвистическое выражение, множество) и предусмотреть возможность приписывания меры неточности всем высказываниям.

Механизмы оперирования с неточными высказываниями можно разделить на два типа.

1. **Механизмы, носящие «присоединенный» характер: пересчет мер неточности как бы сопровождает процесс вывода, ведущийся на точных высказываниях.** Для разработки присоединенной модели неточного вывода в основанной на правилах вывода системе необходимо задать функции пересчета, позволяющие вычислять:

- а) меру неточности x антецедента правила (его левой части) по мерам неточности x_i составляющих его высказываний:
 $x = f(x_1, \dots, x_n)$;
- б) меру неточности y консеквента правила (его правой части) по мерам неточности правила (r) и посылки правила (x) $y = h(r, x)$;
- в) объединенную меру неточности высказывания A по мерам, полученным из правил, консеквентом которых является A :
 $y^0 = g(y_1, \dots, y_m)$.

Для возможности пересчета значений неопределенности при дедуктивном выводе достаточно функций $f(\bullet)$ и $h(\bullet)$. Введение меры неточности позволит привнести в процесс вывода нечто принципиально новое — возможность объединения силы нескольких свидетельств, подтверждающих или опровергающих одну и ту же гипотезу. Другими словами, при использовании мер неточности целесообразно выводить одно и то же утверждение различными путями (с последующим объединением значений неточности), что совершенно бессмысленно в традиционной дедуктивной логике. Для *объединения свидетельств* требуется функция пересчета $g(\bullet)$, занимающая центральное место в пересчете. Заметим, что, несмотря на «присоединенность» механизмов вывода этого типа, их реализация в базах знаний оказывает влияние на общую стратегию вывода: с одной стороны, необходимо выводить гипотезу всеми возможными путями для того, чтобы учесть все релевантные этой гипотезе свидетельства, с другой — предотвратить многократное влияние силы одних и тех же свидетельств.

2. ***Механизмы оперирования с неточными высказываниями со схемами вывода, специально ориентированными на используемый язык представления неточности.*** Как правило, каждому шагу вывода соответствует пересчет мер неточности, обусловленный соотношением на множестве высказываний (соотношением может быть элементарная логическая связь, например $A=B\&C$, безотносительно к тому, является ли это отношение фрагментом какого-либо правила). Таким образом, механизмы второго типа применимы не только к знаниям, выраженным в форме правил. Вместе с тем для них, как и для механизмов «присоединенного» типа, одной из главных является проблема объединения свидетельств.

9.2 Типы неточного вывода

Вероятностные методы принятия решений основанные на байесовском подходе. В такой системе в качестве весов, приписанных высказываниям, выступают вероятности соответствующих событий. Множества событий организованы в сети вывода, по которым производится пересчет априорных вероятностей высказываний в апостериорные. Правила в системе имеют вид «если E , то H », причем в сети вывода для каждой пары высказываний $\{E, H\}$ эти правила присутствуют или отсутствуют одновременно.

Знания людей о проблемной области (проблеме) можно представить в виде совместного распределения вероятностей $P(x_1, \dots, x_n)$ на множестве препозиционных переменных (т. е. переменных, значениями которых являются высказывания) $x_1, \dots, x_n$. Интерес представляет вычисление условных вероятностей $P(H|E)$, где H (гипотеза) и E (свидетельство) —

некоторые композиционные высказывания, составленные из элементарных (т. е. типа {значение x_i есть V_j^i }).

Все условные вероятности такого типа можно получить из совместного распределения вероятностей $P(x_1, \dots, x_n)$.

Пусть фиксирован некоторый произвольный порядок d переменных $x_1, \dots, x_n$, каждой из которых соответствует вершина в графе. Тогда

$$P(x_1, \dots, x_n) = P(x_n | (x_{n-1}, \dots, x_1)) \dots P(x_2 | x_1) P(x_1).$$

Пусть $S_i \subseteq \{x_1, \dots, x_{i-1}\}$ — минимальное подмножество вершин множества $\{x_1, \dots, x_{i-1}\}$, удовлетворяющее условию

$$P(x_i | S_i) = P(x_i | x_{i-1}, \dots, x_1).$$

Можно показать, что такое подмножество единственно. Проведем от вершин множества S_i направленные дуги к вершине x_i . Осуществив эту операцию для всех вершин x_i , получим ориентированный ациклический граф, в котором отражены многие соотношения независимости, неявно содержащиеся в $P(x_1, \dots, x_n)$.

Наоборот, порядок d переменных и ориентированный ациклический граф с совокупностью условных распределений вероятностей однозначно определяют распределение $P(x_1, \dots, x_n)$. Такой «локальный» способ определения $P(x_1, \dots, x_n)$ соответствует заданию системы вероятностных суждений, которыми наиболее естественно оперируют эксперты. Графы (а точнее, гиперграфы), построенные описанным способом, носят название *сетей доверия* или *байесовских сетей*.

Вероятностные методы принятия решений основанные на теории нечеткой логики. Вид функции пересчета $f(\bullet)$ постулируется, т. е. для конъюнкции и дизъюнкции используются формулы нечеткой логики:

$$P(A \vee B) = \max(P(A), P(B)),$$

$$P(A \& B) = \min(P(A), P(B)).$$

В основу главы 9 положен материал учебного пособия [2], справочного пособия [5].

10. ИЗВЛЕЧЕНИЕ ЗНАНИЙ ИЗ ДАННЫХ МЕТОДАМИ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА ДАННЫХ

10.1 Особенности систем интеллектуального анализа данных

В связи с высокой трудоемкостью извлечения знаний по обычной технологии инженерии знаний, в течение последних нескольких лет интенсивно разрабатывались методы автоматического извлечения знаний из накопленных фактов. В основе этих методов лежат известные из логики методы индуктивного вывода и ряд методов распознавания образов. В западной литературе эти методы получили название *«раскопка данных и открытие знаний»*. В отечественной литературе прижился термин «Интеллектуальный анализ данных» (ИАД).

Классической основой извлечения знаний из накопленных данных является математическая статистика. Методы математической статистики оказались полезными, главным образом, для проверки заранее сформулированных гипотез и для «грубого» разведочного анализа, составляющего основу оперативной аналитической обработки данных.

Интеллектуальный анализ данных — это процесс поддержки принятия решений, основанный на поиске в данных скрытых закономерностей, то есть извлечения информации, которая может быть охарактеризована как знания.

Интеллектуальный анализ данных является кратким обозначением довольно широкого спектра процедур автоматического анализа данных высокоинтеллектуальными технологиями. Эти методы позволяют извлекать из «сырых» данных ранее неизвестные зависимости между параметрами объектов и закономерности поведения классов объектов. Подобные программные продукты позволяют как бы «осмыслить» данные, оценивая их как с количественной, так и с качественной точки зрения.

В общем случае процесс ИАД состоит из трех стадий:

- 1. выявление закономерностей (свободный поиск);*
- 2. использование выявленных закономерностей для предсказания неизвестных значений (прогностическое моделирование);*
- 3. анализ исключений, предназначенный для выявления и толкования аномалий в найденных закономерностях.*

Иногда в явном виде выделяют промежуточную стадию проверки достоверности найденных закономерностей между их нахождением и использованием (стадия валидации), рис. 10.1.



Рисунок 10.1 – Стадии интеллектуального анализа данных

Все методы ИАД подразделяются на две большие группы по принципу работы с исходными обучающими данными:

1. *Методы рассуждений на основе анализа прецедентов* - исходные данные могут храниться в явном детализированном виде и могут непосредственно использоваться для прогностического моделирования и/или анализа исключений.

Главной проблемой этой группы методов является затруднительность их использования на больших объемах данных, хотя именно при анализе информации из больших хранилищ данных методы ИАД приносят наибольшую пользу.

2. *Методы, в которых информация вначале извлекается из первичных данных и преобразуется в некоторые формальные конструкции (их вид зависит от конкретного метода).* Согласно предыдущей классификации, этот этап выполняется на стадии свободного поиска, которая у методов первой группы в принципе отсутствует. Таким образом, для прогностического моделирования и анализа исключений используются результаты этой стадии, которые гораздо более компактны, чем сами массивы исходных данных. При этом полученные конструкции могут быть либо «прозрачными» (интерпретируемыми), либо «черными ящиками» (не трактуемыми).

Различают следующие подходы к отбору параметров для поиска функциональных зависимостей и вида (или выбора) класса самих зависимостей:

1. *Пользователь сам выдвигает гипотезы относительно зависимостей между данными.* Фактически традиционные технологии анализа развивали именно этот подход. Действительно,

гипотеза приводила к построению отчета, анализу отчета и выдвижению новой гипотезы и т. д. Это справедливо в том случае, когда процесс поиска полностью контролируется человеком. Во многих системах ИАД в этом процессе автоматизирована проверка достоверности гипотез, что позволяет оценить вероятность тех или иных зависимостей в базе данных.

2. Зависимости между данными ищутся автоматически.

Процесс поиска зависимостей распадается на три этапа:

1. ***Обнаружение зависимостей (discovery)*** состоит в просмотре базы данных с целью автоматического выявления зависимостей. Проблема здесь заключается в отборе действительно важных зависимостей из огромного числа существующих в БД.
2. ***Прогнозирование (predictive modelling)*** предполагает, что пользователь может предъявить системе записи с незаполненными полями и запросить недостающие значения. Система сама анализирует содержимое базы и делает правдоподобное предсказание относительно этих значений.
3. ***Анализ аномалий (forensic analysis)*** — это процесс поиска подозрительных данных, сильно отклоняющихся от устойчивых зависимостей.

При выборе системы ИАД следует уделить внимание следующим моментам:

1. Система ИАД должна предсказывать значения целевой переменной и решать задачи классификации состояний объекта с тем, чтобы подбирать наилучшие модели для каждого класса состояний.
2. Система должна автоматически выполнять тесты, определяющие статистическую значимость развиваемой модели. Произвольный комплекс даже случайно сгенерированных данных может быть объяснен, если включить достаточно большое число свободных параметров в модель. Однако такая модель не имеет какой бы то ни было предсказательной силы. Эта проблема называется «подгонкой» и часто является недостатком систем, основанных на нейронных сетях.
3. Полученная модель должна быть легко интерпретируема. Если вы не можете понять, какие знания модель содержит, как точно целевая переменная зависит от независимых переменных, вы практически не способны контролировать результаты. Вы не сможете, опираясь на личный опыт, увидеть в найденной модели возможные противоречия. Нейросети особенно опасны в этом смысле, так как построенная модель представляет собой «черный ящик». Куда более привлекательное представление знаний

обеспечивает технология эволюционного программирования и приобретения знаний в символическом виде. В данном случае обнаруженные зависимости представляются в виде формулы, связывающей целевую и независимые переменные. Такая формула может содержать как математические зависимости, так и логические конструкции.

4. Система должна находить правила разнообразного вида. Перед началом исследования вы не можете сказать точно, какой именно тип отношений между переменными скрыт в ваших данных. Поэтому нужно быть готовым к переборке разнообразных видов зависимостей, чтобы не пропустить оптимальный вариант.
5. Контроль за процессом обработки данных усилится, если воспользоваться мультистратегической ИАД-системой. Такая система обладает целым набором взаимно дополняемых инструментов, которые позволяют пользователю анализировать данные, исходя из различных условий. Использование сочетания различных методов позволяет значительно повысить значимость получаемых результатов и общие характеристики системы.
6. Важное значение имеет также время обработки данных. Однако этот параметр очень сложно определить точно, поскольку он сильно зависит от характеристик исследуемых данных.

10.2 Типы закономерностей, выявляемых методами интеллектуального анализа данных

Выделяют пять стандартных типов закономерностей, которые позволяют выявлять методы ИАД:

1. ассоциация,
2. последовательность,
3. классификация,
4. кластеризация,
5. прогнозирование.

Ассоциация имеет место в том случае, если несколько событий связаны друг с другом. Например, исследование, проведенное в супермаркете, может показать, что 65% купивших кукурузные чипсы берут также и «кока-колу», а при наличии скидки за такой комплект «колу» приобретают в 85% случаев. Располагая сведениями о подобной ассоциации, менеджерам легко оценить, насколько действенна предоставляемая скидка.

Последовательность. Если существует цепочка связанных во времени событий, то говорят о последовательности. Так, после покупки дома в 45%

случаев в течение месяца приобретается и новая кухонная плита, а в пределах двух недель 60% новоселов обзаводятся холодильником.

Классификация. С помощью классификации выявляются признаки, характеризующие группу, к которой принадлежит тот или иной объект. Это делается посредством анализа уже классифицированных объектов и формулирования некоторого набора правил. Там, где кластеризация помогает определить классы, классификация приписывает новые записи к существующим классам. Например, банк может изучить базу данных своих заемщиков, чтобы разделить их на две группы: те, которые могут разориться и те, которые чувствуют себя хорошо.

Кластеризация отличается от классификации тем, что сами группы заранее не заданы. С помощью кластеризации средства ИАД самостоятельно выделяют различные однородные группы данных. Кластеризация часто используется для того, чтобы помочь маркетологу выявить различные группы в их базе данных клиентов. Компании используют эти методы для того, чтобы разработать программы целевого маркетинга.

Например, компания может использовать кластеризацию для того, чтобы идентифицировать:

- покупателей, которые вероятно приобретут домой видеоплееры;
- магазины, которые вероятно продают спутниковые тарелки;
- покупателей с различными способами использования мобильных телефонов.

Прогнозирование. Основой для всевозможных систем прогнозирования служит историческая информация, хранящаяся в БД в виде временных рядов. Если удастся построить, найти шаблоны, адекватно отражающие динамику поведения целевых показателей, есть вероятность, что с их помощью можно прогнозировать поведение системы в будущем.

Предположим, что мы выбрали следующие характеристики клиента для нашей базы данных:

- Zip код,
- возраст,
- пол,
- номер телефона,
- дата первого обращения в магазин,
- общий объем покупок,
- объем покупок за последний год,
- дата последней покупки,
- стоимость последней покупки,
- число пересланных по почте рекламных проспектов клиенту за последний год,

- ежегодный заработок,
- является ли клиент арендатором или владельцем офиса.

Процесс ИАД начинается с формулировки интересующего пользователя вопроса. Например, нас может интересовать, каковы характерные особенности клиента, совершившего максимальное количество покупок за последний год, и наоборот. Нам необходимо изучить записи о клиентах, контакты с которыми продолжаются более года и которые получили по почте за последний год одно и то же количество наших рекламных проспектов. Сможем ли мы обнаружить явную зависимость между независимыми переменными и целевой переменной с желаемой точностью? Если такая модель найдена — мы можем использовать ее для определения, кому из клиентов имеет смысл продолжать рассылать рекламу, а кому — нет, и проводить следующую рекламную кампанию, опираясь, в первую очередь, именно на них. Это позволит нам сэкономить большое количество денег без ущерба для уровня продаж.

Затем мы можем определить эффективность нашей рекламной кампании. Например, попробуем узнать, какой отклик находит наша рекламная стратегия у клиентов. Для этого мы, как обычно, выберем только тех клиентов, контакты с которыми поддерживаются более года. Однако теперь мы изучим тех клиентов, которые попадают в одну и ту же группу по объему покупок согласно предыдущей модели, но получили различное количество почтовой рекламы за последний год, и постараемся построить модель зависимости уровня продаж от количества рассылок за последний год. Если уровень откликов возрастает с числом посланных клиенту рекламных проспектов — все прекрасно. Однако если количество рекламных проспектов не входит в формулу, предсказывающую объем продаж, или, что еще хуже, снижает их уровень — тогда что-то не в порядке с нашей почтовой рекламой. Нам необходимы срочные меры по изменению рекламной стратегии.

10.3 Этапы функционирования типовой системы интеллектуального анализа данных

Система ИАД позволяет представить обнаруженные закономерности в символической форме — как математические формулы, таблицы предсказаний, структурные законы и алгоритмы, т.е. в форме, которая естественна и удобна для понимания, причем система не только «открывает» знания, но и объясняет их.

Функционирование системы ИАД можно разбить на три этапа.

1. *Вначале работает модуль предварительной обработки данных, который существенно сужает пространство поиска, отбрасывая малозначимые точки и оценивая наилучшую точность для порождаемых многомерных зависимостей (в специальной*

терминологии это именуется препроцессором данных по обнаружению функционально-связанных кластеров в массивах данных; фильтрация шума и случайных выбросов).

Если удастся установить взаимозависимость, даже слабую, об этом будет сделано сообщение с указанием статистических характеристик, с визуализацией значимой области и выпавших точек, а близость оценивается классической суммой наименьших квадратов отклонений.

2. *Построение линейной многомерной зависимости специально выделяется в отдельную задачу.* Какая бы зависимость в общем случае ни обнаружилась, всегда полезно иметь под рукой наиболее простую.
3. *На третьем этапе программа автоматически порождает и отбрасывает различные гипотезы о взаимозависимости данных в форме функциональных процедур.* При этом она не нуждается ни в каких предположениях о форме гипотез: степенные, показательные, тригонометрические и их комбинации, составные «если ..., то», а также алгоритмы. Если предложенных средств окажется недостаточно, класс может быть расширен.

10.4 Пример функционирования системы интеллектуального анализа данных

Рассмотрим этапы подготовки данных к анализу. Попутно подчеркнем те особенности системы ИАД, которые важны для успешного выполнения этой задачи.

Возьмем в качестве примера базу данных, содержащую записи о клиентах, значения целевой переменной для которых — «объем продаж за последний год» — известны.

1. Сначала рассмотрим записи только о тех клиентах, контакт с которыми поддерживается уже более года и которые получили две наши почтовые рекламы за последний год.

Обычно для работы достаточно взять только часть данных — от 2-3 до 50 тысяч записей. Этого вполне достаточно для построения значимой модели. Поэтому система ИАД, должна быть способна обработать такое число записей. Помимо этого она должна обладать механизмами свободного манипулирования данными, поскольку может понадобиться разбить данные на некоторые подгруппы и оставить часть данных для тестирования, объединить данные или привести их во взаимодействие. Например, полагая, что клиенты различных возрастных категорий будут обладать различными покупательскими особенностями, может понадобиться разделить их на группы и изучать каждую из них отдельно.

Заметим, что практически всегда нужны сочетания атрибутов числовых, логических и категориальных типов. Например, в данных «пол» и «арендатор/владелец» — логические переменные, тогда как «ZIP код» — категориальный тип. Поэтому система ИАД, должна поддерживать все типы атрибутов. К тому же большинство задач по маркетингу содержат даты, и поэтому временной формат должен также ею поддерживаться.

Пусть необходимо подсчитать количество дней, прошедшее с некоторого события. Например, вместо даты первого контакта, рассмотрим, сколько времени прошло с тех пор. Для этого мы должны вычислить новую переменную, вычитая соответствующую дату из сегодняшней.

2. Создаем новую базу данных, содержащую только те переменные, которые мы собираемся включить в исследование. Например, переменные «дата» заменяем на переменные, указывающие временной промежуток между некоторыми событиями. Исключаем переменные, описывающие недавние торговые сделки, поскольку они не могут влиять на общий объем продаж за предыдущий год, и изменяем тип переменной «ZIP код» с числового на категориальный.

Например, мы считаем, что 10% ваших клиентов составляют одинокие мужчины в возрасте 30-35 лет с годовым доходом более 40 000\$. Применим это правило к исследуемому набору данных в качестве новой переменной, описывающей наши предположения. Включим эту новую независимую переменную при запуске одного из методов исследования. Таким образом, мы посоветуем системе, какое правило применить первым. Если наше предположение верно — соответствующая переменная будет строго включена в конечную формулу. Если же система не включит созданную переменную в конечный результат — нам лучше пересмотреть свою точку зрения.

3. Автоматическое построение эмпирической модели, которая описывает зависимость целевой переменной от независимых. Если полученная модель не удовлетворяет точности, надежности и легкости понимания полученных зависимостей, которые предсказывают будущее значение целевой переменной, то такая система, фактически бесполезна.

Теперь запускаем метод исследования «Найти закон», выбрав в качестве целевой переменной общий объем продаж за последний год и установив желаемую ошибку — 10%. Система определяет ясный вид отношения, связывающего целевую переменную с независимыми параметрами, характеризующими клиента.

В качестве конечного продукта система генерирует отчет, который содержит одно текстовое и два графических окна. В текстовом окне в ясном виде показывается лучшая из найденных моделей, которая объясняет данные наиболее надежно, точно и значимо. Точность характеризуется стандартной ошибкой, с которой построенная модель будет предсказывать значения

целевой переменной. Значимость определяет тот факт, что модель объясняет данные неслучайно. Этого довольно легко достичь, поскольку модель представлена вам в форме ясного математического отношения, которое включает алгебраические и логические конструкции.

5. Последний шаг в цикле ИАД — это развитие стратегии, основанной на полученной модели. Он не относится к ИАД непосредственно. Если сформулирован вопрос, и построенная модель корректна, то она позволяет ответить на него. Теперь нам необходимо использовать полученные знания для принятия решения о поведении в будущем, о том, какие дополнительные и доступные методы нужно задействовать.

Теперь подсчитаем, что именно дает нам исследование данных? Основное приобретение — это получаемая нами возможность принимать разумно обоснованные решения, основанные на модели, автоматически выведенной на основании существующих данных. До процесса исследования эти знания были надежно укрыты за бесчисленным количеством «сырых», необработанных данных.

В основу главы 10 положен материал учебных пособий [1, 2].

ЧАСТЬ 4. ПРИНЦИПЫ ПОСТРОЕНИЯ И ФУНКЦИОНИРОВАНИЯ ПРИКЛАДНЫХ СИСТЕМ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

11. ЭКСПЕРТНЫЕ СИСТЕМЫ

11.1 Экспертные системы: базовые понятия

В современном обществе при решении задач управления сложными многопараметрическими и сильно связанными системами, объектами, производственными и технологическими процессами приходится сталкиваться с решением не формализуемых либо трудно формализуемых задач. Такие задачи часто возникают в следующих областях: авиация, космос и оборона, нефтеперерабатывающая промышленность и транспортировка нефтепродуктов, химия, энергетика, металлургия, целлюлозно-бумажная промышленность, телекоммуникации и связь, пищевая промышленность, машиностроение, производство цемента, бетона и т.п. транспорт, медицина и фармацевтическое производство, административное управление, прогнозирование и мониторинг. Наиболее значительными достижениями в этой области стало создание систем, которые ставят диагноз заболевания, предсказывают месторождения полезных ископаемых, помогают в проектировании электронных устройств, машин и механизмов, решают задачи управления реакторами и другие задачи.

Экспертные системы – это прикладные системы ИИ, в которых база знаний представляет собой формализованные эмпирические знания высококвалифицированных специалистов (экспертов) в какой-либо узкой предметной области. Экспертные системы предназначены для замены при решении задач экспертов в силу их недостаточного количества, недостаточной оперативности в решении задачи или в опасных (вредных) для них условиях.

Экспертная система (ЭС) - программа, которая использует знания специалистов (экспертов) о некоторой конкретной узко специализированной предметной области и в пределах этой области способна принимать решения на уровне эксперта-профессионала [1].

Осознание полезности систем, которые могут копировать дорогостоящие или редко встречающиеся человеческие знания, привело к широкому внедрению и расцвету этой технологии в 80-е, 90-е годы прошлого века.

Основу успеха ЭС составили два важных свойства [1]:

- 1. в ЭС знания отделены от данных, и мощность экспертной системы обусловлена в первую очередь мощностью базы знаний и только во вторую очередь используемыми методами решения задач;*

2. *решаемые ЭС задачи являются неформализованными или слабоформализованными и используют эвристические, экспериментальные, субъективные знания экспертов в определенной предметной области.*

Основными категориями решаемых ЭС задач являются [1]:

- *диагностика,*
- *управление (в том числе технологическими процессами),*
- *интерпретация,*
- *прогнозирование,*
- *проектирование,*
- *отладка и ремонт,*
- *планирование,*
- *наблюдение (мониторинг),*
- *обучение.*

С экспертными системами связаны следующие распространенные заблуждения [1]:

1. ЭС будут делать не более (а скорее даже менее) того, чем может эксперт, создавший данную систему. Для опровержения данного постулата можно построить самообучающуюся ЭС в области, в которой вообще нет экспертов, либо объединить в одной ЭС знания нескольких экспертов, и получить в результате систему, которая может то, чего ни один из ее создателей не может.
2. ЭС никогда не заменит человека-эксперта. Уже заменяет, иначе зачем бы их создавали?

Примерами широко известных и эффективно используемых (или использованных в свое время) экспертных систем являются [1]:

- DENDRAL – ЭС для распознавания структуры сложных органических молекул по результатам их спектрального анализа (считается первой в мире экспертной системой);
- MOLGEN – ЭС для выработки гипотез о структуре ДНК на основе экспериментов с ферментами;
- XCON – ЭС для конфигурирования (проектирования) вычислительных комплексов VAX 11 в корпорации DEC в соответствии с заказом покупателя;
- MYCIN – ЭС диагностики кишечных заболеваний;
- PUFF – ЭС диагностики легочных заболеваний;
- MACSYMA – ЭС для символьных преобразований алгебраических выражений;

- YES/MVS – ЭС для управления многозадачной операционной системой MVS больших ЭВМ корпорации IBM;
- DART – ЭС для диагностики больших НМД корпорации IBM;
- PROSPECTOR – ЭС для консультаций при поиске залежей полезных ископаемых;
- AIRPLANE – экспертная система для помощи летчику при посадке на авианосец;
- ЭСПЛАН – ЭС для планирования производства на Бакинском нефтеперерабатывающем заводе;
- МОДИС – ЭС диагностики различных форм гипертонии;
- МИДАС – ЭС для идентификации и устранения аварийных ситуаций в энергосистемах.

11.2 Классификация экспертных систем

1. По назначению ЭС делятся на:

1.1. ЭС общего назначения.

1.2. Специализированные ЭС:

1.3. проблемно-ориентированные для задач диагностики, проектирования, прогнозирования

1.4. предметно-ориентированные для специфических задач, например, контроля ситуаций на атомных электростанциях.

2. По степени зависимости от внешней среды выделяют:

2.1. Статические ЭС, не зависящие от внешней среды.

2.2. Динамические, учитывающие динамику внешней среды и предназначенные для решения задач в реальном времени. Время реакции в таких системах может задаваться в миллисекундах, и эти системы реализуются, как правило, на языке C++.

3. По типу использования различают:

3.1. Изолированные ЭС.

3.2. ЭС на входе/выходе других систем.

3.3. Гибридные ЭС или, иначе говоря, ЭС интегрированные с базами данных и другими программными продуктами (приложениями).

4. По сложности решаемых задач различают:

4.1. Простые ЭС - до 1000 простых правил.

4.2. Средние ЭС - от 1000 до 10000 структурированных правил.

4.3. Сложные ЭС - более 10000 структурированных правил.

5. По стадии создания выделяют:

- 5.1. Исследовательский образец ЭС, разработанный за 1-2 месяца с минимальной БЗ.
- 5.2. Демонстрационный образец ЭС, разработанный за 2-4 месяца, например, на языке типа LISP, PROLOG, CLIPS
- 5.3. Промышленный образец ЭС, разработанный за 4-8 месяцев, например, на языке типа CLIPS с полной БЗ.
- 5.4. Коммерческий образец ЭС, разработанный за 1,5-2 года, например, на языке типа C++, Java с полной БЗ.

11.3 Составные части экспертной системы и порядок ее функционирования

Обобщенная структурная схема ЭС приведена на рис. 11.1.

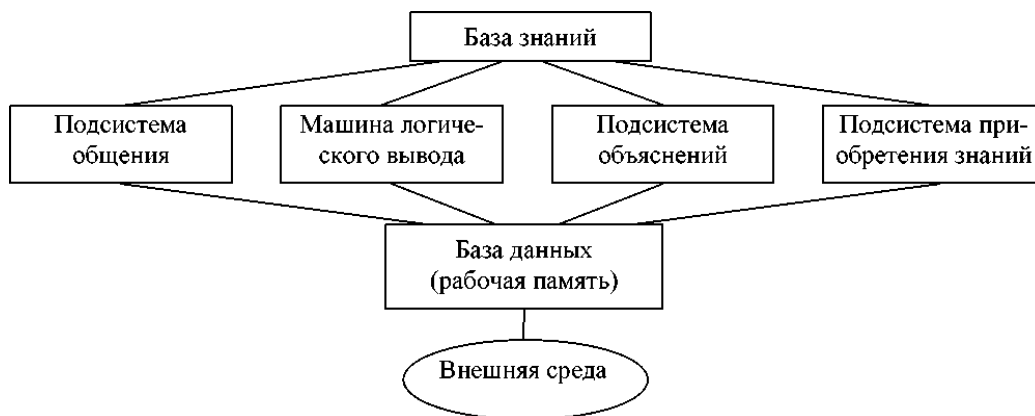


Рисунок 11.1 - Обобщенная структурная схема ЭС

Основными компонентами ЭС являются ниже перечисленные элементы.

Машина логического вывода – механизм рассуждений, оперирующий знаниями и данными с целью получения новых данных из знаний и других данных, имеющих в рабочей памяти. Для этого обычно используется программно реализованный механизм дедуктивного логического вывода (какая-либо его разновидность) или механизм поиска решения в сети фреймов или семантической сети.

Машина логического вывода может реализовывать рассуждения в виде:

- дедуктивного вывода (прямого, обратного, смешанного);
- нечеткого вывода;
- вероятностного вывода;
- унификации (подобно тому, как это реализовано в Прологе);

- поиска решения с разбиением на последовательность подзадач;
- поиска решения с использованием стратегии разбиения пространства поиска с учетом уровней абстрагирования решения или понятий, с ними связанных;
- монотонного или немонотонного рассуждения;
- рассуждений с использованием механизма аргументации;
- ассоциативного поиска с использованием нейронных сетей;
- вывода с использованием механизма лингвистической переменной.

***Подсистема общения** служит для ведения диалога с пользователем, в ходе которого ЭС запрашивает у пользователя необходимые факты для процесса рассуждения, а также, дающая возможность пользователю в какой-то степени контролировать и корректировать ход рассуждений экспертной системы.*

***Подсистема объяснений** необходима для того, чтобы дать возможность пользователю контролировать ход рассуждений экспертной системы. Если нет этой подсистемы, экспертная система выглядит для пользователя как «вещь в себе», решениям которой можно либо верить, либо нет. Нормальный пользователь выбирает последнее, и такая ЭС не имеет перспектив для использования.*

***Подсистема приобретения знаний** служит для корректировки и пополнения базы знаний. В простейшем случае это – интеллектуальный редактор базы знаний, в более сложных экспертных системах – средства для извлечения знаний из баз данных, неструктурированного текста, графической информации и т.д.*

Порядок взаимодействия составных частей ЭС приведен на функциональной схеме ЭС – рис. 11.2.

Основу ЭС составляет **подсистема логического вывода**, которая использует информацию из **базы знаний** (БЗ), генерирует рекомендации по решению искомой задачи. Чаще всего для представления знаний в ЭС используются системы продукций и семантические сети. Допустим, БЗ состоит из фактов и правил (если <посылка> то <заключение>). Если ЭС определяет, что посылка верна, то правило признается подходящим для данной консультации и оно запускается в действие. Запуск правила означает принятие заключения данного правила в качестве составной части процесса консультации.

Обязательными частями любой ЭС являются также **модуль приобретения знаний** и **модуль отображения и объяснения решений**. В большинстве случаев, реальные ЭС в промышленной эксплуатации работают также на основе **баз данных** (БД). Только одновременная работа со знаниями и большими объемами информации из БД позволяет ЭС получить

неординарные результаты, например, поставить сложный диагноз (медицинский или технический), открыть месторождение полезных ископаемых, управлять ядерным реактором в реальном времени.

