Министерство сельского хозяйства Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования
Иркутский Государственный Аграрный Университет им. А.А. Ежевского

Кафедра информатики и математического моделирования

БЕНДИК Н.В., ФЕДУРИНА Н.И.

Учебно-методическое пособие
«Интеллектуальные информационные системы»

Для студентов
направления подготовки 09.03.03 «Прикладная информатика»,
профиль «Прикладная информатика в экономике»

Иркутск, 2017

УДК 519.863:631.151.6

Печатается по решению научно-методического совета ФГБОУ ВО Иркутского ГАУ (протокол № от _____2017 г.).

Рецензенты:

Бендик, Н.В. Интеллектуальные информационные системы. Учебно-методическое пособие для студентов направления «Прикладная информатика» /Н.В. Бендик, Н.И. Федурина. – Иркутск: Изд-во Иркутский ГАУ, 2017. – 160 с. – ил.

В учебно-методическом пособии "Интеллектуальные информационные системы" рассматриваются способы построения информационных систем для решения неформализованных задач в различных сферах творческой деятельности человека. Особое внимание уделяется вопросам построения экспертных систем, которые являются наиболее значительным результатом практической реализации теории искусственного интеллекта. Рассматриваются процедуры имитации мыслительной деятельности человека в определенной предметной области, алгоритмы выделения признаков для описания ситуаций в условиях неопределенности. Изучаются математические и алгоритмические основы интеллектуальных информационных систем: модели представления знаний на основе систем продукций, семантических сетей и фреймов; выводы на знаниях; нечеткая информация и выводы; нейронные сети; методы эвристического поиска решений и программирования задач.

Пособие содержит краткие сведения по теории интеллектуальных систем, задания на лабораторные работы и порядок их выполнения.

Содержание

Введение

Интеллектуальные информационные системы проникают во все сферы нашей жизни, поэтому становятся неотъемлемым элементом при решении задач автоматизации и управления сложными объектами и процессами. Современное понятие интеллектуальных информационных систем (ИИС) сформировалось в процессе развития теоретических основ кибернетики, современной теории управления, теории алгоритмов, развития современных информационных технологий и обобщения накопленных научных знаний, методов и средств в области искусственного интеллекта (ИИ). Целью пособия является обзорное ознакомление студентов, обучающихся по направлению «Прикладная информатика» и другим родственным направлениям, с проблематикой и областями использования искусственного интеллекта в автоматизированных системах обработки информации и управления, освещение теоретических и организационно-методических вопросов построения и функционирования систем, основанных на знаниях, привитие навыков практических работ по проектированию баз знаний. В результате изучения лекционного материала студенты получат знания по архитектуре и классификации ИС, методам представления знаний, областям применения, а также научатся выбирать адекватные проблемной области методы проектировании базы знаний.

Учебно-методическое пособие «Интеллектуальные информационные системы» предназначены для подготовки студентов направления «Прикладная информатика». По своему содержанию соответствует типовой программе данного направления. В пособие включено 8 лабораторных работ. Выполнение каждой из них рассчитано на 4 часа.

Описание каждой работы включает:

- теоретическая часть;
- порядок выполнения работы;
- варианты заданий;
- контрольные вопросы.

Студент по каждой работе обязан представить отчет и ответить на контрольные вопросы.

Требования к содержанию отчета

Отчет должен содержать:

- название темы варианта;
- цель работы;
- теоретическую часть;
- схему;
- ответы на контрольные вопросы.

I
Теоретические основы интеллектуальных информационных систем

1 Введение в интеллектуальные информационные технологии

В основе стратегии интеллектуальных технологий лежит понятие парадигмы - концептуального представления на суть проблемы или задачи и принцип ее решения. Центральная парадигма интеллектуальных технологий - это обработка знаний. Системы, ядром которых является база знаний или модель предметной области, описанная на языке сверхвысокого уровня, приближенном к естественному языку, называют интеллектуальными. Чаще всего интеллектуальные системы применяются для решения сложных задач, связанных с использованием слабо формализованных знаний специалистов - практиков, а также с логической обработкой информации. Например, поддержка принятия решения в сложных ситуациях, анализ визуальной информации, управление в электрических цепях электрооборудования и сетях распределения электроэнергии; поиск неисправностей в электронных устройствах, диагностика отказов контрольно-измерительного оборудования и т. д. Типичными примерами ИИС являются экспертные системы (ЭС) и искусственные нейронные сети (ИНС), берущие на себя решение вопросов извлечения и структурирования знаний, а также технологические аспекты разработки систем, основанных на знаниях.