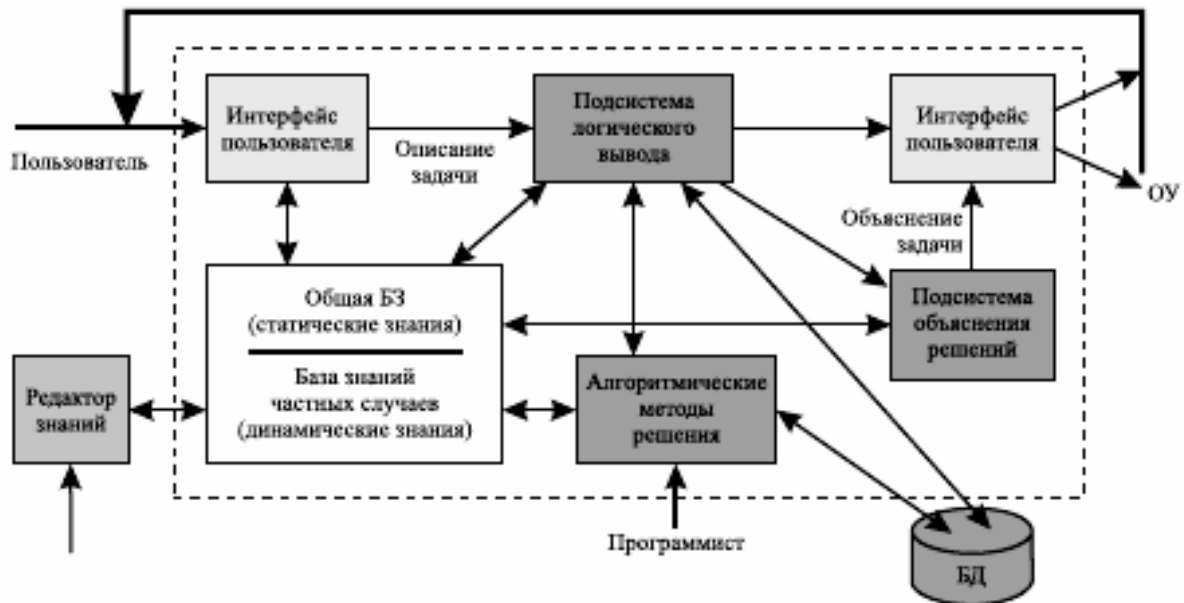


Рисунок 11.2 – Функциональная схема ЭС

11.4 Функционирование базы знаний экспертной системы

Обычно при описании баз знаний (как правило продукционных) экспертных систем правила представляются в наглядном виде, например:

ПРАВИЛО 1:
 ЕСЛИ
 Образование=Высшее И Возраст=Молодой И
 Коммуникабельность=Высокая
 ТО
 Шансы найти работу=Высокие КД=0.9.

При срабатывании этого правила в базу данных интеллектуальной системы (например, экспертной системы) добавляется факт, означающий, что шансы найти работу высоки с достоверностью 0.9 или 90 % (значение коэффициента достоверности КД). Понятия "Образование", "Возраст", "Коммуникабельность" служат для задания условия (в данном случае, конъюнкции), при котором срабатывает правило.

Факты хранятся в базе данных продукционной системы в форме

(Объект, значение, коэффициент достоверности)

или

(Объект, атрибут, значение, коэффициент достоверности).

Кроме правил в продукционных базах знаний могут использоваться метаправила для управления логическим выводом. Пример метаправила для гипотетической базы знаний:

ЕСЛИ
Экономика = развивается
ТО
Увеличить приоритет правила 1

При интерпретации (выполнении) правила в ходе проверки условия система проверяет факты, находящиеся уже в базе данных, и, если соответствующего факта нет, обращается за ним к источнику данных (пользователю, базе данных и т.д.) с вопросом (или запросом).

В БЗ ЭС могут использоваться и другие структуры для хранения фактов, такие как семантические сети или фреймы. В этом случае говорят о комбинации разных методов представления знаний или о гибридных интеллектуальных экспертных системах.

Для представления нечетких знаний факты и правила в продукционных системах снабжаются коэффициентами достоверности (или уверенности), которые могут принимать значения из разных интервалов в разных системах (например, (0, 1), (0, 100), (-1, +1)).

11.4.1 Обратный метод логического дедуктивного вывода

Для решения задач в продукционной интеллектуальной системе существует два основных метода дедуктивного логического вывода: обратный и прямой. Может использоваться и комбинация этих двух методов.

При обратном логическом выводе процесс интерпретации правил начинается с правил, непосредственно приводящих к решению задачи. В них в правой части находятся заключения с фактами, являющимися решением (целевыми фактами). При интерпретации этих правил в процесс решения могут вовлекаться другие правила, результатом выполнения которых являются факты, участвующие в условиях конечных правил и т.д.

Метод обратного логического вывода можно применять тогда, когда необходимо минимизировать количество обращений к источнику данных (например, пользователю), исключив из рассмотрения заведомо ненужные для решения задачи факты.

В самом общем виде алгоритм обратного логического вывода, записанный на псевдокоде в виде функции, выглядит так (условие ограничено конъюнкцией элементарных условий).

функция Доказана_Цель (Цель): boolean;

Поместить Цель в стек целей.

пока стек целей не пуст **цикл**

Выбор цели из стека целей и назначение ее текущей.

Поиск множества правил, в правой части которых находится текущая цель (множества подходящих правил).

Считать, что Цель не доказана.

пока множество подходящих правил не пусто
и Цель не доказана

цикл

Выбор из этого множества одного текущего правила с использованием определенной стратегии. Считать текущим элементарным условием первое.

пока не проверены все элементарные условия правила и не
надо прервать проверку условия

цикл

если в текущем элементарном условии участвует
факт, встречающийся в правой части какого-то
правила

то

если не Доказана_Цель (Этот факт) **то**
Надо прервать проверку условия

конец если иначе

Запросить информацию о факте.

Проверить элементарное условие.

если элементарное условие истинно **то**
Добавить факт в базу данных. Перейти к
следующему элементарному Условию.

иначе

Надо прервать проверку условия.

конец если конец если конец цикла

если условие правила истинно **то**
Выполнить заключение. Исключить Цель из стека
целей. Считать, что Цель доказана.

конец если конец цикла конец цикла конец функции.

Существует много различных стратегий выбора правила из подходящих. Наиболее простой и часто встречающейся стратегией является «первая попавшаяся». При этой стратегии решение задачи зависит от порядка расположения (перебора) правил в базе знаний.

11.4.2 Прямой метод логического дедуктивного вывода

В прямом методе логического вывода интерпретация правил начинается от известных фактов, т.е. сначала выполняются правила, условия которых можно проверить с использованием фактов, уже находящихся в базе данных.

В общем виде алгоритм прямого вывода приведен ниже.

пока Цель не доказана **цикл**

Формирование множества подходящих правил (по их условиям и наличию фактов).

Выбор одного правила из этого множества (с использованием определенной стратегии выбора). Считать текущим элементарным условием первое.

пока не проверены все элементарные условия правила и не надо прервать проверку условия

цикл

если элементарное условие истинно **то**

Перейти к следующему элементарному условию.

иначе

Надо прервать проверку условия.

конец если конец цикла

Выполнить заключение.

если при формировании заключения появился целевой факт **то**

Считать, что Цель доказана.

конец если конец цикла.

Метод прямого логического вывода можно применять тогда, когда факты появляются в базе данных не зависимо от того, какую задачу сейчас требуется решить (какой целевой факт доказать) и в разные моменты времени. В этом случае можно говорить о том, что факты управляют логическим выводом (решением задачи). Кроме того, этот метод целесообразно применять для формирования вторичных признаков (фактов) из первичных для подготовки решения задачи в дальнейшем с применением обратного логического вывода.

В основу главы 11 положен материал учебного пособия [1], работы [6] и учебных курсов [7, 8].

12. ЭТАПЫ ПРОЕКТИРОВАНИЯ ЭКСПЕРТНОЙ СИСТЕМЫ

Методика разработки ЭС включает в себя следующие этапы [1]:

1. Идентификация,
2. Концептуализация,
3. Формализация,
4. Выполнение,
5. Тестирование,
6. Опытная эксплуатация.

Последовательность этапов построения ЭС приведена на рисунке 12.1.

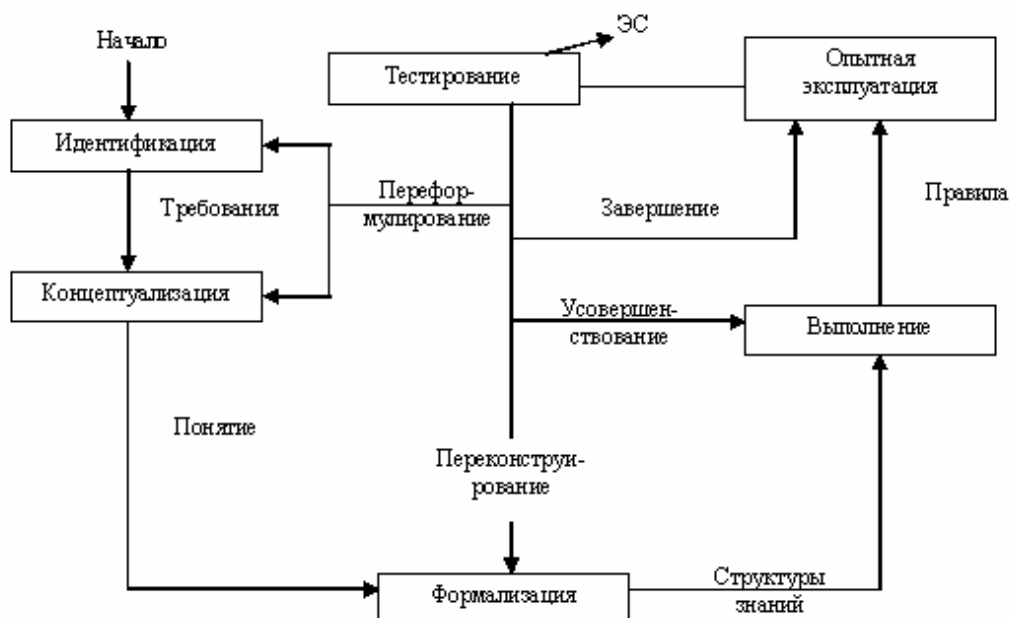


Рисунок 12.1 - Этапы разработки ЭС

12.1 Этап идентификации

Этап идентификации связан, прежде всего, с осмыслением тех задач, которые предстоит решить будущей ЭС, и формированием требований к ней. Результатом данного этапа является ответ на вопрос, что надо сделать и какие ресурсы необходимо задействовать (идентификация задачи, определение участников процесса проектирования и их роли, выявление ресурсов и целей).

Идентификация задачи заключается в составлении неформального (вербального) описания, в котором указываются:

- общие характеристики задачи;
- подзадачи, выделяемые внутри данной задачи;

- *ключевые понятия (объекты), их входные (выходные) данные;*
- *предположительный вид решения,*
- *знания, относящиеся к решаемой задаче.*

При идентификации целей важно отличать цели, ради которых создается ЭС, от задач, которые она должна решать. **Примерами возможных целей являются [1]:**

- *формализация неформальных знаний экспертов;*
- *улучшение качества решений, принимаемых экспертом;*
- *автоматизация рутинных аспектов работы эксперта (пользователя);*
- *тиражирование знаний эксперта.*

12.2 Этап концептуализации

На данном этапе проводится содержательный анализ проблемной области, выявляются используемые понятия и их взаимосвязи, определяются методы решения задач. Этот этап завершается созданием модели предметной области (ПО), включающей основные концепты и отношения. На этапе концептуализации определяются следующие особенности задачи [1]:

- *типы доступных данных;*
- *исходные и выводимые данные,*
- *подзадачи общей задачи;*
- *применяемые стратегии и гипотезы; виды взаимосвязей между объектами ПО, типы используемых отношений (иерархия, причина — следствие, часть — целое и т.п.);*
- *процессы, применяемые в ходе решения;*
- *состав знаний, используемых при решении задачи;*
- *типы ограничений, накладываемых на процессы, которые применены в ходе решения;*
- *состав знаний, используемых для обоснования решений.*

Существует два подхода к процессу построения модели предметной области, которая является целью разработчиков ЭС на этапе концептуализации [1]:

1. **Признаковый или атрибутивный подход** предполагает наличие полученной от экспертов информации в виде троек «объект — атрибут — значение атрибута», а также наличие обучающей информации. Этот подход развивается в рамках направления,

получившего название «формирование знаний» или «машинное обучение» (machine learning).

2. **Структурный (или когнитивный) подход**, осуществляется путем выделения элементов предметной области, их взаимосвязей и семантических отношений.

12.2.1 Атрибутивный подход к построению модели предметной области

Признаковый или атрибутивный подход. Для атрибутивного подхода характерно наличие наиболее полной информации о предметной области: об объектах, их атрибутах и о значениях атрибутов. Кроме того, существенным моментом является использование дополнительной обучающей информации, которая задается группированием объектов в классы по тому или иному содержательному критерию. Тройки объект—атрибут—значение атрибута могут быть получены с помощью так называемого метода реклассификации, который основан на предположении что задача является объектно-ориентированной и объекты задачи хорошо известны эксперту. Идея метода состоит в том, что конструируются правила (комбинации значений атрибутов), позволяющие отличить один объект от другого [1].

На атрибутивном подходе базируются ЭС распознавания образов и автоматического группирования данных.

12.2.2 Структурный (когнитивный) подход к построению модели предметной области

Структурный подход к построению модели предметной области предполагает выделение следующих когнитивных элементов знаний [1]:

1. Понятия.
2. Взаимосвязи.
3. Метапонятия.
4. Семантические отношения.

12.2.2.1 Понятия предметной области

Выделяемые понятия предметной области должны образовывать систему, обладающую следующими свойствами [1]:

- *уникальностью* (отсутствием избыточности);
- *полнотой* (достаточно полным описанием различных процессов, фактов, явлений и т.д. предметной области);

- *достоверностью* (валидностью — соответствием выделенных единиц смысловой информации их реальным наименованиям)
- *непротиворечивостью* (отсутствием омонимии).

Методы построения системы понятий [1]:

- *«Метод локального представления».*
- *«Метод вычисления коэффициента использования».*
- *«Метод формирования перечня понятий».*
- *«Ролевой метод».*
- *Метод «составления списка элементарных действий».*
- *Метод «составление оглавления учебника».*
- *«Текстологический метод».*

12.2.2.2 Взаимосвязи между понятиями предметной области

Группа методов установления взаимосвязей предполагает установление семантической близости между отдельными понятиями. В основе установления взаимосвязей лежит психологический эффект «свободных ассоциаций», а также фундаментальная категория близости объектов или концептов.

Эффект свободных ассоциаций заключается в следующем. Эксперту предъявляется понятие с просьбой назвать как можно быстрее первое пришедшее на ум понятие из сформированной ранее системы понятий. Количество переходов в цепочке может служить мерой «смыслового расстояния» между двумя понятиями.

Выделенные понятия предметной области и установленные между ними взаимосвязи служат основанием для дальнейшего построения системы метапонятий — осмысленных в контексте изучаемой предметной области системы группировок понятий. Для определения этих группировок применяют как неформальные, так и формальные методы.

12.2.2.3 Интерпретация предметной области

Интерпретация, как правило, легче дается эксперту, если группировки получены неформальными методами. В этом случае выделенные классы более понятны эксперту. Причем в некоторых предметных областях совсем не обязательно устанавливать взаимосвязи между понятиями, так как метапонятия, образно говоря, «лежат на поверхности».

12.2.2.4 Установление семантических отношений между понятиями предметной области

Последним этапом построения модели предметной области при концептуальном анализе является установление семантических отношений между выделенными понятиями и метапонятиями. Установить семантические отношения — это значит определить специфику взаимосвязи, полученной в результате применения тех или иных методов. Для этого необходимо каждую зафиксированную взаимосвязь осмыслить и отнести ее к тому или иному типу отношений.

Существует около 200 базовых отношений, например, «часть — целое», «род — вид», «причина — следствие», пространственные, временные и другие отношения. Для каждой предметной области помимо общих базовых отношений могут существовать и уникальные отношения.

Рассмотренные выше методы формирования системы понятий и метапонятий, установления взаимосвязей и семантических отношений в разных сочетаниях применяются на этапе концептуализации при построении модели предметной области.

12.3 Этап формализации

На данном этапе все ключевые понятия и отношения выражаются на некотором формальном языке, который либо выбирается из числа уже существующих, либо создается заново. Другими словами, определяются состав средств и способы представления декларативных и процедурных знаний, осуществляется это представление и в итоге формируется описание решения задачи ЭС на предложенном (инженером по знаниям) формальном языке.

Выходом этапа формализации является описание того, как рассматриваемая задача может быть представлена в выбранном или разработанном формализме. Сюда относятся:

- указание способов представления знаний (фреймы, сценарии, семантические сети и т.д.)
- определение способов манипулирования этими знаниями (логический вывод, аналитическая модель, статистическая модель и др.) и интерпретации знаний.

12.4 Этап выполнения

Цель этого этапа — создание одного или нескольких прототипов ЭС, решающих требуемые задачи. Затем на данном этапе по результатам тестирования и опытной эксплуатации создается конечный продукт, пригодный для промышленного использования. Разработка прототипа

состоит в программировании его компонентов или выборе их из известных инструментальных средств и наполнении базы знаний.

Главное в создании первого прототипа ЭС-1 заключается в том, чтобы этот прототип обеспечил проверку адекватности идей, методов и способов представления знаний решаемым задачам. Создание первого прототипа должно подтвердить, что выбранные методы решений и способы представления пригодны для успешного решения, по крайней мере, ряда задач из актуальной предметной области, а также продемонстрировать тенденцию к получению высококачественных и эффективных решений для всех задач предметной области по мере увеличения объема знаний.

После разработки первого прототипа ЭС-1 круг предлагаемых для решения задач расширяется, и собираются пожелания и замечания, которые должны быть учтены в очередной версии системы ЭС-2.

Развитие ЭС-1 осуществляется за счет добавления:

- «дружественного» интерфейса,
- средств для исследования базы знаний и цепочек выводов, генерируемых системой,
- средств для сбора замечаний пользователей,
- средств хранения библиотеки задач, решенных системой.

При разработке ЭС-2, кроме перечисленных задач, решаются следующие:

- анализ функционирования системы при значительном расширении базы знаний;
- исследование возможностей системы в решении более широкого круга задач и принятие мер для обеспечения таких возможностей;
- анализ мнений пользователей о функционировании ЭС;
- разработка системы ввода-вывода, осуществляющей анализ или синтез предложений ограниченного естественного языка, позволяющей взаимодействовать с ЭС-2 в форме, близкой к форме стандартных учебников для данной области.

Если ЭС-2 успешно прошла этап тестирования, то она может классифицироваться как промышленная экспертная система.

12.5 Этап тестирования

В ходе данного этапа производится оценка выбранного способа представления знаний в ЭС в целом. Для этого инженер по знаниям подбирает примеры, обеспечивающие проверку всех возможностей разработанной ЭС.

Различают следующие источники неудач в работе системы.

1. ***Тестовые примеры.*** Показательные тестовые примеры являются наиболее очевидной причиной неудачной работы ЭС. Поэтому при подготовке тестовых примеров следует классифицировать их по подпроблемам предметной области, выделяя стандартные случаи, определяя границы трудных ситуаций и т.п.
2. ***Ввод-вывод.*** Ввод-вывод характеризуется данными, приобретенными в ходе диалога с экспертом, и заключениями, предъявленными ЭС в ходе объяснений. Методы приобретения данных могут не давать требуемых результатов, так как, например, задавались неправильные вопросы или собрана не вся необходимая информация.
3. ***Правила вывода.*** Наиболее распространенный источник ошибок в рассуждениях находится в правилах вывода. Важная причина здесь часто кроется в отсутствии учета взаимозависимости сформированных правил. Другая причина заключается в ошибочности, противоречивости и неполноте используемых правил. Если неверна посылка правила, то это может привести к употреблению правила в неподходящем контексте. Если ошибочно действие правила, то трудно предсказать конечный результат. Правило может быть ошибочно, если при корректности его условия и действия нарушено соответствие между ними.
4. ***Управляющие стратегии.*** Нередко к ошибкам в работе ЭС приводят применяемые управляющие стратегии. Последовательность, в которой данные рассматриваются ЭС, не только влияет на эффективность работы системы, но и может приводить к изменению конечного результата. Так, рассмотрение правила А до правила В способно привести к тому, что правило В всегда будет игнорироваться системой. Изменение стратегии бывает также необходимо и в случае неэффективной работы ЭС. Кроме того, недостатки в управляющих стратегиях могут привести к чрезмерно сложным заключениям и объяснениям ЭС.

12.6 Этап опытной эксплуатации

На этом этапе проверяется пригодность ЭС для конечного пользователя. Пригодность ЭС для пользователя определяется в основном удобством работы с ней и ее полезностью.

Полезность ЭС - ее способность в ходе диалога определять потребности пользователя, выявлять и устранять причины неудач в работе, а также удовлетворять указанные потребности пользователя (решать поставленные задачи).

Удобство работы с ЭС свойство естественности взаимодействия с ЭС (общение в привычном, не утомляющем пользователя виде), гибкость ЭС (способность системы настраиваться на различных пользователей, а также учитывать изменения в квалификации одного и того же пользователя) и устойчивость системы к ошибкам (способность не выходить из строя при ошибочных действиях неопытных пользователей).

В ходе разработки ЭС почти всегда осуществляется ее модификация. Выделяют следующие виды модификации системы:

- переформулирование понятий и требований,
- переконструирование представления знаний в системе
- усовершенствование прототипа.

В основу главы 12 положен материал учебного пособия [1], работы [6] и учебных курсов [7, 8].

13. ПРИМЕРЫ ПОСТРОЕНИЯ ЭКСПЕРТНЫХ СИСТЕМ

13.1 Пример построения экспертных диагностических систем

Рассмотрим методику формализации экспертных знаний на примере создания экспертных диагностических систем (ЭДС).

Целью создания ЭДС является определение состояния объекта диагностирования (ОД) и имеющихся в нем неисправностей.

Состояниями ОД могут быть:

- исправно,
- неисправно,
- работоспособно.

Неисправностями, например, радиоэлектронных ОД являются обрыв связи, замыкание проводников, неправильное функционирование элементов и т.д.

Число неисправностей может быть достаточно велико (несколько тысяч). В ОД может быть одновременно несколько неисправностей. В этом случае говорят, что неисправности кратные.

Введем следующие определения и ограничения:

- Разные неисправности ОД проявляются во внешней среде информационными параметрами.
- Совокупность значений информационных параметров определяет «информационный образ» (ИО) неисправности ОД.
- ИО может быть **полным**, то есть содержать всю необходимую информацию для постановки диагноза, или, соответственно, **неполным**.
- В случае неполного ИО постановка диагноза носит **вероятностный характер**.

Основой для построения эффективных ЭДС являются знания эксперта для постановки диагноза, записанные в виде информационных образов, и система представления знаний, встраиваемая в информационные системы обеспечения функционирования и контроля ОД, интегрируемые с соответствующей технической аппаратурой.

Для описания своих знаний эксперт с помощью инженера по знаниям должен выполнить следующее.

- Выделить множество всех неисправностей ОД, которые должна различать ЭДС.

- Выделить множество информативных (существенных) параметров, значения которых позволяют различить каждую неисправность ОД и поставить диагноз с некоторой вероятностью.

Для выбранных параметров следует выделить информативные значения или информативные диапазоны значений, которые могут быть как количественными, так и качественными. Например, точные количественные значения могут быть записаны: задержка 25 нс, задержка 30 нс и т.д. Количественный диапазон значений может быть записан: задержка 25-40 нс, 40-50 нс, 50 нс и выше. Качественный диапазон значений может быть записан: индикаторная лампа светится ярко, светится слабо, не светится.

Для более удобного дальнейшего использования качественный диапазон значений может быть закодирован, например, следующим образом:

- светится ярко $P_1 = +++$ (или $P_1 = 3$),
- светится слабо $P_1 = ++$ (или $P_1 = 2$),
- не светится $P_1 = +$ (или $P_1 = 1$).

Процедура получения информации по каждому из параметров определяется индивидуально в каждой конкретной системе диагностирования. Эта процедура может заключаться в автоматическом измерении параметров в ЭДС, в ручном измерении параметра с помощью приборов, качественном определении параметра, например, светится слабо, и т.д.

Процедура создания полных или неполных ИО каждой неисправности в алфавите значений информационных параметров может быть определена следующим образом. Составляются диагностические правила, определяющие вероятный диагноз на основе различных сочетаний диапазонов значений выбранных параметров ОД. Правила могут быть записаны в различной форме. Ниже приведена форма записи правил в виде таблицы 13.1.

Таблица 13.1. Диагностические правила

Номер	P1	P2	P3	Диагноз	Вероятность диагноза	Примечания
1		+++		Неисправен блок А1	0.95	
2	12-15	+		Неисправен блок А2	0.80	

Для записи правил с учетом изменений по времени следует ввести еще один параметр P_0 - время (еще один столбец в таблице). В этом случае диагноз может ставиться на основе нескольких строк таблицы, а в графе Примечания могут быть указаны использованные тесты. Диагностическая таблица в этом случае представлена в таблице 13.2.

Таблица 13.2. Динамические диагностические правила

Номер	P0	P1	P2	P3	Диагноз	Вероятность диагноза	Примечания
1	12:00	+	+	+			тест T1
2	12:15	++	++	+	Неисправен блок АЗ	0.90	

Для записи последовательности проведения тестовых процедур и задания ограничений (если они есть) на их проведение может быть предложен аналогичный механизм. Механизм записи последовательности проведения тестовых процедур в виде правил реализуется, например, следующим образом:

ЕСЛИ: $P_2 = 1$

ТО: тест = T_1, T_3, T_7 .

где T_1, T_3, T_7 - тестовые процедуры, подаваемые на ОД при активизации (срабатывании) соответствующей продукции.

В современных ЭДС применяются различные стратегии поиска решения и постановки диагноза, которые позволяют определить необходимые последовательности тестовых процедур. Однако приоритет в ЭС отдается прежде всего знаниям и опыту, а лишь затем логическому выводу.

13.2 Пример ЭС, основанной на правилах логического вывода и действующую в обратном порядке

Допустим, необходимо построить ЭС в области медицинской диагностики. В этом случае вряд ли нужно строить систему, использующую обучение на примерах, потому что имеется большое количество доступной информации, позволяющей непосредственно решать такие проблемы. К сожалению, эта информация приведена в неподходящем для обработки на компьютере виде.

Возьмите медицинскую энциклопедию и найдите в ней статью, например, о гриппе. Вы обнаружите, что в ней приведены все симптомы, причем они бесспорны. Другими словами, при наличии указанных симптомов всегда можно поставить точный диагноз.

Но чтобы использовать информацию, представленную в таком виде, необходимо обследовать пациента, решить, что у него грипп, а потом сверится с энциклопедией, чтобы убедиться, что у него соответствующие симптомы. Вместе с тем энциклопедия не позволяет определить болезнь так, как надо. Нам нужна не болезнь со множеством симптомов, а система, представляющая группу симптомов с последующим названием болезни. Именно это сейчас и попробуем сделать.

Идеальной будет такая ситуация, при которой машине предоставлено в приемлемом для нее виде множество определений в той или иной области, которые она сможет использовать примерно так же, как человек-эксперт.

С учетом байесовской системы логического вывода примем, что большая часть информации не является абсолютно точной, а носит вероятностный характер.

Полученный формат данных мы будем использовать для хранения симптомов. При слове «симптомы» создается впечатление, что речь идет исключительно о медицине, хотя речь может идти о чем угодно. Суть в том, что компьютер задает множество вопросов, содержащихся в виде символьных строк:

<Симптом_1>, <Симптом_2> и т.д.

№	Симптомы
1	Симптом_1
2	Симптом_2
N	Симптом_N

Например, Симптом_1 может означать строку «Много ли вы кашляете?», или, если вы пытаетесь отремонтировать неисправный автомобиль, — строку «Ослаб ли свет фар?».

Теперь оформим болезни.

№	Болезнь	p	[j, py, pn]
1	Болезнь_1	p1	[j, py, pn]1
2	Болезнь_2	p2	[j, py, pn]2
N	Болезнь_N	pn	[j, py, pn]n

В таком виде мы будем хранить информацию о болезнях. Это не обязательно должны быть болезни — могут быть любые результаты, и каждый оператор содержит один возможный исход и всю информацию, относящуюся к нему.

Поле «болезнь» характеризует название возможного исхода H , например «Грипп». Следующее поле p — это априорная вероятность такого исхода $P(H)$, т.е. вероятность исхода в случае отсутствия дополнительной информации. После этого идет ряд повторяющихся полей из трех элементов.

Первый элемент j — это номер соответствующего симптома (свидетельства, переменной, вопроса, если вы хотите назвать его по-другому).

Следующие два элемента — $P(E : H)$ и $P(E : \text{не } H)$ — соответственно вероятности получения ответа «Да» на этот вопрос, если возможный исход верен и неверен. Например:

2010	Грипп	0.01	(1, 0.9, 0.01); (2, 1, 0.01); (3, 0, 0.01)
------	-------	------	--

Здесь сказано существует априорная вероятность $P(H)=0.01$, что любой наугад взятый человек болеет гриппом.

Допустим, программа задает вопрос 1 (симптом 1). Тогда мы имеем $P(E : H)=0.9$ и $P(E : \text{не } H)=0.01$, а это означает, что если у пациента грипп, то он в девяти случаях из десяти ответит «да» на этот вопрос, а если у него нет гриппа, он ответит «да» лишь в одном случае из ста. Очевидно, ответ «да» подтверждает гипотезу о том, что у него грипп. Ответ «нет» позволяет предположить, что человек гриппом не болеет.

Так же и во второй группе симптомов (2, 1, 0.01). В этом случае $P(E : H)=0.9$, т.е. если у человека грипп, то этот симптом должен присутствовать. Соответствующий симптом может существовать и при отсутствии гриппа ($P(E : \text{не } H)=0.01$), но это маловероятно.

Вопрос 3 исключает грипп при ответе «да», потому что $P(E : H)=0$. Это может быть вопрос вроде такого: «наблюдаете ли вы такой симптом на протяжении большей части жизни?» — или что-нибудь вроде этого.

Нужно подумать, — а если вы хотите получить хорошие результаты, то и провести исследование, — чтобы установить обоснованные значения для этих вероятностей. И если быть честным, то получение такой информации — вероятно, труднейшая задача, в решении которой компьютер также сможет существенно помочь Вам.

Если вы напишете программу общего назначения, ее основой будет теорема Байеса, утверждающая:

$$P(H : E) = \frac{P(E : H) \cdot P(H)}{(P(E : H) \cdot P(H)) + (P(E : \text{не } H) \cdot P(\text{не } H))}.$$

В данном случае мы начинаем с того, что $P(H) = p$ для всех болезней. Программа задает соответствующий вопрос и в зависимости от ответа вычисляет $P(H : E)$. Ответ «да» подтверждает вышеуказанные расчеты, ответ «нет» тоже, но с $(1 - p)$ вместо p и $(1 - pn)$ вместо pn . Сделав так, мы забываем об этом, за исключением того, что априорная вероятность $P(H)$ заменяется на $P(H : E)$. Затем продолжается выполнение программы, но с учетом постоянной коррекции значения $P(H)$ по мере поступления новой информации.

Описывая алгоритм, мы можем разделить программу на несколько частей.

1. *Ввод данных.*
2. *Просмотр данных на предмет нахождения априорной вероятности $P(H)$.* Программа вырабатывает некоторые значения массива правил и размещает их в массиве RULEVALUE. Это делается для того, чтобы определить, какие вопросы (симптомы) являются самыми важными, и

выяснить, о чем спрашивать в первую очередь. Если вы вычислите для каждого вопроса

$$\text{RULEVALUE}[I] = \text{RULEVALUE}[I] + \text{ABS} (P(H : E) - P(H : \text{не } E)),$$

то получите значения возможных изменений вероятностей всех болезней, к которым они относятся.

3. *Программа находит самый важный вопрос и задает его.* Существует ряд вариантов, что делать с ответом: вы можете просто сказать: «да» или «нет». Можете попробовать сказать «не знаю», — изменений при этом не произойдет. Гораздо сложнее использовать шкалу от –5 до +5, чтобы выразить степень уверенности в ответе.

4. *Априорные вероятности заменяются новыми значениями при получении новых подтверждающих свидетельств.*

5. *Подсчитываются новые значения правил. Определяются также минимальное и максимальное значения для каждой болезни, основанные на существующих в данный момент априорных вероятностях и предположениях, что оставшиеся свидетельства будут говорить в пользу гипотезы или противоречить ей.* Важно выяснить: стоит ли данную гипотезу продолжать рассматривать или нет? *Гипотезы, которые не имеют смысла, просто отбрасываются. Те же из них, чьи минимальные значения выше определенного уровня, могут считаться возможными исходами.* После этого возвращаемся к пункту 3.

В основу главы 13 положен материал работы [6] и учебных курсов [7, 8].

14. ОСНОВЫ ЯЗЫКА ПРОГРАММИРОВАНИЯ ПРОЛОГ

Материал изложенных в данном разделе нужно рассматривать не как учебник по языку Пролог, а только как краткий «ликбез», который служит для иллюстрации принципов продукционного программирования, описанных ранее.

14.1 Синтаксис

14.1.1 Термы

Объекты данных в Прологе называются *термами*. Терм может быть константой, переменной или составным термом (структурой). *Константами* являются целые и действительные числа, например:

0, -1, 123.4, 0.23E-5,

(некоторые реализации Пролога не поддерживают действительные числа).

К константам относятся также атомы, такие, как:

голди, а, атом, +, :, 'Фред Блогс', [].

Атом есть любая последовательность символов, заключенная в одинарные кавычки. Кавычки опускаются, если и без них атом можно отличить от символов, используемых для обозначения переменных. Приведем еще несколько примеров атомов:

abcd, фред, ':', Джо.

Полный синтаксис атомов описан ниже.

Как и в других языках программирования, константы обозначают конкретные элементарные объекты, а все другие типы данных в Прологе составлены из сочетаний констант и переменных.

Имена переменных начинаются с заглавных букв или с символа подчеркивания «_». Примеры переменных:

X, Переменная, _3, _переменная.

Если переменная используется только один раз, необязательно называть ее. Она может быть записана как анонимная переменная, состоящая из одного символа подчеркивания "_". Переменные, подобно атомам, являются элементарными объектами языка Пролог.

Завершает список синтаксических единиц сложный терм, или структура. Все, что не может быть отнесено к переменной или константе, называется сложным термом. Следовательно, сложный терм состоит из констант и переменных.

Теперь перейдем к более детальному описанию термов.

14.1.2 Константы

Константы известны всем программистам. В Прологе константа может быть атомом или числом.

14.1.2.1 Атомы

Атом представляет собой произвольную последовательность символов, заключенную в одинарные кавычки. Одинарный символ кавычки, встречающийся внутри атома, записывается дважды. Когда атом выводится на печать, внешние символы кавычек обычно не печатаются. Существует несколько исключений, когда атомы необязательно записывать в кавычках. Вот эти исключения:

1. атом, состоящий только из чисел, букв и символа подчеркивания и начинающийся со строчной буквы;
2. атом, состоящий целиком из специальных символов. К специальным символам относятся: + - * / ^ = : ; ? @ \$ & .

Заметим, что атом, начинающийся с /*, будет воспринят как начало комментария, если он не заключен в одинарные кавычки.

Как правило, в программах на Прологе используются атомы без кавычек.

Атом, который необязательно заключать в кавычки, может быть записан и в кавычках. Запись с внешними кавычками и без них определяет один и тот же атом.

Внимание: допустимы случаи, когда атом не содержит ни одного символа (так называемый «нулевой атом») или содержит непечатаемые символы. (В Прологе имеются предикаты для построения атомов, содержащих непечатаемые или управляющие символы.) При выводе таких атомов на печать могут возникнуть ошибки.

14.1.2.2 Числа

Большинство реализации Пролога поддерживают целые и действительные числа. Чтобы выяснить, каковы диапазоны и точность чисел, следует обратиться к руководству по конкретной реализации.

14.1.3 Переменные

Понятие переменной в Прологе отличается от принятого во многих языках программирования. Переменная не рассматривается как выделенный участок памяти. Она служит для обозначения объекта, на который нельзя

сослаться по имени. Переменную можно считать локальным именем для некоторого объекта.

Синтаксис переменной довольно прост. Она должна начинаться с прописной буквы или символа подчеркивания и содержать только символы букв, цифр и подчеркивания.

Переменная, состоящая только из символа подчеркивания, называется анонимной и используется в том случае, если имя переменной несущественно.

14.1.3.1 Область действия переменных

Областью действия переменной является утверждение. В пределах утверждения одно и то же имя принадлежит одной и той же переменной. Два утверждения могут использовать одно имя переменной совершенно различным образом. Правило определения области действия переменной справедливо также в случае рекурсии и в том случае, когда несколько утверждений имеют одну и ту же головную цель. Этот вопрос будет рассмотрен далее.

Единственным исключением из правила определения области действия переменных является анонимная переменная, например, "_" в цели любит ($x, _$). Каждая анонимная переменная есть отдельная сущность. Она применяется тогда, когда конкретное значение переменной несущественно для данного утверждения. Таким образом, каждая анонимная переменная четко отличается от всех других анонимных переменных в утверждении.

Переменные, отличные от анонимных, называются *именованными*, а неконкретизированные (переменные, которым не было присвоено значение) называются *свободными*.

14.1.4 Сложные термы, или структуры

Структура состоит из атома, называемого главным функтором, и последовательности термов, называемых компонентами структуры. Компоненты разделяются запятыми и заключаются в круглые скобки.

Приведем примеры структурированных термов:

собака (рекс) , родитель (X, Y) .

Число компонент в структуре называется арностью структуры. Так, в данном примере структура собака имеет арность 1 (записывается как собака/1), а структура родитель — арность 2 (родитель/2). Заметим, что атом можно рассматривать как структуру арности 0.

Для некоторых типов структур допустимо использование альтернативных форм синтаксиса. Это синтаксис операторов для структур

арности 1 и 2, синтаксис списков для структур в форме списков и синтаксис строк для структур, являющихся списками кодов символов.

14.1.5 Синтаксис операторов

Структуры арности 1 и 2 могут быть записаны в операторной форме, если атом, используемый как главный функтор в структуре, объявить оператором.

14.1.6 Синтаксис списков

В сущности, список есть не что иное, как некоторая структура арности 2. Данная структура становится интересной и чрезвычайно полезной в случае, когда вторая компонента тоже является списком. Вследствие важности таких структур в Прологе имеются специальные средства для записи списков.

14.1.7 Синтаксис строк

Строка определяется как список кодов символов. Коды символов имеют особое значение в языках программирования. Они выступают как средство связи компьютера с внешним миром. В большинстве реализации Пролога существует специальный синтаксис для записи строк. Он подобен синтаксису атомов. Строкой является любая последовательность символов, которые могут быть напечатаны (кроме двойных кавычек), заключенная в двойные кавычки. Двойные кавычки в пределах строки записываются дважды `""`.

В некоторых реализациях Пролога строки рассматриваются как определенный тип объектов подобно атомам или спискам. Для их обработки вводятся специальные встроенные предикаты. В других реализациях строки обрабатываются в точности так же, как списки, при этом используются встроенные предикаты для обработки списков. Поскольку все строки могут быть определены как атомы или как списки целых чисел, и понятие строки является чисто синтаксическим, мы не будем более к нему возвращаться.

14.2 Утверждения

Программа на Прологе есть совокупность утверждений. Утверждения состоят из целей и хранятся в базе данных Пролога. Таким образом, база данных Пролога может рассматриваться как программа на Прологе. В конце утверждения ставится точка `«.»`. Иногда утверждение называется **предложением**.

Основная операция Пролога — доказательство целей, входящих в утверждение.

Существуют два типа утверждений:

- **факт** — это одиночная цель, которая, безусловно, истинна;
- **правило** — состоит из одной головной цели и одной или более хвостовых целей, которые истинны при некоторых условиях.

Правило обычно имеет несколько хвостовых целей в форме конъюнкции целей.

Конъюнкцию можно рассматривать как логическую функцию И. Таким образом, правило согласовано, если согласованы все его хвостовые цели.

Примеры фактов:

собака (рекс) . родитель (голди.рекс) .

Примеры правил:

собака (X) :- родитель (X.Y), собака (Y) . человек (X) :-
мужчина (X) .

Разница между правилами и фактами чисто семантическая. Хотя для правил мы используем синтаксис операторов (более подробное рассмотрение операторного и процедурного синтаксисов выходит за рамки нашего курса), нет никакого синтаксического различия между правилом и фактом.

Так, правило

собака (X) :- родитель (X,Y) , собака (Y) .

может быть задано как

:-собака (X) ' , ' родитель (X.Y) .собака (Y) .

Запись верна, поскольку :- является оператором "при условии, что", а ' , ' — это оператор конъюнкции. Однако удобнее записывать это как

собака (X) :-родитель (X.Y) ,собака (Y) .

и читать следующим образом: " X — собака при условии, что родителем X является Y и Y — собака".

14.3 Запросы

После записи утверждений в базу данных вычисления могут быть инициированы вводом запроса.

Запрос выглядит так же, как и целевое утверждение, образуется и обрабатывается по тем же правилам, но он не входит в базу данных (программу). В Прологе вычислительная часть программы и данные имеют одинаковый синтаксис. Программа обладает как декларативной, так и процедурной семантикой. Мы отложим обсуждение этого вопроса до последующих лекций. Запрос обозначается в Прологе утверждением ?-, имеющим арность 1. Обычно запрос записывается в операторной форме: за

знаком ? – следует ряд хвостовых целевых утверждений (чаще всего в виде конъюнкции).

Приведем примеры запросов:

?-собака (X) . ?- родитель (X,Y) , собака (Y) .

или, иначе,

'?- '(собака (X)) C?-' ' , ' (родитель (X,Y" , собака (Y)) .

Последняя запись неудобна тем, что разделитель аргументов в структуре совпадает с символом конъюнкции. Программисту нужно помнить о различных значениях символа ','.

Запрос иногда называют управляющей командой (директивой), так как он требует от Пролог-системы выполнения некоторых действий. Во многих реализациях Пролога для управляющей команды используется альтернативный символ, а символ ? – обозначает приглашение верхнего уровня интерпретатора Пролога. Альтернативным символом является :-.

:-write(собака) .

- это управляющая команда, в результате выполнения которой печатается атом собака. Управляющие команды будут рассмотрены ниже при описании ввода программ.

14.4 Ввод программ

Введение списка утверждений в Пролог-систему осуществляется с помощью встроенного предиката `consult`. Аргументом предиката `consult` является атом, который обычно интерпретируется системой как имя файла, содержащего текст программы на Прологе. Файл открывается, и его содержимое записывается в базу данных. Если в файле встречаются управляющие команды, они сразу же выполняются. Возможен случай, когда файл не содержит ничего, кроме управляющих команд для загрузки других файлов. Для ввода утверждений с терминала в большинстве реализации Пролога имеется специальный атом, обычно `user`. С его помощью утверждения записываются в базу данных, а управляющие команды выполняются немедленно.

Помимо предиката `consult`, в Прологе существует предикат `reconsult`. Он работает аналогичным образом. Но перед добавлением утверждений к базе данных из нее автоматически удаляются те утверждения, головные цели которых сопоставимы с целями, содержащимися в файле перезагрузки. Такой механизм позволяет вводить изменения в базу данных. В Прологе имеются и другие методы добавления и удаления утверждений из базы данных. Некоторые реализации языка поддерживают модульную структуру, позволяющую разрабатывать модульные программы.

В заключение раздела дадим формальное определение синтаксиса Пролога, используя форму записи Бэкуса-Наура, иногда называемую бэкусовской нормальной формой (БНФ).

```
запрос ::= голова утверждения
правило ::= голова утверждения :- хвост утверждения
факт ::= голова утверждения
голова утверждения ::= атом | структура
хвост утверждения ::= атом структура,
термы ::= терм [, термы]
терм ::= число | переменная | атом | структура
структура ::= атом (термы)
```

Данное определение синтаксиса не включает операторную, списковую и строковую формы записи. Однако, любая программа на Прологе может быть написана с использованием вышеприведенного синтаксиса. Специальные формы только упрощают понимание программы. Как мы видим, синтаксис Пролога не требует пространного объяснения. Но для написания хороших программ необходимо глубокое понимание языка.

14.5 Унификация

Одним из наиболее важных аспектов программирования на Прологе являются понятия унификации (отождествления) и конкретизации переменных.

Пролог пытается отождествить термы при доказательстве, или согласовании, целевого утверждения. Например, для согласования запроса ? - собака(X) целевое утверждение собака (X) было отождествлено с фактом собака (рекс), в результате чего переменная X стала конкретизированной: X= рекс.

Переменные, входящие в утверждения, отождествляются особым образом — сопоставляются. Факт доказывается для всех значений переменной (переменных). Правило доказывается для всех значений переменных в головном целевом утверждении при условии, что хвостовые целевые утверждения доказаны. Предполагается, что переменные в фактах и головных целевых утверждениях связаны квантором всеобщности. Переменные принимают конкретные значения на время доказательства целевого утверждения.

В том случае, когда переменные содержатся только в хвостовых целевых утверждениях, правило считается доказанным, если хвостовое целевое утверждение истинно для одного или более значений переменных. Переменные, содержащиеся только в хвостовых целевых утверждениях, связаны квантором существования. Таким образом, они принимают

конкретные значения на то время, когда целевое утверждение, в котором переменные были согласованы, остается доказанным.

Терм X сопоставляется с термом Y по следующим правилам. Если X и Y — константы, то они сопоставимы, только если они одинаковы. Если X является константой или структурой, а Y — неконкретизированной переменной, то X и Y сопоставимы и Y принимает значение X (и наоборот). Если X и Y — структуры, то они сопоставимы тогда и только тогда, когда у них одни и те же главный функтор и аргументы и каждая из их соответствующих компонент сопоставима. Если X и Y — неконкретизированные (свободные) переменные, то они сопоставимы, в этом случае говорят, что они сцеплены. В таблице 14.1 приведены примеры отождествимых и неотождествимых термов.

Таблица 14.1. Иллюстрация унификации

Терм 1	Терм 2	Отождествимы ?
Джек (X)	Джек (человек)	да: $X = \text{человек}$
Джек (личность)	Джек (человек)	нет
Джек (X, X)	Джек (23, 23)	да: $X = 23$
Джек (X, X)	Джек (12, 23)	нет
Джек (.)	Джек(12, 23)	да
$f(Y, Z)$	X	да: $X = f(Y, Z)$
X	Z	да: $X = Z$

Заметим, что Пролог находит наиболее общий унификатор термов. В последнем примере (табл.14.1) существует бесконечное число унификаторов:

$X=1, Z=2; X=2, Z=2; \dots$

но Пролог находит наиболее общий: $X=Z$.

Следует сказать, что в большинстве реализаций Пролога для повышения эффективности его работы допускается существование циклических унификаторов. Например, попытка отождествить термы $f(X)$ и X приведет к циклическому унификатору $X=f(X)$, который определяет бесконечный терм $f(f(f(f(f(\dots)))))$. В программе это иногда вызывает бесконечный цикл.

Возможность отождествления двух термов проверяется с помощью оператора $=$.

Ответом на запрос

$?- 3+2=5.$

будет

нет

так как термы не отождествимы (оператор не вычисляет значения своих аргументов), но попытка доказать

`?-строка(поз(X)) -строка(поз(23)).`

закончится успехом при

`X=23.`

Унификация часто используется для доступа к подкомпонентам термов. Так, в вышеприведенном примере `X` конкретизируется первой компонентой терма `поз(23)`, который в свою очередь является компонентой терма `строка`.

Бывают случаи, когда надо проверить, идентичны ли два терма. Выполнение оператора `=` заканчивается успехом, если его аргументы — идентичные термы. Следовательно, запрос

`?-строка(поз(X)) ==строка(поз(23)).`

дает ответ `нет`, поскольку подтерм `X` в левой части (`X` — свободная переменная) не идентичен подтерму `23` в правой части. Однако запрос

`?- строка(поз(23)) ==строка(поз(23)).`

дает ответ `да`.

Отрицания операторов `=` и `-` записываются как `\=` и `\=` соответственно.

14.6 Выражения

В этой части показано, каким образом Пролог выполняет арифметические операции. Будут описаны арифметические операторы и их использование в выражениях, а также рассмотрены встроенные предикаты, служащие для вычисления и сравнения арифметических выражений.

Язык Пролог не предназначен для программирования задач с большим количеством арифметических операций. Для этого используются процедурные языки программирования. Однако в любую Пролог-систему включаются все обычные арифметические операторы:

<code>+</code>	сложение,
<code>-</code>	вычитание,
<code>*</code>	умножение,
<code>/</code>	деление,
<code>mod</code>	остаток от деления целых чисел,
<code>div</code>	целочисленное деление.

В некоторых реализациях языка Пролог присутствует более широкий набор встроенных арифметических операторов.

Диапазоны чисел, входящих в арифметические выражения, зависят от реализации Пролога. Например, система ICLPROLOG оперирует целыми числами со знаком в диапазоне: $-8388606 \dots 8388607$.

14.6.1 Арифметические выражения

Арифметическое выражение является числом или структурой. В структуру может входить одна или более компонент, таких, как числа, арифметические операторы, арифметические списковые выражения, переменная, конкретизированная арифметическим выражением, унарные функторы, функторы преобразования и арифметические функторы.

- **Числа.** Числа и их диапазоны определяются в конкретной реализации Пролога.
- **Арифметические операторы** $+$ $-$ $*$ $/$ mod div
- **Арифметические списковые выражения.** Если x — арифметическое выражение, то список $[x]$ также является арифметическим выражением, например $[1, 2, 3]$. Первый элемент списка используется как операнд в выражении. Скажем, x is $([1, 2, 3] + 5)$ имеет значение 6.

Арифметические списковые выражения полезны и при обработке символов, поскольку последние могут рассматриваться как небольшие целые числа. Например, символ "a" эквивалентен $[97]$ и, будучи использован в выражении, вычисляется как 97. Поэтому значение выражения $"p" + "A" - "a"$ равно 80, что соответствует коду ASCII для "P".

Переменная, конкретизированная арифметическим выражением. Примеры: $x - 5 + 2$ и $y - 3 * (2 + A)$.

Унарные функторы. Примеры: $+(x)$ и $-(y)$.

Функторы преобразования. В некоторых реализациях Пролога имеется арифметика с плавающей точкой, а следовательно, и функторы преобразования. Например:

`float (X)` преобразует целое число x в число с плавающей точкой.

Математические функторы. Пример: квадрат (x) объявлен как оператор и эквивалентен арифметическому выражению $(x * x)$.

14.6.2 Арифметические операторы

Атомы $+$, $-$, $*$, $/$, mod , div — обычные атомы Пролога и могут использоваться почти в любом контексте. Указанные атомы — не встроенные предикаты, а функторы, имеющие силу только в пределах арифметических выражений. Они определены как инфиксные операторы.

Эти атомы являются главными функторами в структуре, а сама структура может принимать только описанные выше формы.

Арифметический оператор выполняется следующим образом. Во-первых, вычисляются арифметические выражения по обе стороны оператора. Во-вторых, над результатом вычислений выполняется нужная операция.

Арифметические операторы определяются Пролог-системой. Если мы напишем предикат

```
среднее(X,Y,Z) :- Z is (X+Y)/2.
```

то, хотя можно определить среднее как оператор,

```
?- op(250^x, среднее).
```

но Пролог выдаст сообщение об ошибке, если встретит выражение

```
Z is X среднее Y.
```

Это произойдет потому, что x среднее y не образует арифметического выражения, а среднее не является арифметическим оператором, определенным в системе.

14.6.3 Вычисление арифметических выражений

В Прологе не допускаются присваивания вида `Сумма=2+4`.

Выражение такого типа вычисляется только с помощью системного предиката `is`, например:

```
Сумма is 2+4.
```

Предикат `is` определен как инфиксный оператор. Его левый аргумент — или число, или неконкретизированная переменная, а правый аргумент — арифметическое выражение.

Попытка доказательства целевого утверждения $X \text{ is } Y$ заканчивается успехом в одном из следующих случаев:

- X — неконкретизированная переменная, а результат вычисления выражения Y есть число;
- X — число, которое равно результату вычисления выражения Y .
Цель $X \text{ is } Y$ не имеет побочных эффектов и не может быть согласована вновь. Если X не является неконкретизированной переменной или числом либо если Y — не арифметическое выражение, возникает ошибка.

Примеры:

- | | |
|-------------------------|---|
| <code>D is 10-5</code> | - заканчивается успехом и <code>D</code> становится равным 5, |
| <code>4 is 2*4-4</code> | - заканчивается успехом, |

$2*4-4$	is	4	- заканчивается неудачей,
a	is	$3+3$	- заканчивается неудачей,
x	is	$4+a$	- заканчивается неудачей,
2	is	$4-x$	- заканчивается неудачей.

Необходимо обратить внимание, что предикат `is` требует, чтобы его первый аргумент был числом или неконкретизированной переменной. Поэтому `m-2 is 3` записано неверно. Предикат `is` не является встроенным решателем уравнений.

14.6.4 Сравнение результатов арифметических выражений

Системные предикаты `==`, `=\=`, `>`, `<`, `>=` и `<=` определены как инфиксные операторы и применяются для сравнения результатов двух арифметических выражений.

Для предиката `@` доказательство целевого утверждения `x@y` заканчивается успехом, если результаты вычисления арифметических выражений `x` и `y` находятся в таком отношении друг к другу, которое задается предикатом `@`.

Такое целевое утверждение не имеет побочных эффектов и не может быть согласовано вновь. Если `x` или `y` — не арифметические выражения, возникает ошибка.

С помощью предикатов описываются следующие отношения:

$x == y-x$	- равно y ,
$x =\!= y-x$	- не равно y ,
$x < y-x$	- меньше y ,
$x > y-x$	- больше y ,
$x <= y-x$	- меньше или равно y ,
$x >= y-x$	- больше или равно y .

Использование предикатов иллюстрируют такие примеры:

$a > 5$	- заканчивается неудачей,
$5+2+7 > 5+2$	- заканчивается успехом,
$3+2 == 5$	- заканчивается успехом,
$3+2 < 5$	- заканчивается неудачей,
$2+1 =\!= 1$	- заканчивается успехом,
$n > 3$	- заканчивается успехом, если n больше 3, и неудачей

в противном случае.

14.7 Структуры данных

Термы Пролога позволяют выразить самую разнообразную информацию. Рассмотрим два вида широко используемых структур данных: списки и бинарные деревья, и покажем, как они представляются термами Пролога.

14.7.1 Списки

Задачи, связанные с обработкой списков, на практике встречаются очень часто. Например понадобилось составить список студентов, находящихся в аудитории. С помощью Пролога возможно определить список как последовательность термов, заключенных в скобки. Приведем примеры правильно построенных списков Пролога:

```
[джек, джон, фред, джилл, джон]
[имя (джон, смит), возраст (джек, 24), X]
[X.У.дата (12, январь, 1986) ,X]
[]
```

Запись $[H|T]$ определяет список, полученный добавлением H в начало списка T . Говорят, что H — голова, а T — хвост списка $[H|T]$. На вопрос

```
?-L=[a|[b, c, d]].
```

будет получен ответ

```
L=[a, b, c, d],
```

а на запрос

```
?-L=[a, b, c, d], L2=[2|L]
```

— ответ

```
L=[a, b, c, d], L2=[2, a, b, c, d]
```

Запись $[H|T]$ используется для того, чтобы определить голову и хвост списка. Так, запрос

```
?- [X|Y]=[a, b, c].
```

дает

```
X=a, Y=[b, c]
```

Заметим, что употребление имен переменных H и T необязательно. Кроме записи вида $[H|T]$, для выборки термов используются переменные. Запрос

```
?-[a, X, Y]=[a, b, c].
```

определит значения

```
X=b Y=c
```

а запрос

?- [личность (X) | T] = [личность (джон), a, b].

значения

X=джон T=[a, b].

14.7.2 Стандартные функции обработки списков

Покажем на примерах, как можно использовать запись вида [H|T] вместе с рекурсией для определения некоторых полезных целевых утверждений для работы со списками.

Принадлежность списку. Сформулируем задачу проверки принадлежности данного терма списку.

Граничное условие:

терм R содержится в списке [H|T], если R=H.

Рекурсивное условие:

терм R содержится в списке [H|T], если R содержится в списке T.

Вариант записи граничного условия на Прологе имеет вид:

содержится (R, L) :-

L=[H|T],

H=R.

Вариант записи рекурсивного условия на Прологе имеет вид:

содержится (R, L) :-

L=[H|T],

содержится (R, T).

Цель L=[H|T] в теле обоих утверждений служит для того, чтобы разделить список L на голову и хвост.

Можно улучшить программу, если учесть тот факт, что Пролог сначала сопоставляет с целью голову утверждения, а затем пытается согласовать его тело. Новая процедура, которую мы назовем «принадлежит», определяется таким образом:

принадлежит (R, [R|T]).

принадлежит (R, [H|T]) :- принадлежит (R, T).

На запрос

?- принадлежит(a, [a, b, c]).

будет получен ответ да.

На запрос

?- принадлежит(b, [a, b, c]).

- ответ да.

Но на запрос

?- принадлежит(d, (a, b, c)).

Пролог дает ответ нет.

В большинстве реализации Пролога предикат «принадлежит» является встроенным.

Соединение двух списков. Задача присоединения списка Q к списку P, в результате чего получается список R, формулируется следующим образом:

Граничное условие:

присоединение списка Q к [] дает Q.

Рекурсивное условие:

Присоединение списка Q к концу списка P выполняется так: Q присоединяется к хвосту P, а затем спереди добавляется голова P.

Определение можно непосредственно написать на Прологе:

соединить([], 0, 0).

соединить(P, Q, R) :-

P=[HP|TP],

соединить(TP, Q, TR),

R=[HP|TR].

Однако, как и в предыдущем примере, воспользуемся тем, что Пролог сопоставляет с целью голову утверждения, прежде чем пытаться согласовать тело:

присоединить([], Q, Q).

присоединить(HP|TP, Q, [HP|TR]) :-

присоединить(TP, Q, TR).

На запрос

?- присоединить([a, b, c], [d, e], L).

будет получен ответ L = [a, b, c, d].

Но на запрос

?- присоединить([a, b], [c, d], [e, f]).

ответом будет нет.

Часто процедура присоединить используется для получения списков, находящихся слева и справа от данного элемента:

```
присоединить (L [джим, р], [джек, билл, джим, тим, джим, боб]).
```

```
L = [джек, билл]
```

```
R = [тим, джим, боб]
```

```
другие решения (да/нет)? да
```

```
L=[джек, билл, джим, тим]
```

```
R=[боб]
```

```
другие решения (да/нет)? да
```

```
других решений нет
```

Индексирование списка. Задача получения N-го термина в списке определяется следующим образом:

Граничное условие:

Первый терм в списке [H|T] есть H.

Рекурсивное условие:

N-й терм в списке [H|T] является (N-I)-м термом в списке T.

Данному определению соответствует программа:

```
/* Граничное условие:
```

```
получить ([H|T], 1, H). /* Рекурсивное условие:
```

```
получить ([H|T], N, Y) :-
```

```
M is N-1,
```

```
получить (T, M, Y).
```

Построение списков из фактов. Иногда бывает полезно представить в виде списка информацию, содержащуюся в известных фактах. В большинстве реализаций Пролога есть необходимые для этого предикаты:

`bagof(X, Y, L)` — определяет список термов L, конкретизирующих переменную X как аргумент предиката Y, которые делают истинным предикат Y.

`setof(X, Y, L)` — все сказанное о предикате `bagof` относится и к `setof`, за исключением того, что список L отсортирован и из него удалены все повторения.

Если имеются факты:

```
собака(пекс) .
```

```
собака(голди) .
```

```
собака(фидо) .
```

```
собака(реке) .
```

то на запрос

?- bagof(D, собака(D), L),

будет получен ответ

L=[реке, голди, фидо, рекс]

в то время как

?- setof(D, собака(D), L).

дает значение

L=[фидо, голди, рекс]

14.7.3 Сложение многочленов

Теперь мы достаточно подготовлены к тому, чтобы использовать списки для решения задач. Вопрос, которым мы займемся, — представление и сложение многочленов.

Представление многочленов. Посмотрим, как можно представить многочлен вида

$$P(x) = 3 + 3x - 4x^3 + 2x^9$$

$$Q(x) = 4x + x^2 - 3x^3 + 7x^4 + 8x^5$$

Заметим, что каждое подвыражение (такое, как $3x^3$, $3x$, 3) имеет самое большее две переменные компоненты: число, стоящее перед x , называемое коэффициентом, и число, стоящее после $^$ - степень. Следовательно, подвыражение представляется термом

$x(\text{Коэффициент}, \text{Степень})$

Так, $5x^2$ записывается как $x(5, 2)$, x^3 представляется как $x(1, 3)$, а поскольку x^0 равно 1, подвыражению 5 соответствует терм $x(5, 0)$.

Теперь запишем многочлен в виде списка. Приведенный выше многочлен $P(x)$, например, будет выглядеть следующим образом:

$[x(3, 0), '+', x(3, 1), '-', x(4, 3), '+', x(2, 9)]$

Воспользуемся тем, что многочлен

$$3 + 3x - 4x^3 + 2x^9$$

допускает замену на эквивалентный

$$3 + 3x + (-4)x^3 + 2x^9$$

Тогда он выражается списком:

$[x(3, 0), '+', x(3, 1), '+', x(-4, 3), '+', x(2, 9)]$

В такой записи между термами всегда стоят знаки '+'. Следовательно, их можно опустить, и многочлен принимает окончательный вид:

$[x(3, 0), x(3, 1), x(-4, 3), x(2, 9)]$

Подразумевается, что между всеми термами списка стоят знаки '+'. Представлением многочлена $Q(x)$ будет

$$[x(4, 1), x(1, 2), x(-3, 3), x(7, 4), x(8, 5)]$$

Сложение многочленов. Теперь напишем целевые утверждения для сложения двух многочленов. Сложение многочленов

$$3-2x^2+4x^3+6x^6$$

$$-1+3x^2-4x^3$$

в результате дает

$$2+x^2+6x^6$$

Аргументами целевого утверждения являются многочлены, представленные в виде списков. Ответ будет получен также в виде списка.

Сложение многочлена P с многочленом Q осуществляется следующим образом.

Граничное условие:

P , складываемый с [], дает P .