Экспертные системы – это быстро прогрессирующее направление в области искусственного интеллекта. Современные ЭС представляют собой сложные программные комплексы, аккумулирующие знания специалистов в конкретных предметных областях и распространяющие этот эмпирический опыт для консультирования менее квалифицированных пользователей. Парадигма ЭС предполагает следующие объекты, а также этапы разработки и функционирования ИС:

- формализация знаний – преобразование экспертом проблемного знания в форму, предписанную выбранной моделью представления знаний;
- формирование базы знаний (БЗ) – вложение формализованных знаний в программную систему;
- дедукция – решение задачи логического вывода на основе БЗ.

Основные факторы, влияющие на целесообразность и эффективность разработки ЭС:

- нехватка специалистов, затрачивающих значительное время для оказания помощи другим;
- выполнение небольшой задачи требует многочисленного коллектива специалистов, поскольку ни один из них не обладает достаточным знанием;
- сниженная производительность, поскольку задача требует полного

анализа сложного набора условий, а обычный специалист не в состоянии просмотреть за отведенное время все эти условия;

- большое расхождение между решениями самых хороших и самых плохих исполнителей;
- большое расхождение между решениями самых хороших и самых плохих исполнителей;
- наличие экспертов, готовых поделиться своим опытом.

Сравнительные свойства прикладных задач для их решения ЭС приведены в таблице 1.

Таблица 1 Критерии применимости ЭС

Применимы ЭС	Неприменимы ЭС
Не могут быть построены строгие алгоритмы или процедуры, но существуют эвристические методы решения.	Имеются эффективные алгоритмические методы.
Есть эксперты, которые способны решить задачу.	Отсутствуют эксперты или их число недостаточно.
По своему характеру задачи относятся к области диагностики, интерпретации или прогнозирования	Задачи носят вычислительный характер.
Доступные данные “зашумленны”.	Известны точные факты и строгие процедуры.
Задачи решаются методом формальных рассуждений.	Задачи решаются процедурными методами, с помощью аналогии или интуитивно.
Знания статичны, неизменны.	Знания динамичны меняются со временем.

Недостатки экспертных систем перед человеком-экспертом:

- экспертная система может быть не пригодна для применения пользователем, если у него нет опыта работы с такими системами;
- вопросно-ответный режим, обычно принятый в таких системах, замедляет получение решений;
- существует проблема приведения знаний, полученных от эксперта, к виду, обеспечивающему их эффективную машинную реализацию;
- человек-эксперт при решении задач обычно обращается к своей интуиции или здравому смыслу, если отсутствуют формальные методы решения или аналоги таких задач.

Достоинства экспертных систем перед человеком- экспертом:

- у них нет предубеждений, они не делают поспешных выводов;

- введенные в машину знания сохраняются, человек же имеет ограниченную базу знаний, и если данные долгое время не используются, то они забываются и навсегда теряются;
- эксперт пользуется побочными знаниями и легко поддается влиянию внешних факторов, которые непосредственно не связаны с решаемой задачей.

Главное отличие ЭС от других программных средств - это наличие базы знаний, в которой знания хранятся в форме, понятной специалистам предметной области, и могут быть изменены и дополнены также в понятной форме. Это и есть языки представления знаний (ЯПЗ).

В России в исследования и разработку ЭС большой вклад внесли работы Д. А. Поспелова (основателя Российской ассоциации искусственного интеллекта и его первого президента), Э. В. Попова, В. Ф. Хорошевского, В. Л. Стефанюка, Г. С. Осипова, В. К. Финна, В. Л. Вагина, В. И. Городецкого и многих других. Современное состояние разработок в области ЭС в России можно охарактеризовать как стадию все возрастающего интереса среди широких слоев специалистов - менеджеров, инженеров, программистов и других. Наибольшие трудности в разработке ЭС вызывает не процесс машинной реализации систем, а этап анализа знаний и проектирования базы знаний. Этим занимается специальная наука - инженерия знаний.