[], складываемый с Q , дает Q .

Рекурсивное условие:

При сложении P с Q , в результате чего получается многочлен R , возможны 4 случая:

1. Степень первого терма в P меньше, чем степень первого терма в Q . В этом случае первый терм многочлена P образует первый терм в R , а хвост R получается при прибавлении хвоста P к Q . Например, если P и Q имеют вид

$$P(x) = 3x^2 + 5x^3$$

$$Q(x) = 4x^3 + 3x^4$$

то первый терм $R(x)$ равен $3x^2$ (первому терму в $P(x)$). Хвост $R(x)$ равен $9x^3 + 3x^4$, т.е. результату сложения $Q(x)$ и хвоста $P(x)$;

2. Степень первого терма в P больше степени первого терма в Q . В данном случае первый терм в Q образует первый терм в R , а хвост R получается при прибавлении P к хвосту Q . Например, если

$$P(x) = 2x^3 + 5x^4$$

$$Q(x) = 3x^3 - x^4$$

то первый терм $R(x)$ равен $3x^3$ (первому терму в $Q(x)$), а хвост $R(x)$ равен $2x^3 + 4x^4$ (результату сложения $P(x)$ и хвоста $Q(x)$);

3. Степени первых термов в P и Q равны, а сумма их коэффициентов отлична от нуля. В таком случае первый терм в R имеет коэффициент, равный сумме коэффициентов первых термов в P и Q . Степень первого

терма в R равна степени первого термина в P (или Q). Хвост R получается при сложении хвоста P и хвоста Q . Например, если P и Q имеют вид

$$P(x) = 2x + 3x^3$$

$$Q(x) = 3x + 4x^4$$

то первый терм многочлена $R(x)$ равен $5x$ (результату сложения первого термина в $P(x)$ с первым термином в $Q(x)$). Хвост $R(x)$ равен $3x^3 + 4x^4$ (результату сложения хвоста $P(x)$ и хвоста $Q(x)$);

4. Степени первых термов в P и Q одинаковы, но сумма коэффициентов равна нулю. В данном случае многочлен R равен результату сложения хвоста P с хвостом Q . Например, если

$$p(x) = 2 + 2x$$

$$Q(x) = 2 - 3x^2$$

то

$$R(x) = 2x - 3x^2$$

(это результат сложения хвостов многочленов $P(x)$ и $Q(x)$).

Рассмотренный процесс сложения многочленов можно непосредственно записать на языке Пролог.

```
/* Граничные условия
слож_мн([], Q, Q).
слож_мн(P, [], P).
/* Рекурсивное условие
/* (a)
слож_мн([x(Pc, Pp) | Pt], [x(Qc, Qp) | Qt],
[x(Pc, Pp) | Rt]) :-
PpQp,
слож_мн(Pt, [x(Qc, Qp) | Qt], Rt).
/* (б)
слож_мн([x(Pc, Pp) | Pt], [x(Qc, Qp) | Qt],
[x(Qc, Qp) | Rt]) :-
PpQp,
слож_мн([x(Pc, Pp) | Pt], Qt, Rt).
/* (в)
слож_мн([x(Pc, Pp) | Pt], [x(Qc, Pp) | Qt],
[x(Rc, Pp) | Rt]) :-
Rc is Pc+Qc,
Rc \= 0,
```

```

слож_мн(Pt, Qt, Rt).
/* (r)
слож_мн([x(Pc, Pp) | Pt],
[x(Qc.Pp) | Qt], Rt) :-
Re is Pc+Qc,
Rc =:= 0,
слож_мн(Pt, Qt, Rt).

```

Заметим, что в двух последних утверждениях проверка на равенство осуществляется следующим образом: степени первых термов складываемых утверждений обозначает одна и та же переменная Pp .

Списки как термы. В начале лекции мы упомянули о том, что список представляется с помощью терма. Такой терм имеет функтор '.', два аргумента и определяется рекурсивно. Первый аргумент является головой списка, а второй — термом, обозначающим хвост списка. Пустой список обозначается []. Тогда список $[a, b]$ эквивалентен терму $.(a, .(b, []))$.

Таким образом, из списков, как и из термов, можно создавать вложенные структуры. Поэтому выражение

```
[[a, b], [c, d], [a], a]
```

есть правильно записанный список, и на запрос:

```
?- [H|T] = [[a, b], c].
```

Пролог дает ответ:

```
H = [a, b] T = [c]
```

В основу главы 14 положен материал учебного курса [7].

15. НЕЙРОННЫЕ СЕТИ

15.1 Проблемы решаемые нейронными сетями

В настоящее время нейронные сети (НС) представляют собой формализованную модель функционирования человеческого мозга и используются для решения широкого спектра задач. Ниже перечисляются основные проблемные вопросы, решаемые с помощью искусственных нейронных сетей (ИНС) в соответствии с классификацией приведенной в работе [3].

1. Классификация/распознавание образов. Задача состоит в указании принадлежности входного образа (например, речевого сигнала или рукописного символа), представленного вектором признаков, одному или нескольким предварительно определенным классам. К известным приложениям относятся распознавание букв, распознавание речи, классификация сигнала электрокардиограммы, классификация клеток крови, распознавание отпечатков пальцев, а также, лиц.

2. Кластеризация/категоризация. При решении задачи кластеризации, которая известна также как классификация образов «без учителя» отсутствует обучающая выборка с метками классов. Алгоритм кластеризации основан на подобии образов и размещает близкие образы в один кластер. Известны случаи применения кластеризации для извлечения знаний, сжатия данных и исследования свойств данных.

3. Аппроксимация функций. Предположим, что имеется обучающая выборка (пары данных вход-выход), которая генерируется неизвестной функцией (x), искаженной шумом. Задача аппроксимации состоит в нахождении оценки неизвестной функции (x). Аппроксимация функций необходима при решении многочисленных инженерных и научных задач моделирования.

4. Предсказание/прогноз. Пусть заданы n дискретных отсчетов в последовательные моменты времени t , Задача состоит в предсказании значения $y(t)$ в некоторый будущий момент времени $t > n+1$.

Предсказание/прогноз имеют значительное влияние на принятие решений в бизнесе, науке и технике. Предсказание цен на фондовой бирже и прогноз погоды являются типичными приложениями техники предсказания/прогноза. Предсказание является также основной задачей, решаемой сомообучаемыми мобильными автономными системами в условиях адаптации к незнакомой окружающей среде.

5. Оптимизация. Многочисленные проблемы в математике, статистике, технике, науке, медицине и экономике могут рассматриваться как проблемы оптимизации. Задачей алгоритма оптимизации является нахождение такого решения, которое удовлетворяет системе ограничений и максимизирует или

минимизирует целевую функцию. Задача коммивояжера, относящаяся к классу NP-полных, является классическим примером задачи оптимизации.

6. Ассоциативная память. Содержимое ассоциативной памяти или памяти, адресуемой по содержанию, может быть вызвано по частичному входу или искаженному содержанию. Ассоциативная память полезна при создании мультимедийных информационных баз данных. А также, она является основой системы управления обучаемых мобильных роботов.

7. Управление. Рассмотрим динамическую систему, заданную совокупностью $\{u(t), y(t)\}$, где $u(t)$ является входным управляющим воздействием, а $y(t)$ – выходом системы в момент времени t . В системах управления с эталонной моделью целью управления является расчет такого входного воздействия $u(t)$, при котором система следует по желаемой траектории, диктуемой эталонной моделью. Примером является оптимальное управление двигателем.

В общем случае все вышеуказанные задачи, решаемые нейронными сетями, можно свести к двум основным [3]:

1. *распознавание (классификация);*
2. *регрессия.*

Задача классификации заключается в формировании нейронной сетью в процессе обучения гиперповерхности в пространстве признаков, разделяющей признаки на классы. И выходы обученной нейронной сети соответствуют распознанному классу входного вектора (набора признаков).

Задача регрессии заключается в аппроксимации нейронной сетью произвольной нелинейной функции. В этом случае значение функции снимается с выхода нейронной сети, а входами являются аргументы. Существует теорема, доказывающая, что многослойный персептрон может аппроксимировать любую нелинейную функцию от n аргументов с какой угодно заданной точностью.

15.2 Биологический нейрон и формальная модель нейрона Маккалоки и Питтса

Биологический нейрон имеет вид, представленный на рисунке 15.1. В 1943 году Дж. Маккалоки и У. Питт предложили формальную модель биологического нейрона как устройства, имеющего несколько входов (входные синапсы – дендриты), и один выход (выходной синапс – аксон) – рисунок 15.2 [1].

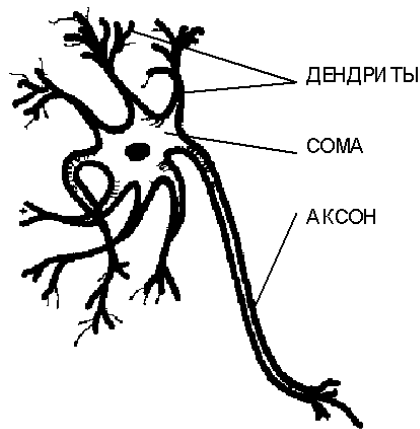


Рисунок 15.1 - Структура биологического нейрона

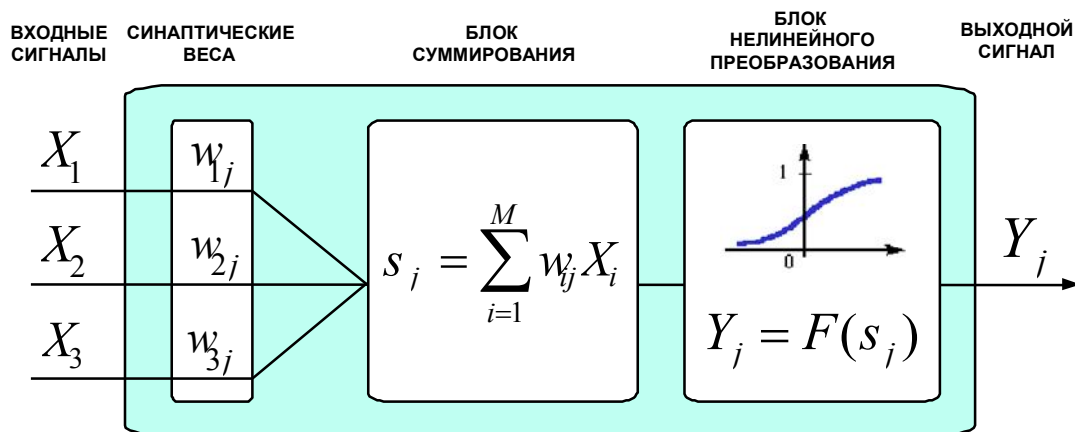


Рисунок 15.2 - Классическая модель нейрона Дж. Маккалоки и У. Питта в обозначениях системной теории информации

Дендриты получают информацию от источников информации (рецепторов) X_i , в качестве которых могут выступать и нейроны. Набор входных сигналов $\{X_i\}$ характеризует объект, его состояние или ситуацию, обрабатываемую нейроном.

Каждому i -му входу j -го нейрона ставится в соответствие некоторый весовой коэффициент w_{ij} , характеризующий степень влияния сигнала с этого входа на аргумент передаточной (активационной) функции, определяющей сигнал Y_j на выходе нейрона. В нейроне происходит взвешенное суммирование входных сигналов, и далее это значение используется как аргумент активационной (передаточной) функции нейрона.

Таким образом, текущее состояние нейрона определяется, как взвешенная сумма его входов [1]:

$$s = \sum_{i=1}^n x_i \cdot w_i, \quad (15.1)$$

а выход нейрона есть функция его состояния [1]:

$$y = f(s). \quad (15.2)$$

15.3 Активационная функция нейрона

Нелинейная функция f в выражении (15.2) называется активационной и может иметь различный вид, как показано на рисунке 15.3.

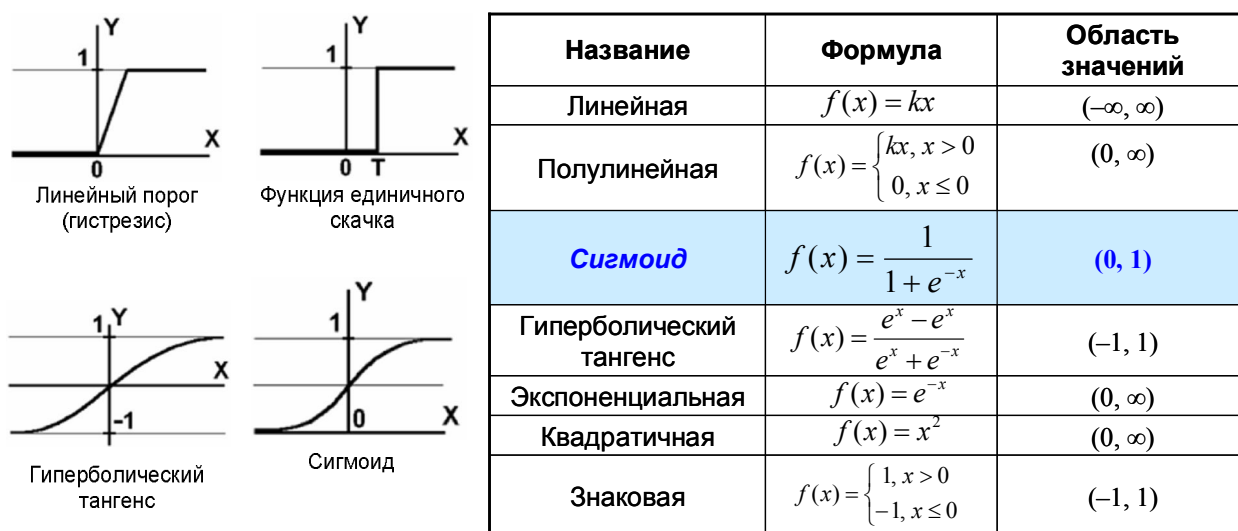


Рисунок 15.3 – Основные функции используемые в виде активационных и их вид

Одной из наиболее распространенных является нелинейная функция с насыщением, так называемая логистическая функция или сигмоид (т.е. функция S-образного вида) [1]:

$$f(x) = \frac{1}{1 + e^{-\alpha x}} \quad (15.3)$$

При уменьшении α сигмоид становится более пологим, в пределе при $\alpha=0$ вырождаясь в горизонтальную линию на уровне 0.5, при увеличении α сигмоид приближается по внешнему виду к функции единичного скачка с порогом T в точке $x=0$. Из выражения для сигмоида очевидно, что выходное значение нейрона лежит в диапазоне $[0, 1]$.

Одно из ценных свойств сигмоидной функции – простое выражение для ее производной, применение которого будет рассмотрено в дальнейшем.

$$f'(x) = \alpha \cdot f(x) \cdot (1 - f(x)) \quad (15.4)$$

Следует отметить, что сигмоидная функция дифференцируема на всей оси абсцисс, что используется в некоторых алгоритмах обучения. Кроме того, она обладает свойством усиливать слабые сигналы лучше, чем

большие, и предотвращает насыщение от больших сигналов, так как они соответствуют областям аргументов, где сигмоид имеет пологий наклон.

15.4 Простейшая нейронная сеть

В качестве примера простейшей нейронной сети (НС) рассмотрим трехнейронный перцептрон (рисунок 3), то есть такую сеть, нейроны которой имеют активационную функцию в виде единичного скачка.

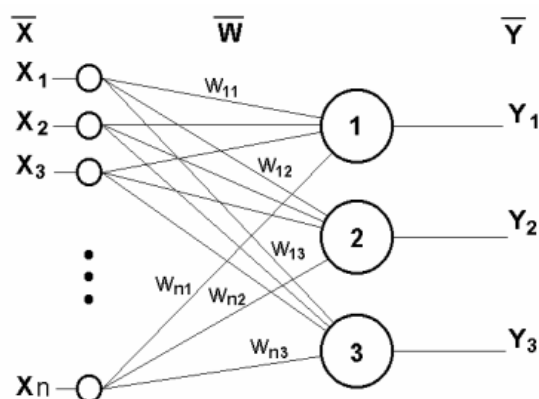


Рисунок 15.4 - Однослойный перцептрон

На n входов НС поступают некие сигналы, проходящие по синапсам на 3 нейрона, образующие единственный слой этой НС и выдающие три выходных сигнала [1]:

$$y_j = f \left[\sum_{i=1}^n x_i \cdot w_{ij} \right], j=1...3 \quad (15.5)$$

Очевидно, что все весовые коэффициенты синапсов одного слоя нейронов можно свести в матрицу W , в которой каждый элемент w_{ij} задает величину i -ой синаптической связи j -ого нейрона. Таким образом, процесс, происходящий в НС, может быть записан в матричной форме:

$$Y = F(XW) \quad (15.6)$$

где X и Y – соответственно входной и выходной сигнальные векторы, $F(V)$ – активационная функция, применяемая поэлементно к компонентам вектора V .

15.5 Однослойная нейронная сеть и перцептрон Розенблатта

Perceptron Розенблатта (F.Rosenblatt, 1957) явился исторически первой искусственной нейронной сетью, способной к перцепции (восприятию) и формированию реакции на воспринятый стимул. Термин «Perceptron» происходит от латинского *perceptio*, что означает восприятие, познание. Русским аналогом этого термина является «Перцептрон». Его автором перцептрон рассматривался не как конкретное техническое вычислительное

устройство, а как модель работы мозга. Современные работы по искусственным нейронным сетям редко преследуют такую цель.

Простейший классический персептрон содержит элементы трех типов (рисунок 15.5), назначение которых в целом соответствует нейрону рефлекторной нейронной сети, рассмотренному выше.

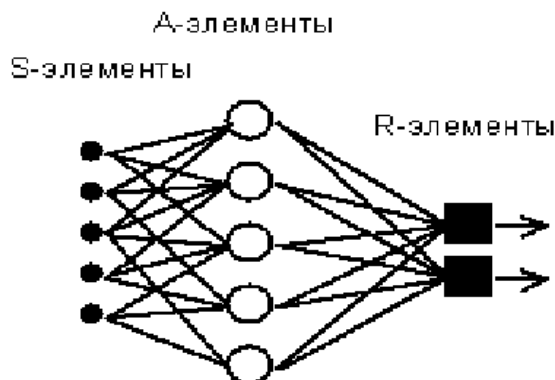


Рисунок 15.5 - Элементарный персептрон Розенблатта

S-элементы – это сенсоры или рецепторы, принимающие двоичные сигналы от внешнего мира. Каждому S-элементу соответствует определенная градация некоторой описательной шкалы.

Далее сигналы поступают в слой ассоциативных или А-элементов (показана часть связей от S к А-элементам). Только ассоциативные элементы, представляющие собой формальные нейроны, выполняют совместную аддитивную обработку информации, поступающей от ряда S-элементов с учетом изменяемых весов связей (рисунок 15.5). Каждому А-элементу соответствует определенная градация некоторой классификационной шкалы [1].

R-элементы с фиксированными весами формируют сигнал реакции персептрона на входной стимул. R-элементы обобщают информацию о реакциях нейронов на входной объект, например, могут выдавать сигнал об идентификации данного объекта, как относящегося к некоторому классу только в том случае, если все нейроны, соответствующие этому классу выдадут результат именно о такой идентификации объекта. Это означает, что в R-элементах может использоваться мультипликативная функция от выходных сигналов нейронов. R-элементы также, как и А-элементы, соответствует определенным градациям классификационных шкал [1].

Розенблатт считал такую нейронную сеть трехслойной, однако по современной терминологии, представленная сеть является однослойной, так как имеет только один слой нейропроцессорных элементов.

Если бы R-элементы в персептроне Розенблатта были тождественными по функциям А-элементам, то нейронная сеть классического персептрона

была бы двухслойной. Тогда бы А-элементы выступали для R-элементов в роли S-элементов.

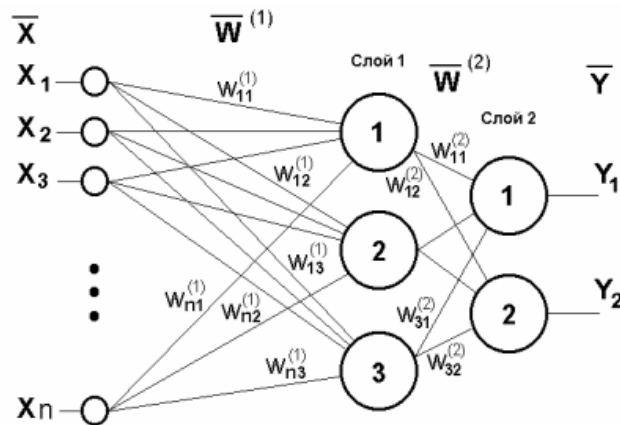


Рисунок 15.6 - Двухслойный персептрон

Однослойный персептрон характеризуется матрицей синаптических связей $||W||$ от S- к А-элементам. Элемент матрицы отвечает связи, ведущей от i -го S-элемента (строки) к j -му А-элементу (столбцы). Эта матрица очень напоминает матрицы абсолютных частот и информативностей, формируемые в семантической информационной модели, основанной на системной теории информации.

С точки зрения современной нейроинформатики однослойный персептрон представляет в основном чисто исторический интерес, вместе с тем на его примере могут быть изучены основные понятия и простые алгоритмы обучения нейронных сетей.

15.6 Машинное обучение нейронной сети на примерах

15.6.1 Обучение на примерах

Обучение классической нейронной сети состоит в подстройке весовых коэффициентов каждого нейрона.

Пусть имеется набор пар векторов $\{x^\alpha, y^\alpha\}$, $\alpha = 1..p$, называемый обучающей выборкой, состоящей из p объектов.

Вектор $\{x^\alpha\}$ характеризует систему признаков конкретного объекта α обучающей выборки, зафиксированную S-элементами.

Вектор $\{y^\alpha\}$ характеризует картину возбуждения нейронов при предъявлении нейронной сети конкретного объекта α обучающей выборки [1]:

$$x_i^\alpha = \begin{cases} 1, & \text{если у объекта } \alpha \text{ наблюдается } i - \text{й признак,} \\ 0, & \text{если у объекта } \alpha \text{ } i - \text{й признак не наблюдается;} \end{cases}$$

$$y_j^\alpha = \begin{cases} 1, & \text{если при предъявлении объекта } \alpha \text{ активизируется } j - \text{й нейрон,} \\ 0, & \text{если при предъявлении объекта } \alpha \text{ } j - \text{й нейрон не активизируется;} \end{cases}$$

Будем называть нейронную сеть обученной на данной обучающей выборке, если при подаче на вход сети вектора $\{x^\alpha\}$ на выходе всегда получается соответствующий вектор $\{y^\alpha\}$ т.е. каждому набору признаков соответствуют определенные классы.

Ф.Розенблаттом предложен итерационный алгоритм обучения из 4-х шагов, который состоит в подстройке матрицы весов, последовательно уменьшающей ошибку в выходных векторах [1]:

- Шаг 0: Начальные значения весов всех нейронов полагаются случайными.
- Шаг 1: Сети предъявляется входной образ x^α , в результате формируется выходной образ.
- Шаг 2: Вычисляется вектор ошибки, делаемой сетью на выходе.
- Шаг 3: Вектора весовых коэффициентов корректируются таким образом, что величина корректировки пропорциональна ошибке на выходе и равна нулю если ошибка равна нулю:
 - модифицируются только компоненты матрицы весов, отвечающие ненулевым значениям входов;
 - знак приращения веса соответствует знаку ошибки, т.е. положительная ошибка (значение выхода меньше требуемого) приводит к усилению связи;
 - обучение каждого нейрона происходит независимо от обучения остальных нейронов, что соответствует важному с биологической точки зрения, принципу локальности обучения.
- Шаг 4: Шаги 1-3 повторяются для всех обучающих векторов. Один цикл последовательного предъявления всей выборки называется эпохой. Обучение завершается по истечении нескольких эпох, если выполняется по крайней мере одно из условий:
 - когда итерации сойдутся, т.е. вектор весов перестает изменяться;
 - когда полная просуммированная по всем векторам абсолютная ошибка станет меньше некоторого малого

значения.

Данный метод обучения был назван Ф.Розенблаттом «методом коррекции с обратной передачей сигнала ошибки». Имеется в виду передача сигнала ошибка от выхода сети на ее вход, где и определяются, и используются весовые коэффициенты. Позднее этот алгоритм называли « α -правилом».

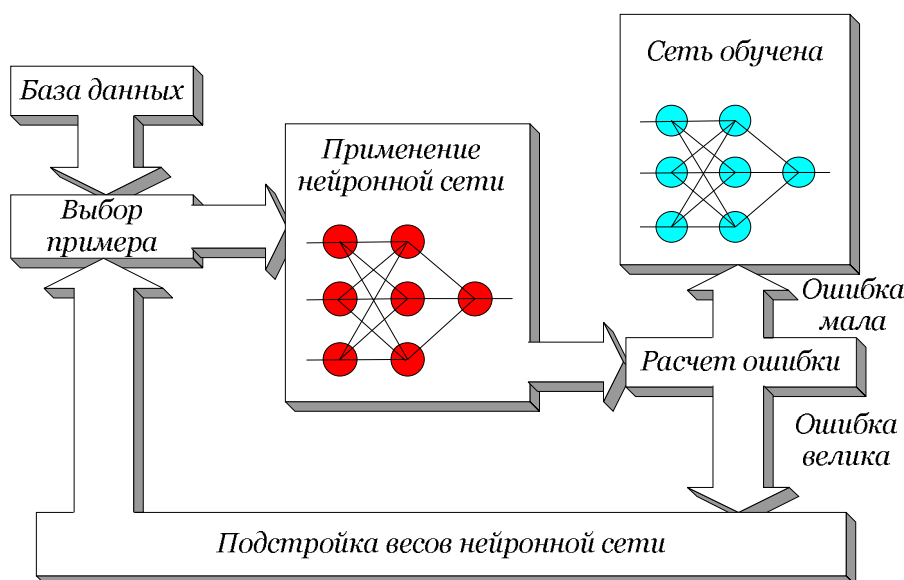


Рисунок 15.6 - Обучение классической нейронной сети

Данный алгоритм относится к широкому классу алгоритмов обучения с учителем, т.к. в нем считаются известными не только входные вектора, но и значения выходных векторов, т.е. имеется учитель, способный оценить правильность ответа ученика, причем в качестве последнего выступает нейронная сеть.

Розенблаттом доказана «Теорема о сходимости обучения» по α -правилу. Эта теорема говорит о том, что персептрон способен обучиться любому обучающему набору, который он способен представить. Но она ничего не говорит о том, какие именно обучающие наборы он способен представить.

Эти алгоритмы обучения с учителем можно интерпретировать как итерационное изменение положения разделяющей гиперплоскости при обнаружении примеров неправильной классификации (рисунок 15.7).

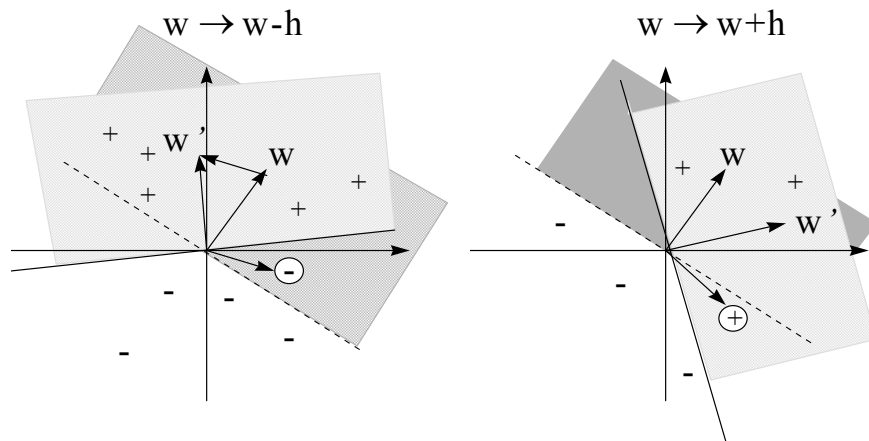


Рисунок 15.7 – Интерпретация обучения НС как итерационного изменение положения разделяющей гиперплоскости при обнаружении примеров неправильной классификации

Применяемые в настоящее время алгоритмы обучения можно интерпретировать как методы оптимизации. Для того их построения вводится функция оценивания (cost function), характеризующая величину невязки между реальным (y) и желаемым (t) выходами сети (персептрона):

$$E(W) = \frac{1}{2L} \sum_{l=1}^L (y^l - t^l)^2 = \frac{1}{2L} \sum_{l=1}^L (Wx^l - t^l)^2,$$

и используется тот или иной метод поиска ее минимума.

Обычно применяется метод градиентного спуска, при котором изменение весов связей можно записать в виде

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}}.$$

При выборе функции оценивания в квадратичной форме она соответствует некоторому параболоиду в многомерном пространстве весов связей. Но, хотя такая поверхность имеет единственный минимум, он как правило расположен на дне длинного «оврага». При этом обучение становится неприятно медленным, а вблизи дна оврага возникают осцилляции (рисунок 15.8). Приходится вводить специальные меры для ускорения процесса сходимости (например, метод моментов) - рисунок 15.9.

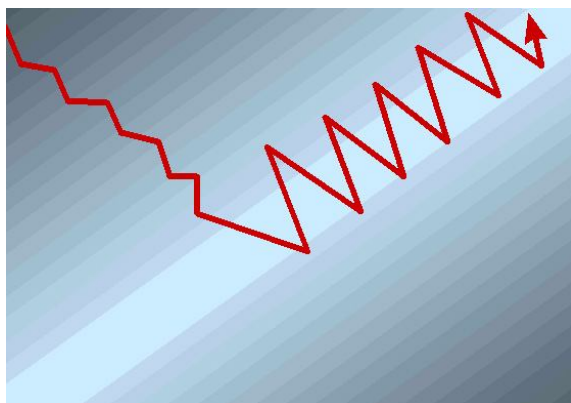


Рисунок 15.8 – Фазовая траектория функции оценивания при методе градиентного спуска

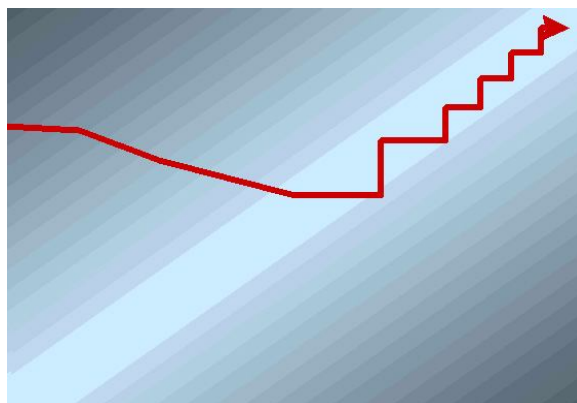


Рисунок 15.9 – Фазовая траектория функции оценивания при использовании мер ускорения сходимости

15.6.2 Решение задач классификации и линейного разделения множеств

При прямоугольной передаточной функции (15.7) каждый нейрон представляет собой пороговый элемент, который может находиться только в одном из двух состояний:

- возбужденном (активном), если взвешенная сумма входных сигналов больше некоторого порогового значения;
- заторможенном (пассивном), если взвешенная сумма входных сигналов меньше некоторого порогового значения.

$$Y_j = \begin{cases} 1, & \sum_{i=1}^M W_{ij} x_i > \theta \\ 0, & \sum_{i=1}^M W_{ij} x_i \leq \theta \end{cases} \quad (15.7)$$

Следовательно, при заданных значениях весов и порогов, каждый нейрон имеет единственное определенное значение выходной активности для каждого возможного вектора входов. При этих условиях множество входных векторов, при которых нейрон активен ($Y=1$), отделено от множества векторов, на которых нейрон пассивен ($Y=0$) гиперплоскостью (15.8).

$$\sum_{i=1}^M W_{ij} x_i - \theta = 0. \quad (15.8)$$

Следовательно, нейрон способен отделить только такие два множества векторов входов, для которых существует гиперплоскость, отделяющая одно множество от другого (рисунок 15.10). Такие множества называют линейно разделимыми.

Необходимо отметить, что линейно-разделимые множества составляют лишь очень незначительную часть всех множеств. Поэтому данное ограничение персептрона является принципиальным. Оно было преодолено лишь в 80-х годах путем введения нескольких слоев нейронов в сетях Хопфилда и нео-когнитроне Фукушимы.

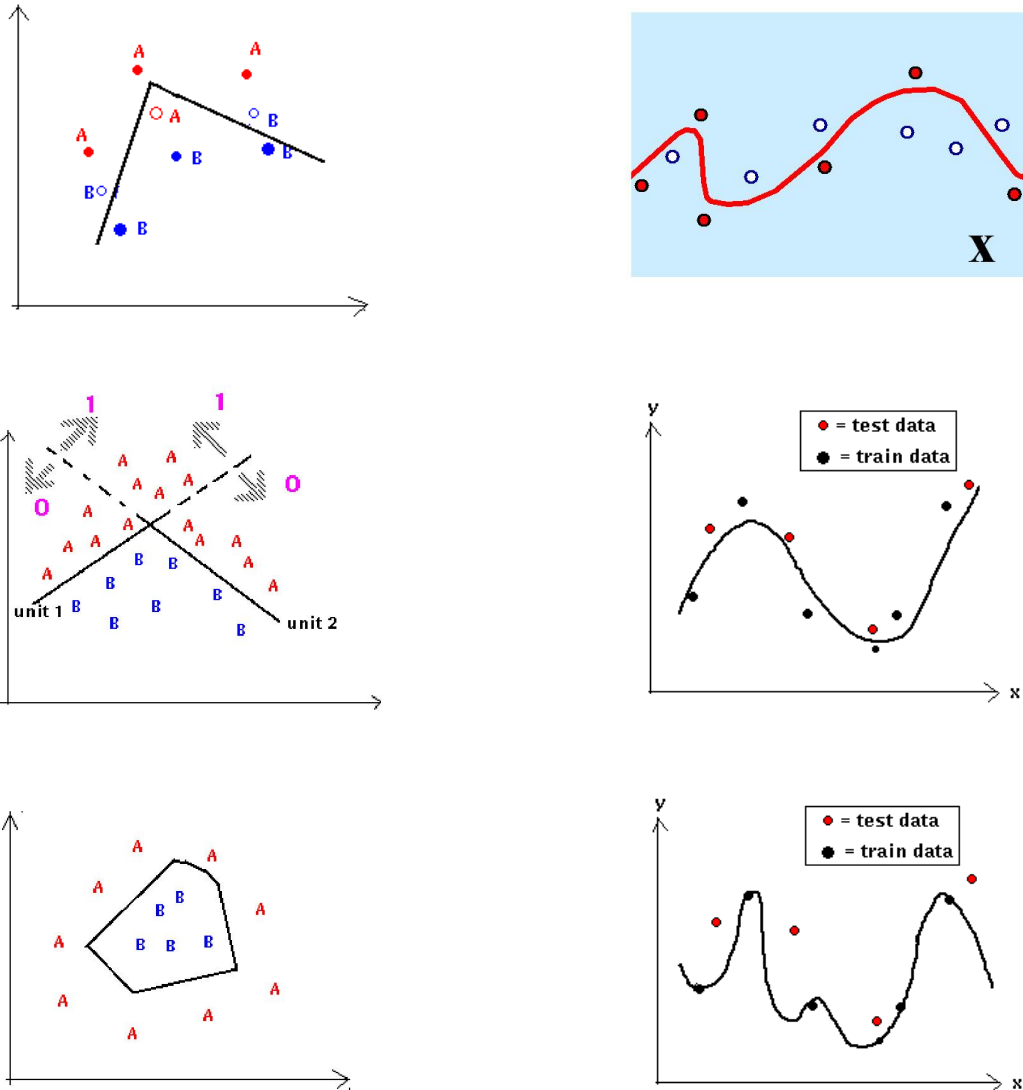


Рисунок 15.10 – Задача классификации и разделения множеств решаемая НС

15.6.3 Проблемы обучения нейронной сети

В завершении остановимся на некоторых проблемах, которые остались нерешенными после работ Ф.Розенблатта:

1. Возможно ли обнаружить линейную разделимость классов до обучения сети?
2. Как определить скорость обучения, т.е. количество итераций, необходимых для достижения заданного качества обучения?

3. Как влияют на результаты обучения последовательность предъявления образов и их количество?
4. Имеет ли алгоритм обратного распространения ошибки преимущества перед простым перебором весов?
5. Каким будет качество обучения, если обучающая выборка содержит не все возможные на практике пары векторов и какими будут ответы персептрона на новые вектора, отсутствующие в обучающей выборке?

Особенно важным представляется последний вопрос, т.к. индивидуальный опыт принципиально всегда не является полным.

15.6.4 Пример решения задачи нейроном

Представим себе, что необходимо решать задачу определения пола студентов по их внешне наблюдаемым признакам.

Есть, конечно, и более надежные способы, но мы их рассматривать не будем, т.к. они требуют дополнительных затрат для получения исходной информации и превращают задачу в тривиальную.

Поэтому будем рассматривать такие описательные шкалы и градации:

1. Длина волос: длинные, средние, короткие.
2. Наличие брюк: да, нет.
3. Использование духов или одеколона: да, нет.

Составим таблицу для определения весовых коэффициентов (таблица 15.1). Пусть столбцы этой таблицы соответствуют состояниям нейрона, а строки – дендритам, соединенным с соответствующими органами восприятия, которые способны устанавливать наличие или отсутствие соответствующего признака.

Таблица 15.1 – Определение весовых коэффициентов нейронов непосредственно на основе эмпирических данных

Описательные шкалы и градации	Классификационные шкалы и градации	
	Юноши	Девушки
Длина волос:		
– длинные	5	15
– средние	10	10
– короткие	15	5
Наличие брюк:		
– да;	30	10
– нет	0	20
Использование духов или одеколона:		
– да;	5	20
– нет	25	10

Тогда один из простейших способов определить значения весовых коэффициентов на дендритах будет заключаться в том, чтобы на пересечениях строк и столбцов просто проставить суммарное количество студентов в обучающей выборке, обладающих данным признаком.

Если нейрон должен выдавать высокий выходной сигнал, когда на входе ему предъявляется юноша и низкий – когда девушка, то весовые коэффициенты на дендритах берутся из столбца: "Юноши". И наоборот, если нейрон должен выдавать высокий выходной сигнал, когда на входе ему предъявляется девушка и низкий – когда юноша, то весовые коэффициенты на дендритах берутся из столбца: "Девушки".

Можно представить себе сеть из двух нейронов, в которой весовые коэффициенты на дендритах взяты из столбцов: "Юноши" и "Девушки".

Большее количество нейронов для решения данной задачи будет избыточным. Его имеет смысл использовать в том случае, когда мы хотим повысить надежность идентификации объектов нейронной сетью и различные сходные по смыслу нейроны будут использовать независимые друг от друга рецепторы.

Например, если мы не только видим идентифицируемый объект, но можем его и обонять, и ощупывать, то это повышает надежность его идентификации. В этом состоит общепринятый в физике критерий реальности – принцип наблюдаемости, согласно которому объективное существование установлено для тех объектов и явлений, существование которых установлено несколькими, по крайней мере, двумя, независимыми способами.

В общем случае в нейронной сети каждому классу (градации классификационной шкалы) будет соответствовать один нейрон и объект, признаки которого будут измерены рецепторами на входе нейронной сети, будет идентифицирован сетью как класс, соответствующий нейрону с максимальным уровнем сигнала на выходе.

Психологические тесты обычно позволяют тестировать респондента сразу по нескольким шкалам. Очевидно, нейронные сети, реализующие эти тесты, будут иметь как минимум столько нейронов, сколько шкал в психологическом тесте.

15.7 Классификация нейронных сетей

В настоящее время существует большое разнообразие моделей нейронных сетей. Приведем их классификацию в соответствии с работой [3].

Нейронные сети различают [3]:

1. *по структуре сети (связей между нейронами),*
2. *по особенностям модели нейрона,*

3. по особенностям обучения сети.

1. По структуре нейронные сети можно разделить на [3]:

1.1 по признаку связности:

- *неполносвязные (или слоистые),*
- *полносвязные,*

1.2 по виду связей:

- *со случайными связями,*
- *с регулярными связями,*
- *с обратными (рекуррентными) связями;*

1.3. по симметрии связей:

- *с симметричными связями,*
- *с несимметричными связями.*

Неполносвязные нейронные сети описываются неполносвязным ориентированным графом. Наиболее распространенным типом таких нейронных сетей являются персептроны: однослойные (простейшие персептроны) и многослойные, с прямыми, перекрестными и обратными связями.

В нейронных сетях с прямыми связями нейроны j -ого слоя по входам могут соединяться только с нейронами i -ых слоев, где $j > i$, т.е. с нейронами нижележащих слоев.

В нейронных сетях с перекрестными связями допускаются связи внутри одного слоя, т.е. выше приведенное неравенство заменяется на $j \geq i$. В нейронных сетях с обратными связями используются и связи j -ого слоя по входам с i -ым при $j < i$. Кроме того, по виду связей различают персептроны с регулярными и случайными связями. Нейронные сети с обратными связями называют рекуррентными.

2. По особенностям модели нейрона нейронные сети можно разделить на [3]:

2.1 по используемым на входах и выходах сигналам

- *аналоговые,*
- *бинарные;*

2.2 по изменяемым показателям состояния нейрона:

- *веса синапсов нейронов,*
- *веса синапсов и пороги нейронов,*
- *установление новых связей между нейронами;*

2.3 по принципу моделирования времени:

- *сети с непрерывным временем,*
- *сети с дискретным временем.*

Для программной реализации применяется, как правило, дискретное время.

3. По особенностям обучения нейронных сетей различают [3]:

3.1 по наличию учителя:

- *с учителем (supervised neural networks),*
- *без учителя (nonsupervised).*

3.2 по способу обучения:

- *обучение по входам,*
- *обучение по выходам;*

3.3 по способу предъявления примеров:

- *предъявление одиночных примеров,*
- *предъявления «страницы» (множества) примеров.*

При обучении с учителем предполагается, что есть внешняя среда, которая предоставляет обучающие примеры (значения входов и соответствующие им значения выходов) на этапе обучения или оценивает правильность функционирования нейронной сети и в соответствии со своими критериями меняет состояние нейронной сети или поощряет (наказывает) нейронную сеть, запуская тем самым механизм изменения ее состояния.

Под состоянием нейронной сети, которое может изменяться, обычно понимается:

- *веса синапсов нейронов (карта весов – map) (коннекционистский подход);*
- *веса синапсов и пороги нейронов (обычно в этом случае порог является более легко изменяемым параметром, чем веса синапсов);*
- *установление новых связей между нейронами (свойство биологических нейронов устанавливать новые связи и ликвидировать старые называется пластичностью).*

Кроме того, есть так называемые «растущие» нейронные сети, в которых количество нейронов изменяется в процессе обучения. Алгоритмы обучения таких сетей называются конструктивными.

При обучении по входам обучающий пример представляет собой только вектор входных сигналов, а при обучении по выходам в него входит и вектор выходных сигналов, соответствующий входному вектору.

В случае предъявления одиночных примеров изменение состояния нейронной сети (обучение) происходит после предъявления каждого примера. В случае предъявления «страницы» примеров – на основе анализа сразу их всех.

В основу главы 15 положен материал учебных пособий [1, 3] и работы [9], дополненные материалом работ [10, 11, 12].

16. МНОГОСЛОЙНЫЕ НЕЙРОННЫЕ СЕТИ

16.1 Многослойный персептрон

Каким же образом в многослойных (иерархических) нейронных сетях преодолевается принципиальное ограничение однослойных нейронных сетей, связанное с требованием линейной разделимости классов?

Часто то, что не удастся сделать сразу, вполне возможно сделать по частям. Для этого изменяются задачи, решаемые слоями нейронной сети. Оказывается в 1-м слое не следует пытаться на основе первичных признаков, фиксируемых рецепторами, сразу идентифицировать классы, а нужно лишь сформировать линейно-разделимую систему вторичных признаков, которую уже во 2-м слое связать с классами (рисунок 16.1).

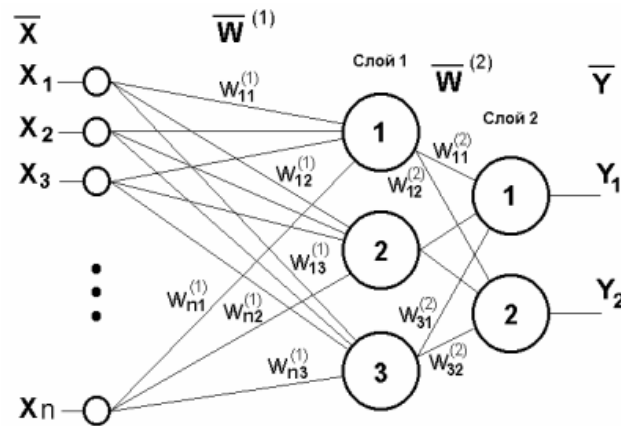


Рисунок 16.1 - Задачи, решаемые слоями двухслойной нейронной сети

В многослойной сети выходные сигналы нейронов предыдущего слоя играют роль входных сигналов для нейронов последующего слоя, т.е. нейроны предыдущего слоя выступают в качестве рецепторов для нейронов последующего слоя.

Связи между смежными слоями нейронов будем называть непосредственными, а связи между слоями, разделенными N промежуточных слоев, будем называть связями N -го уровня опосредованности. Непосредственные связи — это связи 0-го уровня опосредованности. Промежуточные слои нейронов в многослойных сетях называют скрытыми.

Персептрон переводит входной образ, определяющий степени возбуждения рецепторов, в выходной образ, определяемый нейронами самого верхнего уровня, которых обычно, не очень много. Состояния возбуждения нейронов на верхнем уровне иерархии сети характеризуют принадлежность входного образа к тем или иным классам.

Таким образом, **многослойный персептрон** – это обучаемая распознающая система, реализующая корректируемое в процессе обучения линейное решающее правило в пространстве вторичных признаков, которые обычно являются фиксированными случайно выбранными линейными пороговыми функциями от первичных признаков.

При обучении на вход персептрона поочередно подаются сигналы из обучающей выборки, а также указания о классе, к которому следует отнести данный сигнал. Обучение персептрона заключается в коррекции весов при каждой ошибке распознавания. Если персептрон ошибочно отнес сигнал, к некоторому классу, то веса функции, истинного класса увеличиваются, а ошибочного уменьшаются. В случае правильного решения все веса остаются неизменными.

Этот чрезвычайно простой алгоритм обучения обладает замечательным свойством: если существуют значения весов, при которых выборка может быть разделена безошибочно, то при определенных, легко выполнимых условиях эти значения будут найдены за конечное количество итераций.

При идентификации, распознавании, прогнозировании на вход многослойного персептрона поступает сигнал, представляющий собой набор первичных признаков, которые и фиксируются рецепторами. Сначала вычисляются вторичные признаки. Каждому такому вторичному признаку соответствует линейная от первичных признаков. Вторичный признак принимает значение 1, если соответствующая линейная функция превышает порог. В противном случае она принимает значение 0. Затем для каждого из классов вычисляется функция, линейная относительно вторичных признаков. Персептрон вырабатывает решение о принадлежности входного сигнала к тому классу, которому соответствует функция от вторичных параметров, имеющая наибольшее значение.

Показано, что для представления произвольного нелинейного функционального отображения, задаваемого обучающей выборкой, достаточно всего двух слоев нейронов. Однако на практике, в случае сложных функций, использование более чем одного скрытого слоя может давать экономию полного числа нейронов.

16.2 Модель Хопфилда

В модели Хопфилда (J. J. Hopfield, 1982) впервые удалось установить связь между нелинейными динамическими системами и нейронными сетями.

Модель Хопфилда является обобщением модели многослойного персептрона путем добавления в нее следующих двух новых свойств:

1. В нейронной сети все нейроны непосредственно связаны друг с другом: силу связи i -го нейрона с j -м обозначим как W_{ij} .

2. Связи между нейронами симметричны: $W_{ij} = W_{ji}$, сам с собой нейрон не связан $W_{ii} = 0$.

Каждый нейрон может принимать лишь два состояния:

$$x_i = \begin{cases} 1, \\ -1 \end{cases}$$

Изменение состояний возбуждения всех нейронов может происходить либо последовательно, либо одновременно (параллельно), но свойства сети Хопфилда не зависят от типа динамики.

Взаимодействие нейронов сети описывается выражением:

$$E = \frac{1}{2} \sum_{i,j=1}^N w_{ij} x_i x_j,$$

где w_{ij} элемент матрицы взаимодействий W , которая состоит из весовых коэффициентов связей между нейронами. В эту матрицу в процессе обучения записывается M «образов» — N -мерных бинарных векторов: $S_m = (s_{m1}, s_{m2}, \dots, s_{mN})$

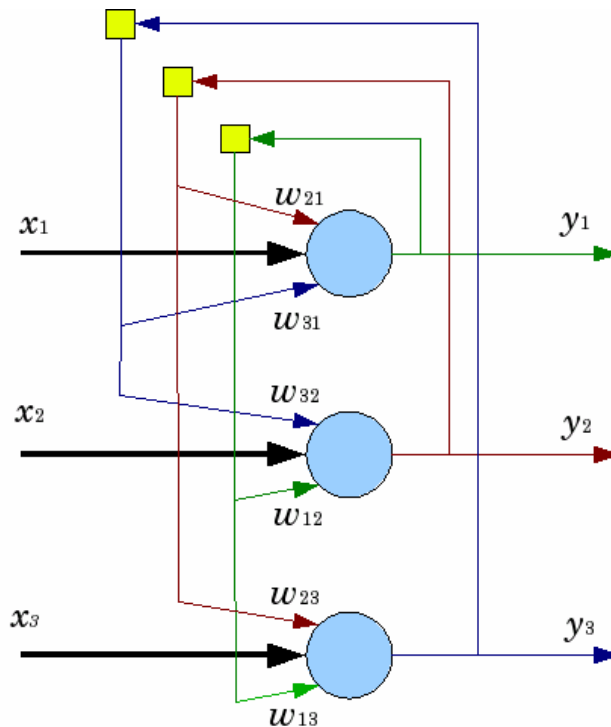


Рисунок 16.2 - Модель Хопфилда

По сути сеть Хопфилда - это **рекуррентная сеть ассоциативной памяти** с динамикой, управляемой системой

$$x(t+1) = G(Wx(t) - \theta).$$

Вместо переменных $\{0, 1\}$ используются переменные $\{-1, 1\}$, и полагается $\theta = 0$, так что динамическая система в компонентах имеет вид:

$$x_i(t+1) = \operatorname{sgn} \left(\sum_{j=1}^n W_{ij} x_j(t) \right), \quad i = 1..n \quad (16.1)$$

где $W_{ij} = n^{-1} \sum_{l=1}^L s_i^l s_j^l$, $s^l = (s_1^l, \dots, s_n^l)^T$ - эталоны памяти, n - число нейронов сети, L - число эталонов.

Емкость памяти $r = L/n$ при больших n удается асимптотически оценить с помощью центральной предельной теоремы:

$$r_{\max} = n / 4 \log n \approx 0.138.$$

Ключавым моментом оказалась идея введения энергии сети (J. Hopfield, 1982):

$$H = -0.5 \cdot \sum_{ij} W_{ij} x_i x_j.$$

Энергия является функцией Ляпунова для динамической системы (16.1): H убывает в силу динамики сети. Кроме того, она соответствует Гамильтониану в статистической механике физических систем, целевой функции в теории оптимизации и функции приспособленности в эволюционной биологии. Векторы памяти, являющиеся устойчивыми аттракторами динамики, располагаются в локальных минимумах «энергетической» поверхности.

Сеть Хопфилда способна распознавать объекты при неполных и зашумленных исходных данных, однако не может этого сделать, если изображение смещено или повернуто относительно его исходного состояния, представленного в обучающей выборке.

При распознавании «работа» сети состоит в релаксации сети из некоторого заданного начального состояния x^0 в одно из состояний устойчивого равновесия. Если состояние x^0 представляет искаженную версию одного из эталонов памяти и точка x^0 фазового пространства динамической системы находится в бассейне притяжения устойчивого аттрактора S^l , то сеть релаксирует в состояние равновесия S^l , что и соответствует «воспроизведению» эталона.

Важной особенностью сети Хопфилда является существование множества дополнительных устойчивых аттракторов динамики помимо $S^1 \dots S^L$ (так называемая «посторонняя» память, или spurious states). Свойства посторонней памяти достаточно хорошо изучены. В частности, имеются строгие результаты, свидетельствующие об экспоненциальном росте числа «посторонних» аттракторов с ростом L .

Стремительное увеличение числа дополнительных аттракторов при возрастании L является основной причиной малой емкости памяти сети Хопфилда. При возрастании L происходит также быстрое снижение качества воспроизведения эталонов в результате сокращения бассейнов притяжения эталонов и уменьшения их устойчивости.

16.3 Когнитрон и неокогнитрон Фукушимы

В целом когнитрон Фукушимы (K.Fukushima, 1975) представляет собой иерархию слоев, последовательно связанных друг с другом, как в персептроне. Однако, при этом есть два существенных отличия:

1. Нейроны образуют не одномерную цепочку, а покрывают плоскость, аналогично слоистому строению зрительной коры человека.
2. Когнитрон состоит из иерархически связанных слоев нейронов двух типов – тормозящих и возбуждающих.

В когнитроне каждый слой реализует свой уровень обобщения информации:

- входные слои чувствительны к отдельным элементарным структурам, например, линиям определенной ориентации или цвета;
- последующие слои реагируют уже на более сложные обобщенные образы;
- в слое наивысшего уровня иерархии активные нейроны определяют результат работы сети – узнавание определенного образа, при этом результатам распознавания соответствуют те нейроны, активность которых оказалась максимальной.

Однако добиться независимости результатов распознавания от размеров и ориентации изображений удалось лишь в неокогнитроне, который был разработан Фукушимой в 1980 году и представляет собой как бы суперпозицию когнитронов, обученных распознаванию объектов различных типов, размеров и ориентации.

16.4 Модель нелокального нейрона

Модель нелокального нейрона: так как сигналы на дендритах различных нейронов вообще говоря коррелируют (или антикоррелируют) друг с другом, то, значения весовых коэффициентов, а значит и выходное значение на аксоне каждого конкретного нейрона вообще говоря не могут быть определены с использованием значений весовых коэффициентов на дендритах только данного конкретного нейрона, а должны учитывать интенсивности сигналов на всей системе дендритов нейронной сети в целом (рисунок 16.3).

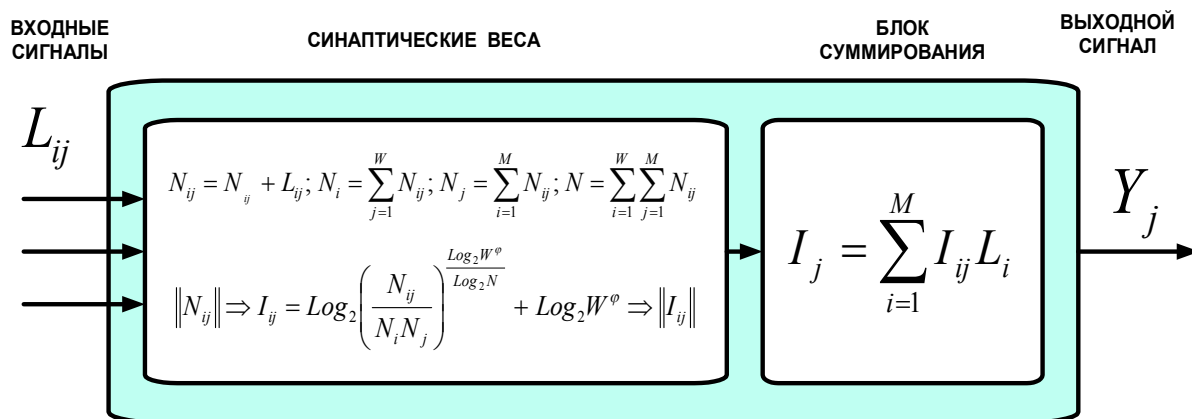


Рисунок 16.3- Модель нелокального нейрона
в обозначениях системной теории информации

За счет учета корреляций входных сигналов (если они фактически присутствуют в структуре данных), т.е. наличия общего самосогласованного информационного поля исходных данных всей нейронной сети (информационное пространство), нелокальные нейроны ведут себя так, как будто связаны с другими нейронами, хотя могут быть и не связаны с ними синаптически по входу и выходу ни прямо, ни опосредованно. Самосогласованность семантического информационного пространства означает, что учет любого одного нового факта в информационной модели вообще говоря приводит к изменению всех весовых коэффициентов всех нейронов, а не только тех, на рецепторе которых обнаружен этот факт и тех, которые непосредственно или опосредованно синаптически с ним связаны.

В традиционной (т.е. локальной) модели нейрона весовые коэффициенты на его дендритах однозначно определяются заданным выходом на его аксоне и никак не зависят от параметров других нейронов, с которыми с нет прямой или опосредованной синаптической связи. Это связано с тем, что в общепринятой энергетической парадигме Хопфилда весовые коэффициенты дендритов имеют смысл интенсивностей входных воздействий. В методе «обратного распространения ошибки» процесс переобучения, т.е. интерактивного перерасчета весовых коэффициентов, начинается с нейрона, состояние которого оказалось ошибочным и захватывает только нейроны, ведущие от рецепторов к данному нейрону.

Корреляции между локальными нейронами обусловлены сочетанием трех основных причин:

- наличием в исходных данных определенной структуры: корреляцией входных сигналов;
- синаптической связью локальных нейронов;
- избыточностью (дублированием) нейронной сети.

16.5 Динамические нейронные сети

Сети из формальных нейронов вполне приемлемы для математического изучения вычислительных особенностей систем из большого числа активных элементов, осуществляющих параллельную обработку информации.

Изучение аспектов функционирования биологических нейронных сетей требует следующего естественного шага – построения моделей нейрона, воспроизводящих импульсный характер нейронной активности. Это привело к серии моделей импульсных («спайковых») нейронов и изучению нейронных сетей из «спайковых» нейронов с импульсным взаимодействием.

Далее, экспериментально было обнаружено, что при обработке информации целый ряд структур мозга при обработке информации используют в качестве «рабочих инструментов» колебательную активность, синхронизацию и резонанс. Примерами таких структур являются обонятельная система (обонятельная луковица и кора), слуховая система (улитка уха и слуховая кора), зрительная система (сетчатка и зрительная кора). Это стимулировало моделирование осцилляторных нейронных сетей. Активным элементом таких сетей является нейронный осциллятор, образованный парой нейронов, связанных возбуждающей и тормозной связями.

Были также построены модели искусственных осцилляторных сетей ассоциативной памяти, активными элементами которых являются различные модели осцилляторов. Эталоны памяти в таких сетях являются не состояния устойчивого равновесия, а предельные циклы – состояния устойчивых синхронизованных колебаний осцилляторной сети. Воспроизведение эталонов осуществляется посредством синхронизации. Были даже предложены теоретические модели оптических нейрокомпьютеров, построенных на осцилляторах, связанных через галаграфическую среду.

Работа построенных сетей порождает динамические методы обработки информации, основанные на синхронизации ансамблей сетевых осцилляторов.

16.6 Проблемы развития нейронных сетей

К основным проблемам нейронных сетей можно отнести следующее.

1. «Проблема интерпретируемости весовых коэффициентов». Сложность содержательной интерпретации смысла интенсивности входных сигналов и весовых коэффициентов.
2. «Проблема интерпретируемости передаточной функции». Сложность содержательной интерпретации и обоснования аддитивности аргумента и вида активационной (передаточной) функции нейрона (Проблемы

интерпретируемости приводят к снижению ценности полученных результатов работы сети ввиду того что в реальных нейронных сетях, реализуемых на базе известных программных пакетов, количество нейронов чаще всего составляет единицы и десятки).

3. «Комбинаторный взрыв», возникающий при определении структуры связей нейронов, подборе весовых коэффициентов и передаточных функций («проблема размерности»). Проблема размерности приводит – к очень жестким ограничениям на количество выходных нейронов в сети, на количество рецепторов и на сложность структуры взаимосвязей нейронов с сети. Достаточно сказать, что количество выходных нейронов в реальных нейронных сетях, реализуемых на базе известных программных пакетов, обычно не превышает несколько сотен.
4. «Проблема линейной разделимости», возникающая потому, что возбуждение нейронов принимают лишь булевы значения 0 или 1. Проблема линейной разделимости приводит к необходимости применения многослойных нейронных сетей для реализации тех приложений, которые вполне могли бы поддерживаться сетями с значительно меньшим количеством слоев (вплоть до однослойных), если бы значения возбуждения нейронов были не дискретными булевыми значениями, а континуальными значениями, нормированными в определенном диапазоне.

Перечисленные проблемы предлагается решить путем использования модели нелокального нейрона, обеспечивающего построение нейронных сетей прямого счета.

16.7 Нейрокомпьютеры, нейропроцессоры, нейропакеты

Некоторые характерные черты нейрокомпьютера, который есть надежда создать в будущем, можно указать уже теперь. Он будет:

- состоять из сетей, содержащих большое число параллельно работающих активных элементов (до $10^6 - 10^8$),
- использовать ассоциативную обработку информации (храня в памяти наборы эталонов и сравнивая с ними поступающую информацию),
- использовать обучение вместо программирования (адаптивное управление параметрами нейронов» и архитектурой связей в сетях).

Пока же созданы отдельные специализированные «нейрочипы» (нейропроцессоры).

Сегодня программная эмуляция (компьютерное моделирование) пока является основным способом моделирования нейронных сетей. Имеется

целый ряд нейропакетов, созданных для моделирования разнообразных нейросетей. Существуют универсальные и специализированные нейропакеты. Они предоставляют пользователю следующие возможности:

- выбор типа нейросети из набора стандартных;
- синтеза архитектуры сети;
- использования стандартных алгоритмов обучения;
- включения нестандартных алгоритмов обучения (с выбором нестандартных критериев обучения);
- использования различных вариантов визуализации результатов в процессе обучения и работы сети;
- включения программных модулей, написанных пользователем (реализовано лишь в некоторых нейропакетах).

В основу главы 16 положен материал учебных пособий [1, 3] и работы [9], дополненные материалом работ [10, 11, 12].

17. ТЕОРИЯ НЕЧЕТКИХ МНОЖЕСТВ

17.1 Понятие нечеткого множества

В классической теории множеств непустое подмножество A из универсального множества X однозначно определяется характеристическим функционалом

$$I_A(x) = \begin{cases} 1, & \text{если } x \in A \\ 0, & \text{если } x \notin A \end{cases}, \quad (17.1)$$

т.е. подмножество A определяется как совокупность объектов, имеющих некоторое общее свойство, наличие или отсутствие которого у любого элемента x задается характеристическим функционалом. Причем относительно природы объекта не делается никаких предположений.