Экспертные системы имеют две категории пользователей и два отдельных входа, соответствующих различным целям взаимодействия пользователей с ЭС. К первой категории относятся обычные пользователи, которым требуется консультация ЭС. Вторую категорию представляют эксперты в предметной области и инженеры знаний. В их функции входит заполнение базы знаний с помощью специализированной диалоговой компоненты ЭС - подсистемы приобретения знаний. Подсистема приобретения знаний предназначена для добавления в базу знаний новых правил и модификации имеющихся. В ее задачу входит приведение правила к виду, позволяющему подсистеме вывода применять это правило в процессе работы. В более сложных системах предусмотрены еще и средства для проверки вводимых или модифицируемых правил на непротиворечивость с имеющимися правилами. Диалог с ЭС осуществляется через диалоговый процессор - специальную компоненту ЭС. Существуют две основные формы диалога с ЭС - это диалог на ограниченном подмножестве естественного языка с использованием словаря (меню) и диалог на основе из нескольких возможных действий. База знаний представляет наиболее важную компоненту экспертной системы. В отличие от всех остальных компонент ЭС, база знаний – есть «переменная» часть системы, которая может пополняться и модифицироваться инженерами знаний и опыта использования ЭС между консультациями, а в некоторых системах и в процессе консультации. Существует несколько способов представления знаний в ЭС. Общим для всех способов является то, что знания представлены в символьной форме (тексты, списки и другие символьные структуры). Тем самым, в ЭС реализуется принцип символьной природы рассуждений, который заключается в том, что

процесс рассуждения представляется как последовательность символьных преобразований. Подсистема вывода - программная компонента экспертных систем, реализующая процесс ее рассуждений на основе базы знаний и рабочего множества. Она выполняет две функции: во-первых, просмотр существующих фактов из рабочего множества и правил из базы знаний и добавление (по мере возможности) в рабочее множество новых фактов и, во-вторых, определение порядка просмотра и применения правил. Эта подсистема управляет процессом консультации, сохраняет для пользователя информацию о полученных заключениях, и запрашивает у него информацию, когда для срабатывания очередного правила в рабочем множестве оказывается недостаточно данных. Цель ЭС - вывести некоторый заданный факт, который называется целевым утверждением. В результате применения правил добиться того, чтобы этот факт был включен в рабочее множество, либо опровергнуть этот факт, то есть убедиться, что его вывести невозможно. Целевое утверждение может быть либо «заложено» заранее в базу знаний системы, либо извлекается системой из диалога с пользователем. Работа системы представляет собой последовательность шагов, на каждом из которых из базы выбирается некоторое правило, которое применяется к текущему содержимому рабочего множества. Цикл заканчивается, когда выведено либо опровергнуто целевое утверждение. Цикл работы экспертной системы иначе называется логическим выводом.

Другим актуальным направлением разработки ИИС является создание интеллектуальных нейронных сетей (ИНС). Характер разработок в области ИНС принципиально отличается от ЭС. В основе нейронных сетей лежит преимущественно поведенческий подход к решаемой задаче: сеть «учится на примерах» и подстраивает свои параметры при помощи так называемых алгоритмов обучения через механизм обратной связи. Парадигма ученика включает следующие положения и последовательность действий:

- формирования базы данных на основе обработки наблюдений и изучения опыта частных примеров;
- индуктивное обучение - изучение аппроксимирующих, вероятностных и логических механизмов получения общих выводов из частных утверждений, то есть превращение базы данных (БД) в базу знаний (БЗ) и обоснование процедуры извлечения знаний из БЗ;
- дедукция - выбор информации из БЗ на основе обоснованной или предполагаемой процедуре.

В рамках этой парадигмы самообучающиеся системы являются менее изученными, чем экспертные системы. Искусственные нейронные сети индуцированы биологией, так как они состоят из элементов, функциональные возможности которых аналогичны большинству элементарных функций биологического нейрона. Нейрон реализует достаточно простую передаточную функцию, позволяющую преобразовать возбуждения на входах, с учетом весов входов, в значение возбуждения на выходе нейрона. Функционально законченный фрагмент мозга имеет входной слой нейронов –