Задание некоторого множества в этом случае эквивалентно заданию его характеристического функционала, поэтому все операции над множествами можно выразить через действия над их характеристическими функционалами.

Основные операции объединения, пересечения и разности двух подмножеств A и B из X с характеристическими функционалами $I_A(x) < I_B(x)$ соответственно определяются следующим образом для каждого $x \in X$:

$$\begin{aligned} I_{A \cup B}(x) &= I_A(x) + I_B(x) - I_A(x) I_B(x); \\ I_{A \cap B}(x) &= I_A(x) I_B(x); \\ I_{A \setminus B}(x) &= I_A(x) - I_{A \cap B}(x) = I_A(x)(1 - I_B(x)). \end{aligned} \quad (17.2)$$

Операции объединения и пересечения могут быть записаны также в несколько ином виде:

$$\begin{aligned} I_{A \cup B}(x) &= \max(I_A(x), I_B(x)); \\ I_{A \cap B}(x) &= \min(I_A(x), I_B(x)). \end{aligned} \quad (17.3)$$

Однако такие понятия, как множество «больших» или «малых величин», уже не являются множествами в классическом смысле, так как не определены границы их степеней *малости*, которые позволили бы провести классификационную процедуру (17.1) и *четко* отнести каждый объект к определенному классу. Большинство классов реальных объектов и процессов относятся именно к такому нечетко определенному типу. Поэтому возникает необходимость введения понятия о нечетком подмножестве как о классе с непрерывной градацией степеней принадлежности.

Для нечеткого подмножества, являющегося расширением понятия множества в классическом смысле, на пространстве объектов $X = \{x\}$ вводится уже не функционал вида (17.1), а характеристическая функция,

задающая для всех элементов степень наличия у них некоторого свойства, по которому они относятся к подмножеству A . Эта характеристическая функция для нечеткого множества традиционно носит название **функции принадлежности**.

Нечеткое подмножество A множества X характеризуется **функцией принадлежности** $\mu_A: X \rightarrow [0, 1]$, которая ставит в соответствие каждому элементу $x \in X$ число $\mu_A(x)$ из интервала $[0, 1]$, характеризующее степень принадлежности элемента x подмножеству A . Причем 0 и 1 представляют собой соответственно низшую и высшую степень принадлежности элемента к определенному подмножеству.

Точкой перехода A называется элемент x множества X , для которого $\mu_A(x)=0,5$.

Если в классической теории множеств понятие характеристического функционала играет второстепенную роль, то для нечетких множеств функция принадлежности становится единственно возможным средством их описания. С формальной точки зрения нет необходимости различать нечеткое множество и его функцию принадлежности. В этом смысле теория нечетких множеств (ТНМ) можно рассматривать как теорию функций специального вида - обобщенных характеристических функций.

Численное значение функции принадлежности характеризует степень принадлежности элемента некоторому нечеткому множеству, являющемуся в выражении естественного языка некоторой, как правило, элементарной характеристикой явления (степени загрязненности участка, степени эффективности режима, степени надежности продукции и т.д.).

Заде Л. ввел понятие **лингвистической переменной**, значениями которой являются слова и или предложения естественного языка, которые описываются нечеткими значениями.

Например, лингвистическая переменная «ВОЗРАСТ» принимает нечеткие значения «молодой», «не молодой», «старый», «не очень старый» и т.д.

Пример 17.1. Нечеткое подмножество, обозначаемое термином «старый», можно определить функцией принадлежности приведенной на рисунке 17.1. В этом примере носителем нечеткого множества «старый» является интервал $[50, 100]$, а точкой перехода значение $x = 55$.

$$\mu_A(x) = \begin{cases} 0, & \text{при } 0 \leq x < 50, \\ (1 + (\frac{x-50}{5})^2)^{-1}, & \text{при } 50 \leq x \leq 100 \end{cases}$$

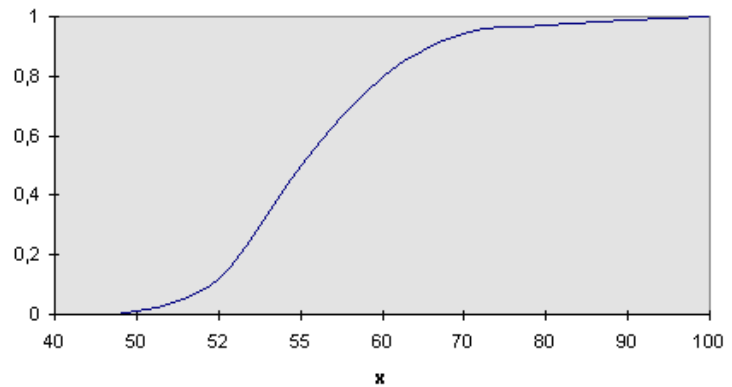


Рисунок 17.1 - Функция принадлежности для значений термина *старый*

Другими примерами нечетких ситуаций могут служить модели и принятие решения для процессов разработки газовых месторождений, износа оборудования и т.д. Все эти процессы протекают монотонно и трудно бывает выделить четкую границу между допустимыми и недопустимыми состояниями (например, до которой можно считать трубу чистой и за которой ее состояние становится загрязненным).

Существуют достаточно четкие области, где классификация, а соответственно и решения, будут достаточно однозначными - область, близкая к идеальному состоянию трубы, и область, близкая к полному загрязнению (закупорка). Наиболее сложно принимать решение, когда состояние системы приходится на переходный режим между этими двумя крайними состояниями, и когда этот переход не скачкообразен, а непрерывен. Такая ситуация очень типична для реальных систем, и многие понятия естественного языка не могут быть формализованы с помощью классических математических понятий, так как граница между двумя классифицируемыми состояниями (например, «чистый» - «загрязненный») является нечеткой, размытой.

Таким образом, основное предположение состоит в том, что нечеткое множество, несмотря на расплывчатость его границ, может быть точно определено путем сопоставления каждому элементу x числа, лежащего между 0 и 1, которое представляет степень его принадлежности к A .

Носителем нечеткого подмножества A называется четкое подмножество из X , на котором $\mu_A(x) > 0$.

$$\sigma(x) = \{x / \mu_A(x) > 0\}. \quad (17.4)$$

Для практических приложений носители нечетких множеств всегда ограничены. Так, носителем нечеткого множества допустимых режимов для системы может служить четкое подмножество (интервал), для которого степень допустимости не равна нулю (рис. 17.2).

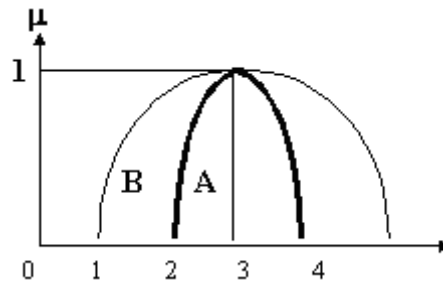


Рисунок 17.2 - Понятие носителя нечеткого множества (выделен жирной чертой)

Высотой d нечеткого множества A называется максимальное значение функции принадлежности этого множества.

$$d = \max_{x \in X} \mu_A(x),$$

Если $d=1$, то нечеткое множество называется **нормальным**.

Одноточечным нечетким множеством называется множество, носитель которого состоит из единственной точки. Нечеткое множество A иногда рассматривают как объединение составляющих его одноточечных множеств:

$$A = \mu_1/x_1 + \dots + \mu_n/x_n,$$

где: знак «+» обозначает операцию объединения; μ_i - степень принадлежности x_i к множеству A .

F -множествами называют совокупность всех нечетких подмножеств $F(X)$ произвольного (базового) множества X , а их функции принадлежности F - функциями. Как правило, под μ_x понимают сужение функции принадлежности со всего X на $\sigma(A)$.

Для обозначения F -множеств используют запись вида:

$$A = \langle \mu_A, \sigma(A) \rangle.$$

Например,

$$A = \langle e^{-(x-a)^2}, [c, d] \rangle \text{ или } B = \langle \sin x, [0, \pi] \rangle.$$

Кроме того, при необходимости данная форма обозначения может применяться и для обычных (четких) подмножеств из X .

17.2 Возможности применения теории нечетких множеств для описания различных видов неопределенности

Для реальных сложных систем характерно наличие одновременно разнородной информации:

1. точечных замеров и значений параметров;

2. допустимых интервалов их изменения;
3. статистических законов распределения для отдельных величин;
4. лингвистических критериев и ограничений, полученных от специалистов-экспертов и т.д.

Наличие в сложной многоуровневой иерархической системе управления одновременно различных видов неопределенности делает необходимым использование для принятия решений теории нечетких множеств, которая позволяет адекватно учесть имеющиеся виды неопределенности.

Разработанные в настоящее время количественные методы принятия решений (такие, как максимизация ожидаемой полезности, минимаксная теория, методы максимального правдоподобия, теория игр, анализ «затраты – эффективность» и другие) помогают выбирать наилучшие из множества возможных решений лишь в условиях одного конкретного вида неопределенности или в условиях полной определенности. К тому же, большая часть существующих методов для облегчения количественного исследования в рамках конкретных задач принятия решений базируется на крайне упрощенных моделях действительности и излишне жестких ограничениях, что уменьшает ценность результатов исследований и часто приводит к неверным решениям.

Применение для оперирования с неопределенными величинами аппарата теории вероятности приводит к тому, что фактически неопределенность, независимо от ее природы, отождествляется со случайностью, между тем как основным источником неопределенности во многих процессах принятия решений является нечеткость или расплывчатость (fuzziness).

В отличие от случайности, которая связана с неопределенностью, касающейся принадлежности или непринадлежности некоторого объекта к нерасплывчатому множеству, понятие «нечеткость» относится к классам, в которых могут быть различные градации степени принадлежности, промежуточные между полной принадлежностью и непринадлежностью объектов к данному классу.

При наиболее абстрактном подходе к системе критерий функционирования этой системы на языке теории нечетких множеств можно представить в форме максимизации степени допустимости и эффективности принимаемых решений. Поэтому в качестве подмножества выбрано подмножество допустимых и эффективных значений параметра x . Подмножество эффективных значений параметра x является нечетким для реальных систем, так как нельзя сказать, что лишь одно значение, например $x_2 = 4$, является эффективным, а все остальные значения x неэффективны (рис. 17.3), т.е. $\mu_A(4) = 1$ и $\mu_A(x) = 0$ для $x \neq 4$.

Реально такой грани нет, так как незначительное изменение x ведет лишь к небольшому изменению $\mu_A(x)$, поэтому функции принадлежности вида (1, 2, 3) больше соответствуют действительности. Так, применение выражения вида «должно быть близко к x_2 », которое не является точно сформулированной целью, может быть смоделировано нечетким подмножеством с функцией принадлежности с графиком 1 на рисунке 17.3.

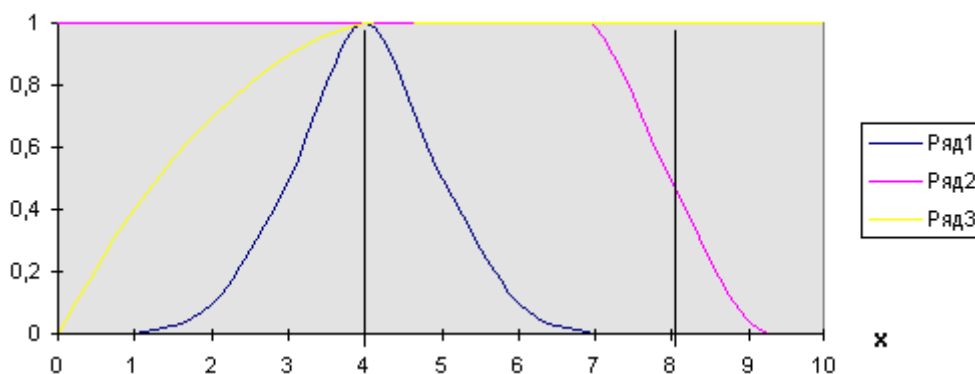


Рисунок 17.3 - Функции принадлежности для четких и нечетких целей и ограничений.

*Ограничения на допустимость режима также могут быть **четкими и нечеткими** (2). Применение нечетких («мягких») ограничений значительно расширяет возможность контроля и управления и делает их адекватными реальной обстановке в системе.*

Например, можно жестко задать в системе газодобычи, что точка росы не может быть выше $x_1=8$ и, таким образом, работа системы при $x>x_1$ недопустима.

Так жестко действует система автоматики и для нее режимы $x>x_1$ недопустимы. В действительности же точка росы не является такой резкой гранью (тем более в условиях большой погрешности ее определения), и работа системы в области $x>x_1$ возможна, только приводит к значительному снижению степени допустимости этого режима. Функция принадлежности типа 2 на рисунке 17.3 больше соответствует этим условиям. Степень размытости (нечеткости) функции $\mu_A(x)$ задает жесткость ограничений или целей, т.е. фактически важность данного ограничения или цели для системы и точность их определения.

Во многих задачах контроля и управления сложной системой нет необходимости в получении оптимального четкого решения для каждого момента времени, так как затраты на накопление информации и жесткое устранение невязок в системе могут превышать достигаемый при этом эффект. Чаще всего конкретное содержание задачи требует обеспечения заданного уровня нечеткости решения.

Учет фактора неопределенности при решении задач во многом изменяет методы принятия решения: меняется принцип представления

исходных данных и параметров модели, становятся неоднозначными понятия решения задачи и оптимальности решения.

Наличие неопределенности может быть учтено непосредственно в моделях соответствующего типа с представлением недетерминированных параметров как случайных величин с известными вероятностными характеристиками, как нечетких величин с заданными функциями принадлежности или как интервальных величин с фиксированными интервалами изменения и нахождения решения задачи с помощью методов стохастического, нечеткого или интервального программирования.

Возможно также и прямое построение зоны неопределенности без непосредственного учета характеристик недетерминированных параметров модели. В этом случае решается ряд детерминированных задач и получается некоторый набор вариантов, оптимальных при конкретных значениях случайных (или нечетких) параметров.

Наиболее часто оказывается, что в процессе принятия решений для некоторых переменных или параметров модели могут быть заданы лишь диапазон их изменения (максимально и минимально допустимые значения $\bar{x}$ и $\underline{x}$) и наиболее правдоподобная оценка x^ .*

В целом алгоритмы на базе нечетких множеств хорошо зарекомендовали себя на практике для самого разнообразного круга задач:

- для создания математической модели многослойного оценивания запасов угля в пластах;
- применение нечетких уравнений и элементов нечеткой логики для диагностирования сложных систем - пакет программ Thermix-2D для анализа динамики АЭС;
- при управлении нестационарным процессом движения морских геолого-геофизических комплексов;
- для оценки показателей качества программных средств;
- в системах искусственного интеллекта для управления работой технологического оборудования;
- в задачах контроля и управления системами разработки месторождений, добычи и транспорта газа ;
- поведение диспетчерского персонала лучше всего описывается лингвистическими правилами поведения, а отклонение от принятых алгоритмов (ошибки и плохая работа диспетчеров, неисправности, возникшие помехи) хорошо моделируется с использованием нечетких алгоритмов.

Успешным является и применение теории нечетких множеств в стохастических системах. Это применение связано с тем, что для многих

систем трудно получить точные значения вероятностных характеристик (например, вероятности отказов компонентов).

При использовании нечетких или интервальных моделей становится возможным сравнение точности результатов, полученных для различных моделей. Анализируя интервалы или функции принадлежности для полученных в результате расчетов величин, можно доказать преимущество одной из моделей в данной ситуации (при $X_1 \subseteq X_2$). Например, необходимость применения в АСУ ТП разработки месторождений трехмерной модели пласта вместо двухмерной. На основе такого анализа могут быть построены блоки A автоматического выбора модели в зависимости от неопределенности информации о коэффициентах моделей, граничных и начальных условий.

*Таким образом, попытки применения какого-либо конкретного математического аппарата (интервального анализа, статистических методов, теории игр, детерминированных моделей и т.д.) для принятия решений в условиях неопределенности позволяет адекватно отразить в модели лишь отдельные виды данных и **приводит к безвозвратной потере информации других типов.***

Так, например, при наличии детерминированных моделей не учитывается накопленная статистика о вероятностных распределениях для некоторых параметров, и производится замена этих распределений соответствующими средними значениями. Кроме того, в этом случае проявляется острый дефицит в информации конкретного типа (например, в функциях распределения вероятностей).

Ввиду недостатка информации для строгого применения вероятностных моделей и трудностей оперирования случайными величинами, а также в связи с тем, что с интервальными величинами можно работать в рамках теории нечетких множеств (ТНМ), последняя приобретает здесь важное значение.

Применение ТНМ позволяет провести также согласование различных нечетких решений при наличии нечетких целей, ограничений, коэффициентов, начальных и граничных условий.

При необходимости форму обозначения F -множеств можно применять для обычных (четких) подмножеств из A . С учетом этого базовое множество и пустое подмножество могут быть записаны в виде

$$X = \langle 1, X \rangle, \emptyset = \langle 0, \emptyset \rangle.$$

Для операций над носителями нечетких множеств можно воспользоваться алгебраическими операциями интервального анализа (интервальной арифметики). Интервальный анализ предназначен для работы в условиях неопределенности с величинами, для которых задан лишь интервал допустимых или возможных значений

$$L = [a, b] = \left\{ x \left| \begin{array}{l} x \in X \\ a \leq x \leq b \end{array} \right. \right\}.$$

Интервальная неопределенность представляется достаточно просто в виде нечеткого множества

$$A = \langle 1, \sigma(A) \rangle.$$

Различие между нечеткостью и случайностью приводит к тому, что математические методы нечетких множеств совершенно не похожи на методы теории вероятностей. Они во многих отношениях проще вследствие того, что понятию вероятностной меры в теории вероятностей соответствует более простое понятие функции принадлежности в теории нечетких множеств. По этой причине даже в тех случаях, когда неопределенность в процессе принятия решений может быть представлена вероятностной моделью, обычно удобнее оперировать с ней методами теории нечетких множеств без привлечения аппарата теории вероятностей.

Получение во всех этих моделях решений в нечеткой форме позволяет довести до сведения специалиста, принимающего решение, что если он согласен или вынужден довольствоваться нечеткой формулировкой проблемы и нечеткими сведениями о модели, то он должен быть удовлетворен и нечетким решением задачи.

17.3 Операции над нечеткими множествами

Все приводимые операции над F -множествами определяются через действия над их F -функциями. Здесь операции над F -множествами даны в интерпретации изложенной в работе [13].

17.3.1 Равные F -множества

Множества A и B из $F(X)$ равны ($A = B$) тогда и только тогда, когда для всех $x \in X$.

17.3.2 F -подмножества

Для $A, B \in F(X)$ множество A является подмножеством B ($A \in B$) тогда и только тогда, когда $\mu_A(x) \leq \mu_B(x)$ для всех $x \in X$.

Например, если $A = \langle 1 - (x - 2)^2, (1, 3) \rangle$ и $B = \langle 1 - \frac{1}{4}(x - 2)^2, (0, 4) \rangle$, то $A \in B$.

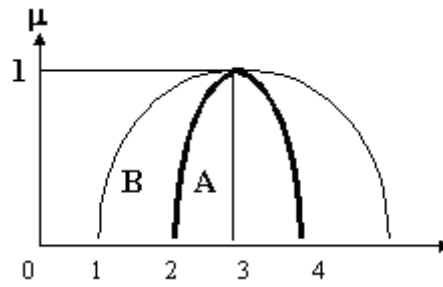


Рисунок 17.4 - Функция принадлежности для нечеткого подмножества A

Замечание. Если A и B четкие множества и $I_A < I_B$, то $A = \emptyset$, что для F -множеств не обязательно.

Например, если

$$A = \left\langle \frac{1}{4} - (x-2)^2, (1,5; 2,5) \right\rangle \text{ и } B = \left\langle 1 - \frac{1}{4}(x-2)^2, (0, 4) \right\rangle,$$

то $\mu_A(x) < \mu_B(x)$, но $A \neq \emptyset$.

17.3.3 Объединение F-множеств

Объединением множеств A и B из $F(X)$ называется множество $C = A \cup B$, F -функция которого определяется следующим образом:

$$\mu_C(x) = \max_{x \in X} (\mu_A(x), \mu_B(x)). \quad (17.5)$$

Объединение соответствует союзу *или* и компактно записывается как:

$$\mu_C = \mu_A(x) \vee \mu_B(x),$$

где символ $\vee$ обозначает операцию взятия *max* (максимума).

Следствие 1. Множество C является наименьшим из множеств, содержащих одновременно A и B .

Доказательство. Пусть F -множество $D \subseteq C$ и содержит A и B , т.е.

$$\mu_D \leq (\mu_C = \mu_A(x) \vee \mu_B(x)) \text{ и } \mu_D \geq \mu_A(x), \mu_D \geq \mu_B(x)$$

$$\text{т.е. } \mu_D \geq (\mu_A(x) \vee \mu_B(x) = \mu_C).$$

Следовательно, $D = C$.

Пример. Если $A = \langle 1 - (x-1)^2, (0; 2) \rangle$ и $B = \langle 1 - (x-2)^2, (1, 3) \rangle$, то

$$\mu_C = \begin{cases} 1 - (x-1)^2; & 0 < x \leq 1,5 \\ 1 - (x-2)^2; & 1,5 < x \leq 3 \end{cases},$$

т.е. $\sigma(A \cup B) = \sigma(A) \cup \sigma(B)$.

17.3.4 Пересечение F-множеств

Пересечением множеств A и B из $F(X)$ называется множество $C=A \cap B$, F -функция которого определяется следующим образом:

$$\mu_C(x) = \min_{x \in X} (\mu_A(x), \mu_B(x)) \quad (17.6)$$

Пересечение соответствует союзу «и», более компактно записывается как

$$\mu_C = \mu_A(x) \wedge \mu_B(x),$$

где символ « $\wedge$ » обозначает операцию взятия $\min$ (минимума).

Пример. Если $A = \langle 1 - (x-1)^2, (0; 2) \rangle$ и $B = \langle 1 - (x-2)^2, (1, 3) \rangle$, то

$$\mu_C = \begin{cases} 1 - (x-2)^2; & 1 < x \leq 1,5 \\ 1 - (x-1)^2; & 1,5 < x \leq 2 \end{cases},$$

т.е. $\sigma(A \cap B) = \sigma(A) \cap \sigma(B)$.

Следствие 2. Множество C является **наибольшим** из множеств, содержащихся одновременно в A и в B .

Доказательство. Пусть F -множество $D \supseteq C$ и принадлежит A и B . Тогда $\mu_D \geq (\mu_C = \mu_A(x) \wedge \mu_B(x))$ и одновременно $\mu_D \leq \mu_A(x), \mu_D \leq \mu_B(x)$ т.е. $\mu_D \leq (\mu_A(x) \wedge \mu_B(x) = \mu_C)$. Следовательно, $D = C$.

17.3.5 Особенности операций пересечения и объединения F-множеств

Известно, что операции объединения и пересечения четких множеств являются коммутативными, ассоциативными и обладают свойствами дистрибутивности по отношению друг к другу. Выявление аналогичных свойств для F -множеств сводится к анализу функций вида

$$f(\alpha, \beta) = \max(\alpha, \beta) \text{ - для объединений,}$$

$$g(\alpha, \beta) = \min(\alpha, \beta) \text{ - для пересечений,}$$

где $\alpha = \mu_A(x); \beta = \mu_B(x); A, B \in F(x)$.

Графически эти функции на плоскости при некотором фиксированном $\beta \in [0, 1]$ изображены на рис. 17.5, где сплошной линией показан график функции g , а пунктиром - f .

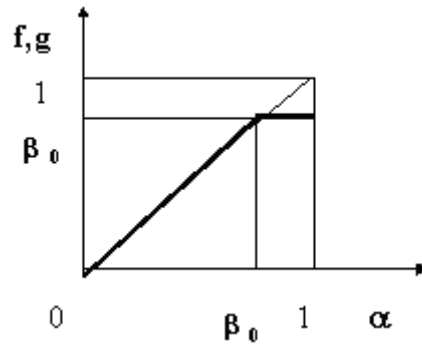


Рис. 17.5 - Графики функций $f(\alpha, \beta_0), g(\alpha, \beta_0)$

Таким образом, f и g являются кусочно-линейными и монотонно возрастающими функциями по каждому из своих аргументов.

Заметим, что если A и B четкие множества с характеристическими функциями $I_A(x)$ и $I_B(x)$, то $A \cup B$ и $A \cap B$ можно представить в виде (17.2), что эквивалентно определению через функции $\min$ и $\max$.

Для F -множеств это уже не верно, так как

$$(\mu_{A \cap B} = \mu_A \cdot \mu_B) \leq (\mu_A \wedge \mu_B),$$

$$(\mu_{A \cup B} = \mu_A + \mu_B - \mu_A \cdot \mu_B) \geq (\mu_A \vee \mu_B).$$

В этом случае следствия 17.1 и 17.2 не выполняются.

Существует несколько способов определения операций объединения и пересечения. Например, для операции пересечения используют иногда алгебраическое произведение функций принадлежности

$$\mu_{A \cap B} = \mu_A \cdot \mu_B.$$

В некоторых случаях $A \cap B$ можно задавать в виде среднего геометрического

$$\mu_{A \cap B} = \sqrt{\mu_A \cdot \mu_B}$$

и, следовательно:

$$\mu_{A \cup B} = 1 - \sqrt{(1 - \mu_A) \cdot (1 - \mu_B)}.$$

Добавим, что $A \cap B$ можно описывать с помощью F -функции:

$$\mu_{A \cap B} = \mu_A \cdot \mu_B + \sqrt{\mu_A \mu_B (1 - \mu_A)(1 - \mu_B)},$$

и $A \cup B$ соответственно, в виде

$$\mu_{A \cup B} = 1 - \left[(1 - \mu_A)(1 - \mu_B) + \sqrt{\mu_A \mu_B (1 - \mu_A)(1 - \mu_B)} \right].$$

Все отмеченные альтернативные варианты объединения и пересечения F -множеств только с определенной степенью точности соответствуют описанию посредством функций $\min$ и $\max$. Поэтому выбор того или иного

подхода зависит от конкретной задачи, когда использование операций $\min$ и $\max$ приводит к неадекватности модели реальной ситуации.

17.3.6 Разность и дополнение F-множеств

Разностью множеств A и B из $F(X)$ называется множество $C=A \setminus B$, с F -функцией вида:

$$\mu_C = \mu_A(x) - \mu_{A \cap B}(x) = \mu_A(x) - \min_{x \in X}(\mu_A(x), \mu_B(x)) = \min_{x \in X}(0, \mu_A(x) - \mu_B(x)). \quad (17.7)$$

Разность $X \setminus A$ называется дополнением F -множества A и обозначается A' . Из (17.7) следует, что

$$\mu_{A'}(x) = 1 - \mu_A(x) \text{ так как } X = \langle 1, X \rangle.$$

Эта операция удобна, например, для перехода от нечеткого множества допустимых значений к множеству недопустимых значений.

Замечание. Если для четких множеств A и B из X всегда выполняются соотношения:

$$(A \setminus B) \cap B = \emptyset,$$

$$A \cap A' = \emptyset,$$

$$A \cup A' = X.$$

то для F -множеств, вообще говоря, это не верно.

Нетрудно проверить, что для A и B из $F(X)$, справедливы следующие соотношения:

1. $A \setminus A = \emptyset,$
2. $A \setminus B \subseteq A,$
3. $A \setminus (A \setminus B) = A \cap B,$
4. $(A \subseteq B) \Leftrightarrow (A \setminus B = \emptyset),$
5. $(A \cap B = \emptyset) \Leftrightarrow (A \setminus B = \emptyset),$
6. $(A \cap B)' = A' \cup B',$
7. $(A \cup B) = A' \cap B',$
8. $(A \subseteq B) \Leftrightarrow (B' \subseteq A').$

Равенства 6 и 7 называются законами де Моргана и следуют, соответственно из тождеств:

$$1 - \max_{x \in X}(\mu_A(x), \mu_B(x)) = \min_{x \in X}(1 - \mu_A(x), 1 - \mu_B(x)),$$

$$1 - \min_{x \in X}(\mu_A(x), \mu_B(x)) = \max_{x \in X}(1 - \mu_A(x), 1 - \mu_B(x)).$$

17.3.7 Другие отношения F-множеств

Следующие соотношения, которые приводятся без доказательств, являются следствием довольно очевидных свойств функций F -множеств. Здесь: $\emptyset(x)=0$ и $U(x)=1$ для всех $x \in X$, а также $A, B, C, A_1, \dots, A_n \in F(x)$.

Таблица 17.1 – Соотношения между F -множествами

Свойство	Формализованная запись свойства
Идемпотентность	$A \cup A = A$ $A \cap A = A$
Коммутативность	$A \cap B = B \cap A$ $A \cup B = B \cup A$
Ассоциативность	$A \cap (B \cap C) = (A \cap B) \cap C$ $A \cup (B \cup C) = (A \cup B) \cup C$
Поглощение	$A \cap (A \cup B) = A$ $A \cup (A \cap B) = A$
Дистрибутивность	$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$ $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$
Инволютивность	$(A')' = A$
Законы де Моргана	$(A \cap B)' = A' \cup B'$ $(A \cup B)' = A' \cap B'$
Граничные условия	$A \cup \emptyset = A$ $A \cap \emptyset = \emptyset$ $A \cup U = U$ $A \cap U = A$

17.4 Понятие о нечеткой логике

По аналогии с булевыми переменными теории нечеткой логики вводятся понятие нечеткой логической переменной – нечеткое высказывание.

Нечеткое высказывание $\tilde{A}$ - предложение, относительно которого можно судить о степени его истинности или ложности в настоящее время. Степень истинности или ложности $d(\tilde{A})$ принимает значения из $[0; 1]$, где 0, 1 - предельные значения степени истинности и совпадают с понятиями «лжи» и «истины» для четких высказываний.

Нечеткое высказывание со степенью истинности 0,5 называется **индифферентностью**, поскольку оно истинно в той же мере, что и ложно.

Примеры: «2 - маленькое число» - нечеткое высказывание, степень истинности которого $d(\tilde{A})=0,9$. «Петров занимается большой общественной работой» - нечеткое высказывание со степенью истинности $d(\tilde{A})=0,3$.

Отрицанием нечеткого высказывания $\tilde{A}$ является $\neg\tilde{A}$, степень истинности которого определяется выражением:

$$d(\neg\tilde{A})=1-d(\tilde{A}).$$

Из этого определения следует, что степень ложности $\neg\tilde{A}$ совпадает со степенью истинности для $\tilde{A}$.

Конъюнкцией нечетких высказываний $\tilde{A}$ и $\tilde{N}$, называется нечеткое высказывание $\tilde{A}\&\tilde{N}$, степень истинности которого совпадает со степенью истинности менее истинного высказывания:

$$d(\tilde{A}\&\tilde{N})=\min(d(\tilde{A}), d(\tilde{N})).$$

Дизъюнкцией нечетких высказываний $\tilde{A}$ и $\tilde{N}$, называется нечеткое высказывание $\tilde{A}\vee\tilde{N}$, степень истинности которого совпадает со степенью истинности более истинного высказывания:

$$d(\tilde{A}\vee\tilde{N})=\max(d(\tilde{A}), d(\tilde{N})).$$

Импликацией нечетких высказываний $\tilde{A}$ и $\tilde{N}$ называется нечеткое высказывание $\tilde{A}\rightarrow\tilde{N}$, степень истинности которого:

$$d(\tilde{A}\rightarrow\tilde{N})=\max(1-d(\tilde{A}), d(\tilde{N})).$$

Истинность импликации не меньше чем степень ложности ее посылки или степень истинности ее следствия.

Пример. Пусть нечеткое высказывание $\tilde{A}$ имеет степень истинности $d(\tilde{A})=0,3$; нечеткое высказывание $\tilde{N}$ - $d(\tilde{N})=0,6$. Импликация этих высказываний $\tilde{A}\rightarrow\tilde{N}$ будет иметь степень истинности

$$d(\tilde{A}\rightarrow\tilde{N})=\max(0,7, 0,6)=0,7.$$

Степень импликации тем выше, чем меньше степень истинности посылки или больше степень истинности следствия.

Эквивалентностью нечетких высказываний $\tilde{A}$ и $\tilde{N}$, называется нечеткое высказывание $\tilde{A}\leftrightarrow\tilde{N}$:

$$d(\tilde{A}\leftrightarrow\tilde{N})=\min[\max(1-d(\tilde{A}), d(\tilde{N})), \max(1-d(\tilde{N}), d(\tilde{A}))].$$

Истинность эквивалентности совпадает со степенью истинности менее истинной из импликаций $\tilde{A}\rightarrow\tilde{N}$ и $\tilde{N}\rightarrow\tilde{A}$.

Если степень истинности высказываний 0 или 1, то все определения соответствуют логическим операциям над четкими высказываниями.

Два высказывания $\tilde{A}$ и $\tilde{N}$ называются **нечетко близкими**, если степень истинности $\tilde{A}\leftrightarrow\tilde{N}$ больше или равна 0,5. В последнем случае будем называть $\tilde{A}$ и $\tilde{N}$ **взаимно нечетко индифферентными**.

Порядок выполнения операций над нечеткими высказываниями следующий:

- скобки,
- отрицание,
- конъюнкция,
- дизъюнкция,
- импликация,
- эквивалентность.

Пример. Вычислим степень истинности составного нечеткого высказывания:

$$\tilde{U} = (\tilde{A} \& \neg \tilde{N} \vee \neg \tilde{A} \& \tilde{N}) \rightarrow \neg(\tilde{A} \& \tilde{I})$$

если: $\tilde{A} = 0,7$, $\tilde{N} = 0,4$, $\tilde{I} = 0,9$.

$$\begin{aligned} d(\tilde{U}) &= \max(1 - d(\tilde{A} \& \neg \tilde{N} \vee \neg \tilde{A} \& \tilde{N}), d(\neg(\tilde{A} \& \tilde{I}))) = \\ &= \max(1 - \max[d(\tilde{A} \& \neg \tilde{N}), d(\neg \tilde{A} \& \tilde{N})], d(1 - (\tilde{A} \& \tilde{I}))) = \\ &= \max(1 - \max[\min(d(\tilde{A}), 1 - d(\tilde{N})), \min(1 - d(\tilde{A}), d(\tilde{N}))], 1 - \min(d(\tilde{A}), d(\tilde{I}))) = \\ &= \max(1 - \max[\min(0,7, 0,6), \min(0,3, 0,4)], 1 - \min(0,7, 0,9)) = 0,4. \end{aligned}$$

В основу главы 17 положен материал монографии [13], дополненный материалом работ [14, 15, 16, 17].

18. ГЕНЕТИЧЕСКИЕ АЛГОРИТМЫ

18.1 Основные понятия, принципы и предпосылки генетических алгоритмов

Генетические Алгоритмы (ГА) – это адаптивные методы функциональной оптимизации, основанные на компьютерном имитационном моделировании биологической эволюции. Основные принципы ГА были сформулированы Голландом (Holland, 1975), и хорошо описаны во многих работах и на ряде сайтов в Internet.

В настоящее время существует ряд теорий биологической эволюции (Ж.-Б. Ламарка, П. Тейяра де Шардена, К.Э. Бэра, Л.С. Берга, А.А. Любищева, С.В. Мейена и др.), однако, ни одна из них не считается общепризнанной. Наиболее известной и популярной, конечно, является теория биологической эволюции Чарльза Дарвина. Эта теория, как и другие, содержит довольно много *нерешенных проблем*. Можно отметить лишь некоторые наиболее известные из них. Как это ни парадоксально, но несмотря на то, что сам Чарльз Дарвин назвал свою работу «Происхождение Видов», но как раз именно *происхождения видов* она и не объясняет. Дело в том, что возникновение нового вида «по алгоритму Дарвина» является крайне маловероятным событием, т. к. для этого требуется случайное возникновение в одной точке пространства и времени сразу не менее 100 особей нового вида, т. е. особей, которые могли бы иметь плодовитое потомство. При меньшем количестве особей вид обречен на вымирание. Поэтому процесс видообразования на основе *случайных* мутаций должен был бы занять несуразно много времени (по некоторым оценкам даже в намного раз больше, чем время существования Вселенной). Кроме того, «алгоритм Дарвина» не объясняет явной *системности* в многообразии возникающих форм, типа *закона гомологичных рядов* Н.И. Вавилова. Поэтому Л.С. Берг предложил очень интересную концепцию *номогенеза – закономерной* или *направленной* эволюции живого. В этой концепции предполагается, что филогенез имеет определенное направление и *смена форма является не случайной, а задается некоторым вектором*, природа которого не ясна. Идеи номогенеза глубоко разработал и развил А.А. Любищев, высказавший гипотезу о математических закономерностях, которые определяют многообразие живых форм. Кроме того, Дарвин не смог показать *механизм наследования*, при котором поддерживается и закрепляется изменчивость. Это было на пятьдесят лет до того, как генетическая теория наследственности начала распространяться по миру, и за тридцать лет до того, как «эволюционный синтез» укрепил связь между теорией эволюции и молодой генетикой.

Тем ни менее и не смотря на свои недостатки, *именно теория Дарвина традиционно и моделируется в ГА*, хотя, конечно, это не исключает

возможности моделирования и других теорий эволюции в ГА. Более того, возможно именно такое компьютерное моделирование и сравнение его результатов с картиной реальной эволюции жизни на Земле может быть и сыграет положительную роль в дальнейшей разработке наиболее адекватной теории биологической эволюции.

Теория Дарвина применима не к отдельным особям, а к **популяциям** – большому количеству особей **одного вида**, т.е. способных давать плодовитое потомство, находящейся в определенной статичной или динамичной внешней среде.

В основе модели эволюции Дарвина лежат **случайные изменения** отдельных материальных элементов живого организма при переходе от поколения к поколению. Целесообразные изменения, которые облегчают выживание и производство потомков в данной конкретной внешней среде, сохраняются и передаются потомству, т.е. **наследуются**. Особи, не имеющие соответствующих приспособлений, погибают, не оставив потомства или оставив его меньше, чем приспособленные (считается, что количество потомства пропорционально степени приспособленности). Поэтому в результате **естественного отбора** возникает популяция из наиболее приспособленных особей, которая может стать основой нового вида.

Естественный отбор происходит в условиях **конкуренции** особей популяции, а иногда и различных видов, друг с другом за различные **ресурсы**, такие, например, как пища или вода. Кроме того, члены популяции одного вида часто конкурируют за привлечение брачного партнера. Те особи, которые наиболее приспособлены к окружающим условиям, будут иметь относительно больше шансов воспроизвести потомков. Слабо приспособленные особи либо совсем не произведут потомства, либо их потомство будет очень немногочисленным. Это означает, что гены от высоко адаптированных или приспособленных особей будут распространяться в увеличивающемся количестве потомков на каждом последующем поколении.

Генетические алгоритмы заимствуют из биологии:

- понятийный аппарат;
- идею коллективного поиска экстремума при помощи популяции особей;
- способы представления генетической информации;
- способы передачи генетической информации в череде поколений (генетические операторы);
- идею о преимущественном размножении наиболее приспособленных особей (приспособленность особи пропорциональна количеству ее потомков).

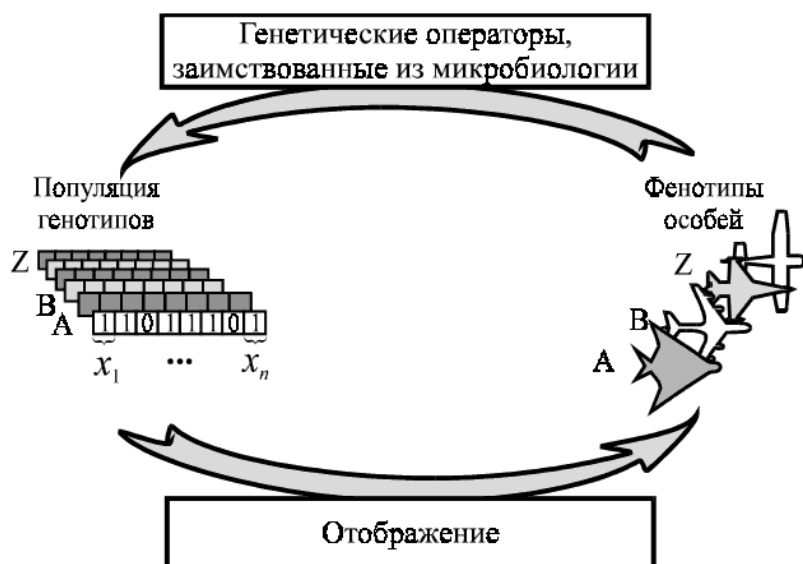


Рисунок 18.1 – Преобразование наследственной информации в ГА

Таким образом, по сути дела каждый конкретный *генетический алгоритм представляют имитационную модель некоторой определенной теории биологической эволюции или ее варианта*. Вместе с тем необходимо отметить, что сами исследователи биологической эволюции пока еще не до конца определились с критериями и методами определения степени существенности для поддерживаемой ими теории эволюции тех или иных биологических процессов, которые собственно и моделируются в генетических алгоритмах.

18.2 Принцип функционирования генетического алгоритма

Пусть дана некоторая целевая функция, в общем случае зависящая от нескольких переменных, и требуется найти такие значения переменных, при которых значение функции максимально.

Генетический алгоритм — это простая модель эволюции в природе, реализованная в виде алгоритма. В нем используются как аналог механизма генетического наследования, так и аналог естественного отбора. При этом используется биологическая терминология. Мы имеем дело с особью (индивидуумом). Особь — это некоторое решение задачи. Будем считать особь тем более приспособленной, чем лучше соответствующее решение (чем больше значение целевой функции это решение дает). Тогда задача максимизации целевой функции сводится к поиску более приспособленной особи. То есть, выбирая наиболее приспособленную особь в текущем поколении, можно получить не абсолютно точный, но близкий к оптимальному ответ. Особи наделяются хромосомами.

Для моделирования наследования в генетических алгоритмах используются следующие термины и понятия [21]:

Хромосома — вектор (последовательность), содержащий набор значений.

Каждая позиция в хромосоме называется **геном**.

Набор хромосом — вариант решения задачи.

Целевая функция оценки приспособленности решений.

Основные операторы генетического алгоритма [21]:

- **Отбор** — селекция (reproduction, selection) осуществляет отбор хромосом в соответствии со значениями их функции приспособленности.
- **Скращивание (crossover)** — операция, при которой две хромосомы обмениваются своими частями.
- **Мутация** — случайное изменение одной или нескольких позиций в хромосоме.

Поясним подробнее функционирование основных операторов.

Отбор — это процесс формирования новой популяции из старой на основании определенных правил, после чего к новой популяции опять применяются операции кроссовера и мутации, затем опять происходит отбор, и так далее. В традиционном генетическом алгоритме реализуется отбор пропорционально приспособленности (так называемая селекция), одноточечный кроссовер и мутация [21].

Вообще говоря, после того как выбраны основные операторы генетического алгоритма (отбор, кроссовер и мутация), сам алгоритм определяется однозначно. При этом существуют различные методы выбора операторов, которые могут дать различные результаты реализации.

Для оператора отбора (селекции) наиболее распространенными являются следующие методы [21]:

1. *метод рулетки (roulette-wheel selection);*
2. *метод элитного отбора;*
3. *метод турнирного отбора (tournament selection).*

Метод рулетки заключается в том, что особи отбираются с помощью *n* «запусков» рулетки. В этом случае колесо рулетки содержит по одному сектору для каждого члена популяции. Размер *i*-ого сектора пропорционален соответствующей величине $P(i)$ вычисляемой по формуле [21]:

$$P(i) = \frac{f(i)}{\sum_{i=1}^n f(i)}.$$

При таком отборе члены популяции с более высокой приспособленностью с большей вероятностью будут чаще выбираться, чем особи с низкой приспособленностью.

Элитный отбор (De Jong, 1975) гарантируют, что при отборе обязательно будут выживать лучший или лучшие члены популяции совокупности (наиболее распространена процедура обязательного сохранения только одной лучшей особи, если она не прошла как другие через процесс отбора, кроссовера и мутации);

Метод турнирного отбора — реализует n турниров, чтобы выбрать n особей. Каждый турнир построен на выборке m элементов из популяции, и выбора лучшей особи среди них. Наиболее распространен турнирный отбор с $m = 2$. При этом решение, какие именно особи переходят в следующую популяцию, также может быть различным [21].

Например, мы можем всегда оставлять в популяции потомков, даже если оценка их приспособленности хуже, чем у родителей. Мы можем сравнивать потомков с родителями и оставлять в популяции лучших. Элитные методы отбора гарантируют, что при отборе обязательно будут выживать лучшие члены популяции.

Наиболее распространена процедура обязательного сохранения только одной лучшей особи, если она не прошла как другие через процесс отбора, кроссовера и мутации. Однако здесь следует иметь в виду, что критерий отбора хромосом не дает нам гарантию нахождения наилучшего решения, поскольку эволюция может пойти по пути неоптимального отбора или из популяции будут исключены неперспективные родители, потомки которых могут оказаться эффективными [21].

После отбора n избранных особей случайным образом разбиваются на $n/2$ пар. Для каждой пары с вероятностью p может применяться скрещивание. Соответственно, с вероятностью $1-p$ скрещивание не происходит и неизменные особи переходят на стадию мутации.

Скрещивание (crossover) — осуществляет обмен частями хромосом, как правило, между двумя хромосомами в популяции. Скрещивание может быть [21]:

- однотоочечным,
- многотоочечным.

Однотоочечное скрещивание работает следующим образом. Сначала, случайным образом выбирается одна из $l-1$ точек разрыва. Точка разрыва — участок между соседними генами. Обе родительские структуры разрываются на два сегмента по этой точке. Затем, соответствующие сегменты различных родителей склеиваются и получаются два генотипа потомков [21].

При **двухточечном скрещивании** выбираются две точки разрыва, и родительские хромосомы обмениваются сегментом, находящимся между этими точками, и равномерное скрещивание, в котором каждый бит первого родителя наследуется первым потомком с заданной вероятностью, в противном случае этот бит передается второму потомку и наоборот. Достаточно часто хорошо работает метод, при котором одинаковый участок генов для хромосом остается неизменным (и является некоторым шаблоном), и осуществляется обмен генов, не входящих в данный участок [21].

Мутация — стохастическое изменение части хромосом. В хромосоме, которая подвергается мутации, каждый ген с вероятностью P_{mut} (обычно очень маленькой) меняется на некоторый другой из заданного диапазона значений [21].

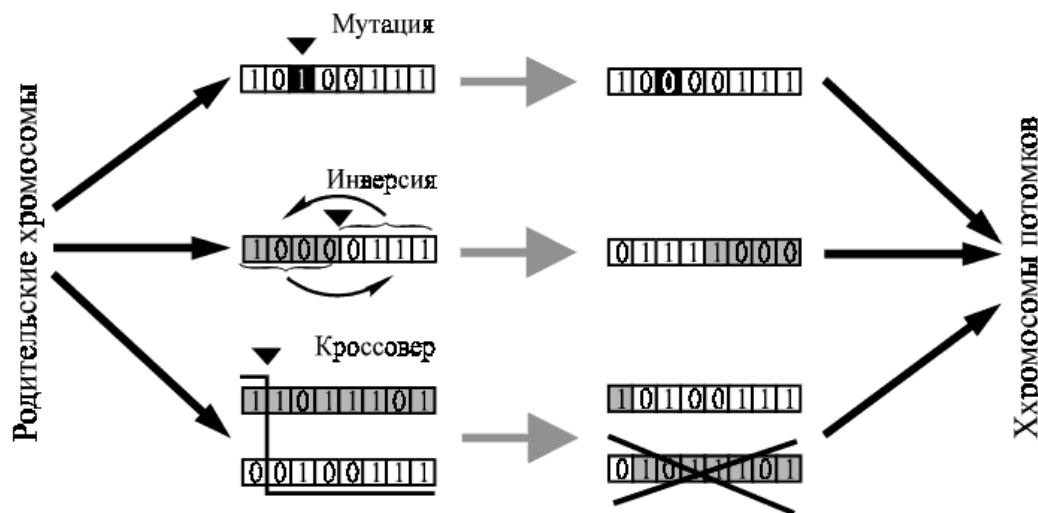


Рисунок 18.2 – Принцип проведения операций в ГА

Работа ГА представляет собой итерационный процесс, который продолжается до тех пор, пока поколения не перестанут существенно отличаться друг от друга, или не пройдет заданное количество поколений или заданное время. Для каждого поколения реализуются отбор, а также операции модификации (рис. 18.2):

- мутация,
- инверсия,
- кроссовер (скрещивание).

18.2.1 Алгоритм функционирования простейшего генетического алгоритма

Рассмотрим алгоритм простейшего генетического алгоритма приведенного на рисунке 18.3.

Шаг 1: генерируется начальная популяция, состоящая из N особей со случайными наборами признаков.

Шаг 2 (борьба за существование): вычисляется *абсолютная приспособленность* каждой особи популяции к условиям среды $f(i)$ и суммарная приспособленность особей популяции, характеризующая приспособленность всей популяции. Затем при *пропорциональном отборе* для каждой особи вычисляется ее *относительный вклад в суммарную приспособленность популяции* $P_s(i)$, т. е. отношение ее абсолютной приспособленности $f(i)$ к суммарной приспособленности всех особей популяции (18.1):

$$P_s(i) = \frac{f(i)}{\sum_{i=1}^N f(i)}. \quad (18.1)$$

В выражении (18.1) сразу обращает на себя внимание возможность сравнения абсолютной приспособленности i -й особи $f(i)$ не с суммарной приспособленностью всех особей популяции, а со средней абсолютной приспособленностью особи популяции (18.2):

$$\bar{f} = \frac{1}{N} \sum_{i=1}^N f(i). \quad (18.2)$$

Тогда получим (18.3):

$$P(i) = \frac{f(i)}{\bar{f}} = \frac{f(i)}{\frac{1}{N} \sum_{i=1}^N f(i)}. \quad (18.3)$$

Если взять логарифм по основанию 2 от выражения (18.3), то получим *количество информации, содержащееся в признаках особи о том, что она выживет и даст потомство* (18.4).

$$I(i) = \log_2 \frac{f(i)}{\bar{f}}. \quad (18.4)$$

Необходимо отметить, что эта формула совпадает с формулой для семантического количества информации Харкевича, если целью считать *индивидуальное выживание и продолжение рода*. Это значит, что даже чисто формально *приспособленность особи представляет собой количество информации, содержащееся в ее фенотипе о продолжении ее генотипа в последующих поколениях*.

Поскольку количество потомства особи пропорционально ее приспособленности, то естественно считать, что *если это количество информации* [1]:

- *положительно*, то данная особь выживает и дает потомство, численность которого пропорциональна этому количеству информации;
- *равно нулю*, то особь доживает до половозрелого возраста, но потомства не дает (его численность равна нулю);
- *меньше нуля*, то особь погибает до достижения половозрелого возраста.

Таким образом, можно сделать фундаментальный вывод, имеющий даже мировоззренческое звучание, о том, что ***естественный отбор представляет собой процесс генерации и накопления информации о выживании и продолжении рода в ряде поколений популяции, как системы.***

Это накопление информации происходит на различных уровнях иерархии ***популяции, как системы***, включающей [1]:

- элементы системы: отдельные особи;
- взаимосвязи между элементами: отношения между особями в популяции, обеспечивающие передачу последующим поколениям максимального количества информации об их выживании и продолжении рода (путем скрещивания наиболее приспособленных особей и наследования рациональных приобретений);
- цель системы: сохранение и развитие популяции, реализуется через цели особей: индивидуальное выживание и продолжение рода.

Фенотип соответствует генотипу и представляет собой его внешнее проявление в признаках особи. Особь взаимодействует с окружающей средой и другими особями в соответствии со своим фенотипом. В случае, если это взаимодействие удачно, то особь передает генетическую информацию, определяющую фенотип, последующим поколениям.

Шаг 3: начало цикла смены поколений.

Шаг 4: начало цикла формирования нового поколения.

Шаг 5 (отбор): осуществляется пропорциональный отбор особей, которые могут участвовать в продолжении рода. Отбираются только те особи популяции, у которых количество информации в фенотипе и генотипе о выживании и продолжении рода положительно, причем вероятность выбора пропорциональна этому количеству информации.

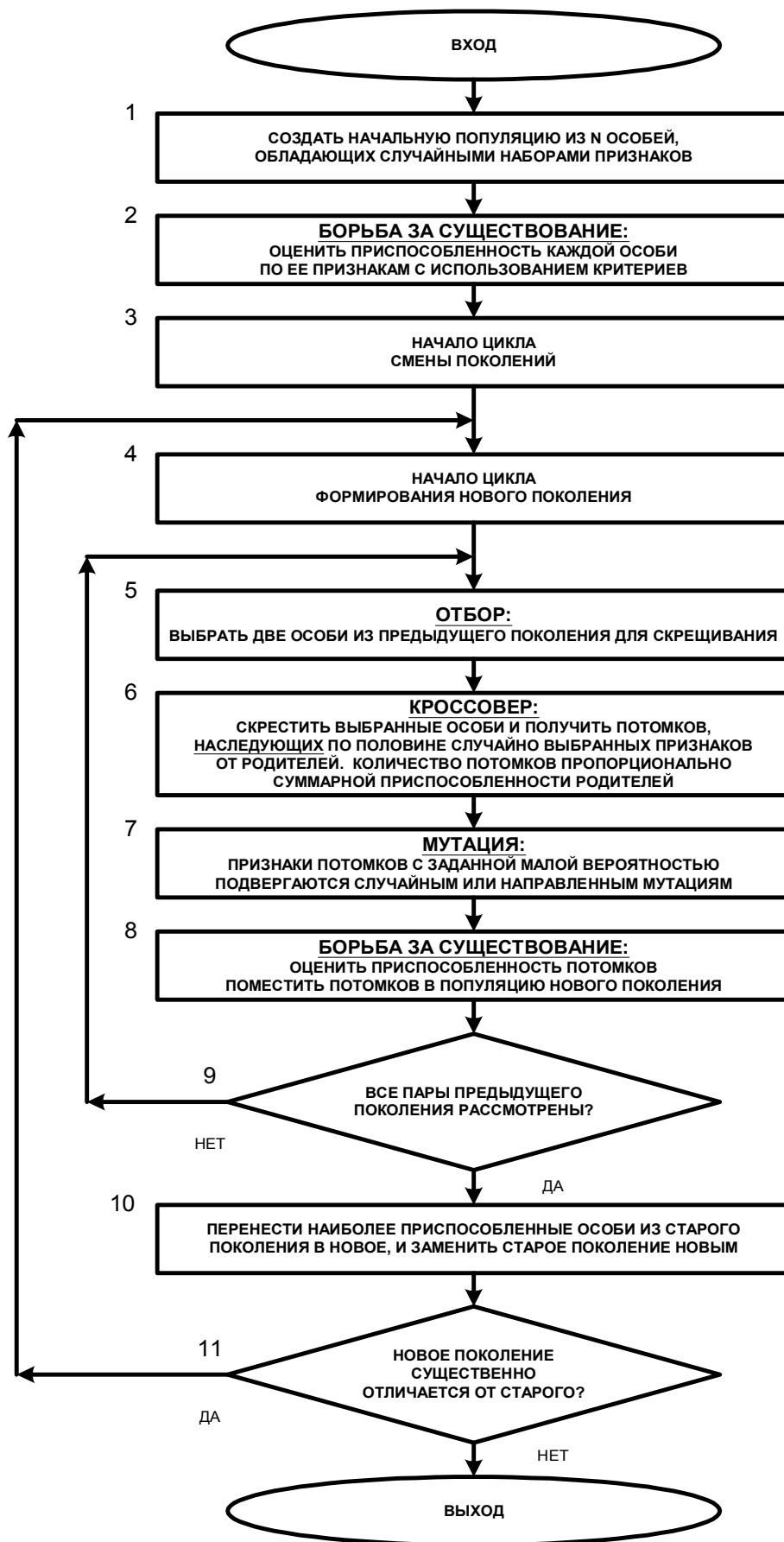


Рисунок 18.3 – Пример генетического алгоритма

Шаг 6 (кроссовер): отобранные для продолжения рода на предыдущем шаге особи с заданной вероятностью P_c подвергаются **скрещиванию** или **кроссоверу (рекомбинации)**.

- Если кроссовер происходит, то потомки получают по половине случайным образом определенных признаков от каждого из родителей. Численность потомства пропорциональна суммарной приспособленности родителей. В некоторых вариантах ГА потомки после своего появления заменяют собой родителей и переходят к мутации.
- Если кроссовер не происходит, то исходные особи – несостоявшиеся родители, переходят на стадию мутации.

Шаг 7 (мутация): выполняются операторы **мутации**. При этом признаки потомков с вероятностью P_m случайным образом изменяются на другие. Отметим, что использование механизма случайных мутаций роднит генетические алгоритмы с таким широко известным методом имитационного моделирования, как метод Монте-Карло.

Шаг 8 (борьба за существование): оценивается приспособленность потомков (по тому же алгоритму, что и на шаге 2).

Шаг 9: проверяется, все ли отобранные особи дали потомство.

Если нет, то происходит переход на шаг 5 и продолжается формирование нового поколения, иначе – переход на следующий шаг 10.

Шаг 10: происходит смена поколений:

- потомки помещаются в новое поколение;
- наиболее приспособленные особи из старого поколения переносятся в новое, причем для каждой из них это возможно не более заданного количества раз;
- полученная новая популяция замещает собой старую.

Шаг 11: проверяется выполнение условия останова генетического алгоритма. **Выход** из генетического алгоритма происходит либо тогда, когда новые поколения перестают существенно отличаться от предыдущих, т.е., как говорят, «алгоритм сходится», либо когда пройдено заданное количество поколений или заданное время работы алгоритма (чтобы не было "зацикливания" и динамического зависания в случае, когда решение не может быть найдено в заданное время).

Если ГА сошелся, то это означает, что решение найдено, т.е. получено поколение, идеально приспособленное к условиям данной фиксированной среды обитания.

Иначе – переход на **шаг 4** – начало формирования нового поколения.

В реальной биологической эволюции этим дело не ограничивается, т.к. любая популяция кроме освоения некоторой экологической ниши пытается также выйти за ее пределы освоить и другие ниши, как правило "смежные". Именно за счет этих процессов жизнь вышла из моря на сушу, проникла в воздушное пространство и поверхностный слой почвы, а сейчас осваивает космическое пространство.

Конечно, реальные генетические алгоритмы, на которых проводятся научные исследования, чаще всего мало похожи на приведенный пример. Исследователи экспериментируют с различными параметрами генетических алгоритмов, например: способами отбора особей для скрещивания; критериями приспособленности и жесткостью влияния факторов среды; способами выбора признаков, передающихся от родителей потомкам (рецессивные и не рецессивные гены и т.д.); интенсивностью, видом случайного распределения и направленностью мутаций; различными подходами к воспроизводству и отбору.

Поэтому под термином «генетические алгоритмы» по сути дела надо понимать не одну модель, а довольно широкий класс алгоритмов, подчас мало похожих друг на друга.

18.2.2 Репродуктивный план Холланда, как пример реализации генетического алгоритма

Не смотря на то, что модели биологической эволюции, реализуемые в ГА, обычно сильно упрощены по сравнению с природным оригиналом, тем ни менее ГА являются мощным средством, которое может с успехом применяться для решения широкого класса прикладных задач, включая те, которые трудно, а иногда и вовсе невозможно, решить другими методами.

Ниже приводится репродуктивный план Холланда (рис. 18.4), который является примером одной из реализаций ГА приведенной в работе [20].

Шаг 1 - Инициализация начальной популяции. Ввести точку отчета $t=0$. Инициализировать образом M генотипов особей A_i и сформировать из них начальную популяцию:

$$B(0) = (A_1(0), \dots, A_M(0)).$$

Вычислить приспособленность особей популяции:

$$v(0) = (\mu_1(0), \dots, \mu_M(0)),$$

а затем – среднюю приспособленность по популяции

$$\hat{\mu}(0) = \frac{1}{M} \sum_{h=1}^M \mu_h(0).$$

Шаг 2 – Выбор родителей для скрещивания. Увеличить время на единицу $t=t+1$. Определить случайную переменную $Rand_t$ на множестве

$\zeta_M = \{1, \dots, M\}$, назначив вероятность выпадения любого $h \in \zeta_M$ пропорциональной $\mu_h(t)/\hat{\mu}(t)$. Сделать одно испытание $Rand_i$ и вычислить результат $i(t)$, который определит номер первого родителя $A_{i(t)}(t)$. Повторным испытанием определить номер второго родителя $i'(t)$.

Шаг 3 – Формирование генотипа потомка. С вероятностью P_c произвести над генотипами выбранных родителей кроссовер. Выбрать с вероятностью 0,5 один из результатов и его как $^1A(t)$. Последовательно применить к $^1A(t)$ оператор инверсии (с вероятностью P_i), а затем мутации (с вероятностью P_m). Полученный генотип потомка сохранить как $A'(t)$.

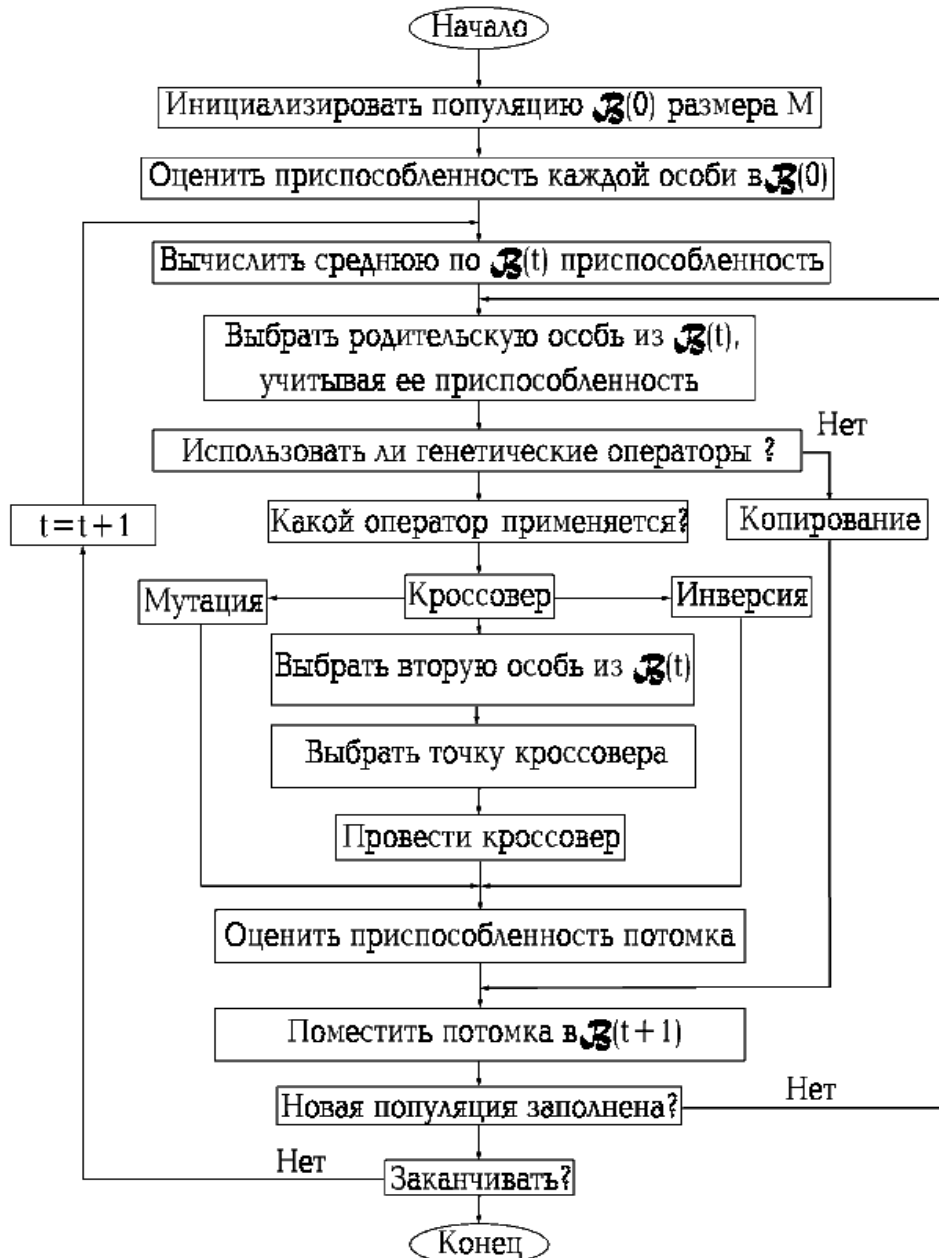


Рисунок 18.4 - Репродуктивный план Холланда

Шаг 4 – Отбор особи на элиминирование и замена ее потомком. С равной вероятностью $1/M$ для всех $h \in \zeta_M$ определить случайным образом номер $j(t)$ особи в популяции, которую заместит потомок. Обновить текущую популяцию $B(t)$ путем замены $A_{i(t)}(t)$ на $A'(t)$.

Шаг 5 – Определение приспособленности потомка. Вычислить приспособленность потомка $\mu_E(A'(t))$. Обновить вектор приспособленности особей $v(t)$ и значение средней приспособленности $\hat{\mu}(t)$.

Шаг 6 – перейти к шагу 2.

18.3 Достоинства и недостатки генетических алгоритмов

К недостаткам ГА следует отнести следующие:

- не гарантирует обнаружения глобального решения за приемлемое время;
- не гарантируют того, что найденное решение будет оптимальным.

В случаях, когда задача может быть решена специально разработанным для нее методом, практически всегда такие методы будут эффективнее ГА как по быстродействию, так и по точности найденных решений.

Тем ни менее ГА применимы для поиска «достаточно хорошего» решения задачи за «достаточно короткое время». *ГА представляют собой разновидность алгоритмов поиска и имеют преимущества перед другими алгоритмами при очень больших размерностях задач и отсутствия упорядоченности в исходных данных, когда альтернативой им является метод полного перебора вариантов.*

Главным достоинством ГА является то, что они могут применяться для решения сложных неформализованных задач, для которых не разработано специальных методов, т.е. ГА обеспечивают решение проблем. Но даже в тех случаях, для которых хорошо работают существующие методики, можно достигнуть интересных результатов сочетая их с ГА.

Генетические алгоритмы представляют собой компьютерное моделирование эволюции. Материальное воплощение сконструированных таким образом систем до сих пор была невозможна без участия человека. Однако в настоящее время *интенсивно ведутся работы, по уменьшению зависимости машинной эволюции от человека, по двум основным направлениям:*

1. *Естественный отбор, моделируемый ГА, переносится из виртуального мира в реальный, например, проводятся эксперименты по реальным битвам роботов на выживание.*

2. *Интеллектуальные системы, основанные на ГА, конструируют роботов, которые в принципе могут быть изготовлены на автоматизированных заводах без участия человека.*

18.4 Примеры применения генетических алгоритмов

В 1994 году Эндрю Кин из университета Саутгемптона использовал генетический алгоритм в дизайне космических кораблей. За основу была взята модель опоры космической станции, спроектированной в NASA из которой после смены 15 поколений, включавших 4 500 вариантов дизайна, получилась модель, превосходящая по тестам тот вариант, что разработали люди.

Джон Коза из Стэнфорда разработал технологию генетического программирования, в которой результатом эволюции становятся не отдельные числовые параметры «особей», а целые имитационные программы, которые являются виртуальными аналогами реальных устройств. Эта технология позволила компании Genetic Programming повторить 15 человеческих изобретений, 6 из которых были запатентованы после 2000 года, то есть представляют собой самые передовые достижения, а один из контроллеров, «выведенных» в GP, даже превосходит аналогичную человеческую разработку.

Пример воплощения ГА в реальной битве роботов на выживание: в 2002 году в британском центре Magna открылся павильон Live Robots, где боролись за выживание 12 роботов двух видов: «гелиофаги», способные добывать электроэнергию с использованием солнечных батарей; «хищники», которые могли получать электроэнергию только от гелиофагов. Те хищники, которые забирали всю энергию у гелиофагов, теряли источник питания и погибали, не передавая свою тактику потомкам, поступавшие же «более разумно» продолжили свой род. В результате возникла равновесная сбалансированная искусственная экосистема с двумя популяциями.

Пример конструирования роботов роботами: в Brandeis University была создана программа Golem, которая сама конструировала роботов. В программу была база деталей, а также механизм мутаций и функция пригодности для «отсеивания» неудачников – тех, кто не научился двигаться. После 600 поколений за несколько дней программа получила модели трех ползающих роботов. Показательно, что роботы оказались симметричными, хотя симметрия никак не была явно прописана в правилах эволюции и исходных данных. Это означает, что она появилась в ходе моделирования машинной эволюции как полезная черта, позволяющая двигаться прямолинейно.

В основу главы 18 положено обобщение материала работ [18- 21].

19. ОБРАБОТКА ЕСТЕСТВЕННОГО ЯЗЫКА В ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМАХ

19.1 Основные понятия о системах, использующих естественный язык

Системы, в которых используется естественный язык (ЕЯ) можно разделить на классы как показано на рис. 19.1 [3].

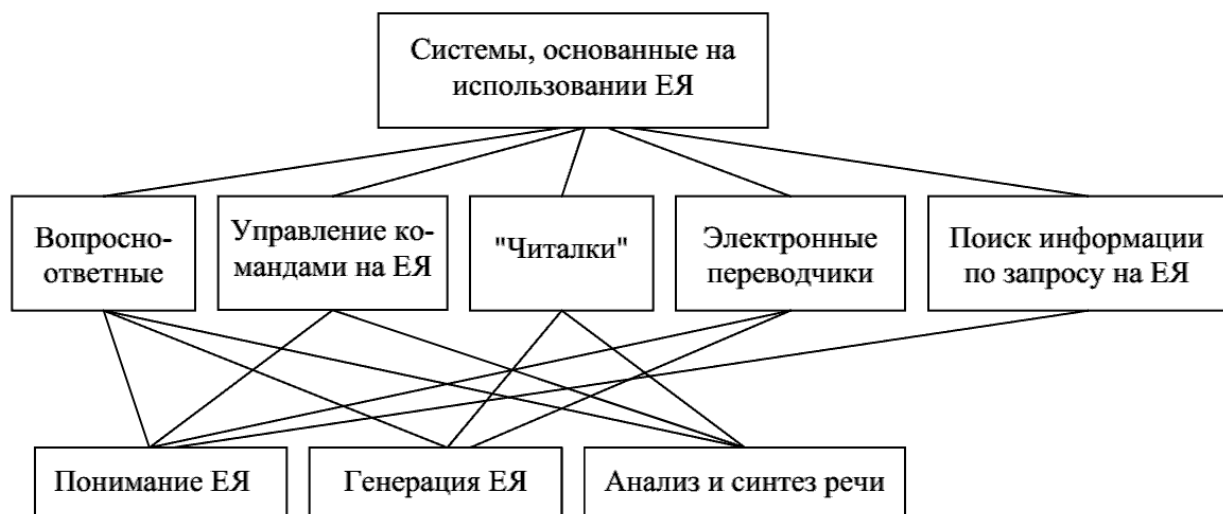


Рисунок 19.1 - Основные классы прикладных систем, основанных на ЕЯ, и решаемые при их создании задачи

Говоря о возможных решениях задачи анализа естественного языка (моделирования понимания системой ИИ естественного языка), можно выделить следующие два основных подхода [3]:

1. **Психолингвистический подход** – состоит в моделировании психологических механизмов человека, обеспечивающих полноценное понимание естественно-языковых текстов.
2. **Утилитарный подход** – состоит в создании технических средств, обеспечивающих взаимодействие на естественном языке с компьютерными системами различного назначения, решающими какие-то свои задачи, сами по себе, чаще всего нелингвистические.

Первый из них связан с большими трудностями, т. к. естественный язык и диалог на нем отражает в себе все многообразие и сложность мышления в естественном мозге. Это – использование контекста, умолчаний, анафорических ссылок, метафор, здравого смысла, внелингвистических методов передачи информации в процессе диалога (жесты, мимика). Таким

образом, его реализацию можно приравнять к реализации искусственного разума в полном объеме.

Утилитарный подход связан с разработкой систем, понимающих ограниченный или похожий на естественный язык (деловой прозы).

Существуют два основных подхода к реализации систем, моделирующих понимание естественного языка (ЕЯ) [3]:

- синтаксически-ориентированный,
- семантически-ориентированный.

В синтаксически-ориентированном подходе строго выдерживается следующая последовательность этапов анализа [3]:

1) морфологический анализ – анализ структуры слов, т. е. распознавание корня и аффиксов (приставок, суффиксов, окончаний), с использованием словарей корней и аффиксов;

2) синтаксический анализ – анализ структуры предложения, т. е. частей предложения (или ролей слов в нем) с использованием грамматики языка;

3) семантический анализ – анализ смысла предложения, т. е. интерпретация его в терминах представления смысла, с использованием базы знаний о предметной области и знаний о синтаксисе представления смысла;

4) прагматический анализ – анализ целей предложения или ожиданий и желаний его источника с целью планирования реакции на анализируемое предложение.

Недостатками синтаксически-ориентированного подхода являются [3]:

- расточительность в использовании ресурсов – времени и памяти за счет необходимости использования огромной грамматики ЕЯ и емких словарей;
- сложность и трудоемкость обнаружения и исправления на последующих этапах ошибок анализа, допущенных на предыдущих этапах с учетом того, что морфологические и синтаксические ошибки в предложениях ЕЯ вполне естественны.

В отличие от этого подхода в семантически-ориентированном анализе главным и первым этапом анализа является анализ семантики (смысла), иногда, предварительный, т. к. далее смысл может уточняться с использованием уже синтаксического и морфологического анализа. В этом случае можно говорить не об анализе, а о распознавании смысла предложения.

19.2 Технологии анализа естественного языка

Под технологией анализа ЕЯ подразумевается перевод некоторого выражения на ЕЯ во внутреннее представление. Фактически все системы анализа ЕЯ могут быть распределены на следующие категории [3]:

- подбор шаблона (Pattern Matching),
- синтаксический анализ,
- семантические грамматики,
- анализ с помощью падежных фреймов,
- «жди и смотри» (Wait And See),
- словарный экспертный (Word Expert) анализ,
- коннекционистский анализ,
- «скользящий» (Skimming) анализ.

Ниже пойдет речь о некоторых наиболее распространенных методах анализа ЕЯ.

19.2.1 Подбор шаблона

Сущность данного подхода состоит в интерпретации ввода в целом, а не в интерпретации смысла и структуры его отдельных составляющих на более низком уровне. При использовании этого метода происходит сравнение уже имеющихся в системе шаблонов-образцов с текстом, поступившим на вход. Обычно шаблоны представлены в виде простого списка соответствий между классами высказываний и интерпретациями. Иногда они дополнены семантическими элементами или другими компонентами более высокого уровня [3].

Построенная таким образом, система ничего не понимает, а лишь поддерживает диалог, сравнивая реплики пациента с шаблонами и присвоенными им соответствующими ответными репликами, такими, как приведено ниже.

Шаблоны	Действия
Х всегда Y:	Вы можете привести какой-нибудь пример? Когда? Действительно, всегда?
Х Вы Y меня:	Почему Вы думаете, что я Y Вас? Вам нравится думать, что я Y Вас, не правда ли? Представьте, что я Y Вас. Что Вы об этом думаете?

При наличии шаблонов указанных выше система способна вести приведенный ниже диалог.

Человек	Люди всегда пристально смотрят на меня.	подбор по 1-му шаблону с учетом <i>всегда</i>
Система	Вы можете привести какой-нибудь пример?	первый ответ по 1-му шаблону
Человек	Например, сейчас ты смотришь на меня.	подбор по 2-му шаблону, $Y=$ смотреть
Система	Почему Вы думаете, что я <u>смотрю</u> на Вас?	первый ответ по 2-му шаблону
Человек	Потому что ты ненавидишь меня.	подбор по 2-му шаблону, $Y=$ ненавидеть
Система	Вам нравится думать, что я ненавижу Вас, не правда ли? и т.д.	второй ответ по 2-му шаблону

19.2.2 Синтаксический анализ

При использовании синтаксического анализа происходит интерпретация отдельных частей высказывания, а не всего высказывания в целом. Обычно сначала производится полный синтаксический анализ, а затем строится внутренне представление введенного текста, либо производится интерпретация.

Деревья анализа и свободно-контекстные грамматики. Большинство способов синтаксического анализа реализовано в виде деревьев. Одна из простейших разновидностей – свободно-контекстная грамматика, состоящая из правил типа $S=NP+VP$ или $VP=V+NP$ и полагающая, что левая часть правила может быть заменена на правую без учета контекста. Свободно-контекстная грамматика широко используется в машинных языках, и с ее помощью созданы высокоэффективные методы анализа [3].

Недостаток этого метода – отсутствие запрета на грамматически неправильные фразы, где, например, подлежащее не согласовано со сказуемым в числе. Для решения этой проблемы необходимо наличие двух отдельных, параллельно работающих грамматик: одной – для единственного, другой – для множественного числа. Кроме того, необходима своя грамматика для пассивных предложений и т.д. Семантически неправильное предложение может породить огромное количество вариантов разбора, из которых один будет превращен в семантическую запись. Всё это делает количество правил огромным и, в свою очередь, свободно-контекстные

грамматики непригодными для систем естественно-языкового интерфейса (СЕЯИ) [3].

Трансформационная грамматика. Трансформационная грамматика была создана с учетом упомянутых выше недостатков и более рационального использования правил ЕЯ, но оказалась непригодной для СЕЯИ. Трансформационная грамматика создавалась Хомским как порождающая, что, следовательно, делало очень затруднительным обратное действие, т.е. анализ [3].

Расширенная сеть переходов. Расширенная сеть переходов была разработана Бобровым (Bobrow), Фрейзером (Fraser) и во многом Вудсом (Woods) как продолжение идей синтаксического анализа и свободно-контекстных грамматик в частности. Она представляет собой (рис. 19.2) узлы и направленные стрелки, «расширенные» (т.е. дополненные) рядом тестов (правил), на основании которых выбирается путь для дальнейшего анализа [3].

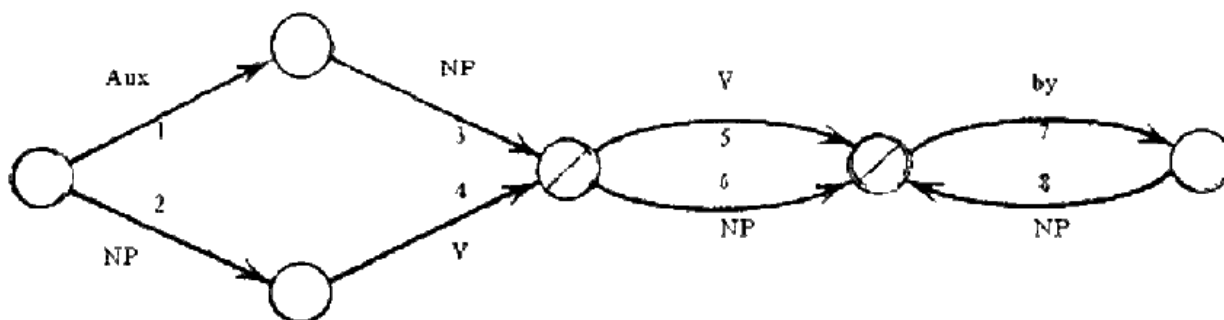


Рисунок 19.2 - Расширенная сеть переходов

Промежуточные результаты записываются в ячейки (регистры). Ниже приводится пример такой сети, позволяющей анализировать простые предложения всех типов (включая пассив), состоящие из подлежащего, сказуемого и прямого дополнения, таких, как

The rabbit nibbles the carrot (Кролик грызет морковь).

Обозначения у стрелок означают номер теста, а также либо признаки, аналогичные применяемым в свободно-контекстных грамматиках (NP), либо конкретные слова (by). Тесты написаны на языке LISP и представляют собой правила типа *если условие=истина*, то присвоить анализируемому слову признак *X* и записать его в соответствующую ячейку.

Разберем алгоритм работы сети на вышеприведенном примере. Анализ начинается слева, т.е. с первого слова в предложении. Словосочетание the rabbit проходит тест, который выясняет, что оно не является вспомогательным глаголом (*Aux*, стрелка 1), но является именной группой

(NP, стрелка 2). Поэтому `the rabbit` кладется в ячейку `Subj`, и предложение получает признак `Type_Declarative`, т.е. повествовательное, и система переходит ко второму узлу. Здесь дополнительный тест не требуется, поскольку он отсутствует в списке тестов, записанных на LISP. Следовательно, слово, стоящее после `the rabbit` – т.е. `nibbles` – глагол-сказуемое (обозначение `V` на стрелке), и `nibbles` записывается в ячейку с именем `V`. Перечеркнутый узел означает, что в нем анализ предложения может в принципе закончиться. Но в нашем примере имеется еще и дополнение `the carrot`, так что анализ продолжается по стрелке 6 (выбор между стрелками 5 и 6 осуществляется снова с помощью специального теста), и словосочетание `the carrot` кладется в ячейку с именем `Obj`. На этом анализ заканчивается (последний узел был бы использован в случае анализа такого пассивного предложения, как `The carrot was nibbled by the rabbit`). Таким образом, в результате заполнены регистры (ячейки) `Subj`, `Type`, `V` и `Obj`, используя которые, можно получить какое-либо представление (например, дерево).

Расширенная сеть переходов имеет свои недостатки [3]:

- немодульность;
- сложность при модификации, вызывающая непредвиденные побочные эффекты;
- хрупкость (когда единственная неграмматичность в предложении делает невозможным дальнейший правильный анализ);
- неэффективность при переборе с возвратами, т.к. ошибки на промежуточных стадиях анализа не сохраняются;
- неэффективность с точки зрения смысла, когда с помощью полученного синтаксического представления оказывается невозможным создать правильное семантическое представление.

19.2.3 Семантические грамматики

Анализ ЕЯ, основанный на использовании семантических грамматик, очень похож на синтаксический, с той разницей, что вместо синтаксических категорий используются семантические. Естественно, семантические грамматики работают в узких предметных областях [3].

Примером служит система `Ladder`, встроенная в базу данных американских судов. Ее грамматика содержит записи типа [3]:

```
S → <present> the <attribute> of <ship>
<present> → what is|[can you] tell me
<ship> → the <shipname>|<classname> class ship
```


Такая грамматика позволяет анализировать такие запросы, как `Can you tell me the class of the Enterprise?` (`Enterprise` – название корабля). В данной системе анализатор составляет на основе запроса пользователя запрос на языке базы данных.

Недостатки семантических грамматик состоят в том, что [3]:

- во-первых, необходима разработка отдельной грамматики для каждой предметной области,
- во-вторых, они очень быстро увеличиваются в размерах.

Способы исправления этих недостатков [3]:

- использование синтаксического анализа перед семантическим;
- применение семантических грамматик только в рамках реляционных баз данных с абстрагированием от общезыковых проблем;
- комбинация нескольких методов (включая собственно семантическую грамматику).

19.2.4 Анализ с помощью падежных фреймов

С созданием падежных фреймов связан большой скачок в развитии СЕЯИ. На сегодняшний день падежные фреймы – один из наиболее часто используемых методов СЕЯИ, т.к. он является наиболее компьютерно-эффективным при анализе как снизу вверх (от составляющих к целому), так и сверху вниз (от целого к составляющим) [3].

Падежный фрейм состоит из [3]:

- заголовка,
- набора ролей (падежей), связанных определенным образом с заголовком.

Фрейм для компьютерного анализа отличается от обычного фрейма тем, что отношения между заголовком и ролями определяется семантически, а не синтаксически, т.к. в принципе одному и то же слово может приписываться разные роли, например, существительное может быть как инструментом действия, так и его объектом [3].

Общая структура фрейма такова [3]:

[Заголовочный глагол

[падежный фрейм

агент: <активный агент, совершающий действие>

объект: <объект, над которым совершается действие>

инструмент: <инструмент, используемый при совершении действия>

реципиент: <получатель действия – часто косвенное дополнение>
направление: <цель (обычно физического) действия>
место: <место, где совершается действие>
бенефициант: <сущность, в интересах которой совершается действие>
коагент: <второй агент, помогающий совершать действие>
]]

Например, для фразы Иван дал мяч Кате падежный фрейм выглядит так:

[Давать
[падежный фрейм
агент: Иван
объект: мяч
реципиент: Катя]
[грам
время: прош
залог: акт]
]

Существуют обязательные, необязательные и запрещенные падежи. Так, для глагола разбить обязательным будет падеж объект – без него высказывание будет незаконченным. Место и коагент будут в данном примере необязательными падежами, а направление и реципиент – запрещенными.

Часто в СЕЯИ бывает полезным использовать семантическое представление в как можно более канонической форме. Наиболее известным способом такой репрезентации являются метод концептуальных зависимостей, разработанный Шенком для глаголов действия. Он заключается в том, что каждое действие представлено в виде одного или более простейших действий [3].

Например, для предложений Иван дал мяч Кате (1) и Катя взяла мяч у Ивана (2), различающихся синтаксически, но оба обозначающих акт передачи, могут быть построены следующие репрезентации с использованием простейшего действия Atrans, применяющегося в грамматике концептуальных зависимостей:

(1)	(2)
[Atrans	[Atrans
отн: обладание	отн: обладание
Агент: Иван	агент: Катя
объект: мяч	объект: мяч
источник: Иван	источник: Иван
реципиент: Катя]	реципиент: Катя]

С помощью такого представления легко выявляются сходства и различия фраз.

Для облегчения анализа также используется деление роли на [3]:

- лексический маркер,
- заполнитель.

Так, для роли объект может быть установлен маркер прямое дополнение, для роли источник – маркер вида <маркер-из>=из | от | ...

В общем случае анализ текста с помощью падежных фреймов состоит из следующих шагов [3]:

1) используя существующие фреймы, подобрать подходящий для заголовка. Если такого нет, текст не может быть проанализирован;

2) вернуть в систему подходящий фрейм с соответствующим заголовком-глаголом;

3) попытаться провести анализ по всем обязательным падежам. Если один или более обязательных заполнителей падежей не найдены, вернуть в систему код ошибки. Такой случай может означать наличие эллипсиса, неверный выбор фрейма, неверно введенный текст или недостаток грамматики.

Следующие шаги используются уже для анализа и исправления таких ситуаций;

4) провести анализ по всем необязательным падежам;

5) если после этого во введенном тексте остались непроанализированные элементы, выдать сообщение об ошибке, связанной с неправильным вводом, недостаточностью данного анализа или необходимостью провести другой, более гибкий анализ.

Преимущества использования падежных фреймов таковы [3]:

- совмещение двух стратегий анализа (сверху вниз и снизу вверх);
- комбинирование синтаксиса и семантики;
- удобство при использовании модульных программ.

19.2.5 Обработка предложений естественного языка с помощью нейронных сетей

В последнее десятилетие для обработки ЕЯ начали использовать нейронные сети.

Нейронные сети используются для решения следующих задач [3]:

- распознавание грамматически правильных предложений ЕЯ (используются рекуррентные нейронные сети);
- представление и распознавание смысла, заключенного в предложении ЕЯ (семантические нейронные сети);
- кластеризация слов и фраз ЕЯ с целью поиска документов по содержанию (модель Хэмминга, самоорганизующиеся карты Кохонена).

В первом случае происходит обучение сети на обучающей выборке из правильных предложений. При этом в процессе обучения нейронная сеть формирует в своем состоянии и своей структуре грамматику языка.

Во втором случае при обучении происходит формирование нейронных сетей, узлами которых являются слова, словосочетания и концепции, взятые из обработанных предложений ЕЯ.

В третьем случае в процессе обучения происходит разбиение предложений (фраз, слов) по степени их похожести в пространстве признаков.

В основу главы 19 положен материал учебного пособия [3].

Заключение

Прогресс в сфере экономики немыслим без применения современных информационных технологий, представляющих собой основу экономических информационных систем. Глобализация финансовых рынков, развитие средств электронной коммерции и формирование в Интернете доступных для анализа баз данных финансово-экономической информации, снижение стоимости программной реализации ИИС, привели за последние два года к беспрецедентному росту их использования в экономике. ИИС способны диагностировать состояние предприятия, оказывать помощь в антикризисном управлении, обеспечивать выбор оптимальных решений по стратегии развития предприятия и его инвестиционной деятельности. Благодаря наличию средств естественно-языкового интерфейса появляется возможность непосредственного применения ИИС бизнес-пользователем, не владеющим языками программирования, в качестве средств поддержки процессов анализа, оценки и принятия экономических решений. ИИС применяются для экономического анализа деятельности предприятия, стратегического планирования, инвестиционного анализа, оценки рисков и формирования портфеля ценных бумаг, финансового анализа, маркетинга и т.д.

Современная динамично изменяющаяся бизнес-среда требует профессионалов, способных в дополнение к экономическим знаниям применять современные информационные технологии, чтобы находить инновационные способы реализации бизнес-процессов. В связи с потребностями многократно возросшего спроса рынка труда на экономистов-аналитиков, многие университеты значительно увеличили объем часов, отводимых на изучение экспертных систем, баз знаний, дедуктивных систем, систем искусственного интеллекта с применением их в бизнесе, предпринимательстве, менеджменте. По материалам рекрутинговых агентств, в Интернете существует устойчивый высокий спрос на специалистов, владеющих современными технологиями проектирования и разработки ИИС.

Таким образом, востребованность выпускника вуза на рынке труда в современных условиях во многом определяется тем, насколько уверенно он владеет и насколько глубоко освоил самые передовые разделы современных информационных технологий и систем. Предлагаемая дисциплина дает представление студентам о состоянии разработки и тенденциях развития интеллектуальных информационных систем, их различных приложениях, что в дальнейшем позволит им стать высокооплачиваемыми и востребованными специалистами в интеллектуально-информационной сфере.

Список использованных источников

1. Луценко Е.В. Интеллектуальные информационные системы: учебное пособие для студентов специальности «прикладная информатика (по отраслям)». – Краснодар: КубГАУ, 2004. – 633 с.
2. Романов В.П. Интеллектуальные информационные системы в экономике: учебное пособие. / Под ред. д.э.н. проф. Н.П. Тихомирова. – М.: изд. «Экзамен», 2003. – 496 с.
3. Гаврилов А.В. Системы искусственного интеллекта. Методические указания для студентов. – Новосибирск: НГТУ, 2004. – 59 с.
4. Интеллектуальные информационные системы: программа дисциплины для студентов специальности «Прикладная информатика в экономике» - Ставрополь: СФ МГГУ им. М.А. Шолохова, 2008. – 22 с.
5. Искусственный интеллект. – В 3-х кн. Кн. 2. Модели и методы: справочник. / Под ред. Д.А. Поспелова – М.: Радио и связь, 1990. – 304 с.
6. Сотник С.Л. Основы проектирования систем искусственного интеллекта [Эл. ресурс]. 1998. – URL: <http://neuroschooll.narod.ru/books/sotnik.html> (дата доступа 01.02.2009).
7. Сотник С.Л. Проектирование систем искусственного интеллекта: курс лекций для Интернет-университета информационных технологий [Эл. ресурс]. – М.: Интернет-университет информационных технологий - www.INTUIT.ru. – URL: <http://www.intuit.ru/departament/expert/artintell/> (дата доступа 01.02.2009).
8. Макушкин В.А., Афонин В.Л. Интеллектуальные робототехнические системы: курс лекций для Интернет-университета информационных технологий [Эл. ресурс]. – М.: Интернет-университет информационных технологий - www.INTUIT.ru. – URL: <http://www.intuit.ru/departament/human/isrob/> (дата доступа 01.02.2009).
9. Терехов С.А. Лаборатория Искусственных нейронных сетей [Эл. ресурс]. – Снежинск: ВНИИТФ НТО-2. – URL: http://alife.narod.ru/lectures/neural/Neu_index.htm (дата доступа 01.02.2009).
10. Короткий С. Нейронные сети: основные положения. – URL: http://www.shestopaloff.ca/kyriako/Russian/Artificial_Intelligence/Some_publications/Korotky_Neuron_network_Lectures.pdf (дата доступа 01.02.2009).
11. Уоссермен Ф. Нейрокомпьютерная техника. – М.: Мир, 1992.
12. Заенцев И.В. Нейронные сети: основные модели. – Воронеж: ВГУ, 1999. – 76 с.
13. Алтунин А.Е., Семухин М.В. Модели и алгоритмы принятия решений в нечетких условиях: монография. - Тюмень: изд. Тюменского гос. университета, 2000. – 352 с.

14. Блюмин С.Л., Шуйкова И.А. Введение в математические методы принятия решений. - Липецк: изд. ЛГПИ. 1999. – 100 с.
15. Батыршин И.З. Основные операции нечеткой логики и их обобщения. – Казань: Отечество, 2001. – 100 с.
16. Рыжов А.П. Элементы теории нечетких множеств и измерения нечеткости. Изд. 2-е испр. – М.:, Диалог-МГУ, 2003. – 81 с.
17. Яхьяева Г.Э. Нечеткие множества и нейронные сети: учебное пособие. — М.: Интернет-Университет Информационных технологий; Бином. Лаборатория знаний, 2008. – 316 с.
18. Исаев С. Популярно о генетических алгоритмах. // Алгоритмы, методы, исходники [Эл. ресурс]. – URL: <http://algolist.manual.ru/ai/ga/ga1.php> (дата доступа 01.02.2009).
19. Андреев А. Электродарвин. // Парадокс [Эл. ресурс]. №3. 2004. – URL: <http://www.fuga.ru/articles/2004/03/genetic-pro.htm> (дата доступа 01.02.2009).
20. Генетические алгоритмы, искусственные нейронные сети и проблемы виртуальной реальности. / Вороновский Г.К. Махотило К.В. Петрашев С.Н. Сергеев С.А. – Х.: Основа, 1997. – 112 с.
21. Кузюрин Н.Н., Мартишин С.А., Храпченко М.В. Генетические алгоритмы в задаче поиска часто встречающихся комбинаций. // Труды Института системного программирования РАН. Т. 6, 2004. – С. 109-136. – URL: <http://cyberleninka.ru/article/n/geneticheskie-algoritmy-v-zadache-poiska-chasto-vstrechayuschihsya-kombinatsiy> (дата доступа 01.02.2009).