рецепторов, возбуждаемых извне, и выходной слой, нейроны которого возбуждаются в зависимости от конфигурации и возбуждения нейронов входного слоя. Распределение величин возбуждения нейронов выходного слоя, чаще всего поиск нейрона, обладающего максимальной величиной возбуждения, позволяет установить соответствие между комбинацией и величинами возбуждений на входном слое. Эта зависимость определяет возможность логического вывода вида «если - то». Управление и формирование данной зависимости осуществляется весами синаптических связей нейронов, которые влияют на направление распространения возбуждения нейронов в сети, приводящие на этапе обучения к «нужным» нейронам выходного слоя. Отсюда следует, что сеть работает в двух режимах: в режиме обучения и в режиме распознавания (рабочем режиме). В режиме обучения производится формирование логических цепочек. В режиме распознавания нейронная сеть по предъявленному образу с высокой достоверностью определяет, к какому типу он относится, какие действия следует предпринять и т.д. Следовательно, под ИНС следует понимать системы, параметры, которых могут изменяться в процессе обучения или самообучения, исходя из накопленного опыта обобщающего предыдущие прецеденты на новые случаи и извлекающего существенные свойства из поступающей информации. Нейронные сети применяются для решения трудно формализуемых задач, в которых информация об объекте является неполной, неточной или нечеткой. Кроме того, связь между входными и выходными параметрами может быть настолько сложна, что моделирование в традиционном смысле становится малоэффективным, а порой просто невозможным. Примеры эффективного применения ИНС являются задачи управления, распознавания образов, анализа данных, моделирования и прогнозирования.

Основные сведения из истории создания ИС

4 октября 1939 г. по решению суда изобретателем первого цифрового электронного компьютера признан Джон Винсент Атанасов и его ассистент Клиффорд Берри (Университет штата Айова). Половинчатое признание первенства Атанасова является следствием скандального судебного решения. По этому решению первые компьютерные инженеры Джон Мочли и Джон Эккерт лишились права на патент, полученный ими в 1964 году, и права называться изобретателями электронно-цифрового компьютера. Однако именно они после нескольких экспериментальных моделей создали в 1945 году в Университете Пенсильвании более известный компьютер ENIAC, с которого началось развитие индустрии.

В 1945 г. построены Вальтером Питтсом и Уорреном МакКуллочем нейронные сети с обратной связью. Примерно в то же время Норберт Винер создал область кибернетики, которая включала математическую теорию обратной связи для биологических и инженерных систем. Важным аспектом данного открытия стала концепция о том, что разум - это процесс получения и обработки информации для достижения определенной цели.

В 1949 г. Дональд Хеббс открыл способ создания самообучающихся искусственных нейронных сетей. Этот процесс, позволяет изменять весовые коэффициенты в нейронной сети так, что данные на выходе отражают связь с информацией на входе.

1950-е гг. отмечены в истории как годы рождения искусственного интеллекта. Алан Тьюринг предложил специальный тест в качестве способа распознать разумность машины. В этом тесте один или несколько людей должны задавать вопросы двум тайным собеседникам и на основании ответов определять, кто из них машина, а кто человек. Если не удавалось раскрыть машину, которая маскировалась под человека, предполагалось, что машина разумна. В 1950-е гг. были также разработаны два языка ИИ. Первый, язык IPL, был создан Ньюэллом, Симоном и Шоу для программы Logic Theorist. IPL являлся языком обработки списка данных и привел к созданию более известного языка LISP. LISP появился в конце 1950-х и вскоре заменил IPL, став основным языком приложений ИИ. Язык LISP был разработан в лабораториях Массачусетского технологического института (MIT). Его автором был Джон МакКарти, один из первых разработчиков ИИ.

В 1960-е гг. наиболее важным было представление знаний. Были построены игрушечные миры. С помощью этих миров создавалась окружающая среда, в которой тестировались идеи по компьютерному зрению, роботехнике и обработке человеческого языка

В начале 1970-х гг. впервые была применена на практике Лотфи Заде нечеткая логика для управления процессами. В 1970-х продолжалось создание языков для ИИ. Был разработан язык ПРОЛ0Г. Язык ПРОЛОГ предназначался для разработки программ, которые управляли символами и работал с правилами и фактами. В то время как ПРОЛОГ распространился за пределами США, язык LISP сохранял свой статус основного языка для приложений ИИ.

1980-е гг. отмечены ростом числа разработок и продаж экспертных систем на языке LISP, которые становились лучше и дешевле. Экспертные системы использовались многими компаниями для разработки полезных ископаемых, прогнозирования и инвестиций. Также были идентифицированы ограничения в работе экспертных систем, поскольку их знания становились все больше и сложнее. Нейронные сети в эти годы также нашли применение при решении ряда различных задач, таких как распознавание речи и возможность самообучения машин.

1990-е гг. стали новой эпохой в развитии приложений ИИ. Элементы ИИ были интегрированы в ряд приложений, такие как системы распознавания фальшивых кредитных карт; системы распознавания лиц; системы автоматического планирования; системы предсказания прибыли и потребности в персонале; конфигурируемые системы «добычи данных» из баз данных; системы персонализации и другие.

Вопросы для самопроверки

1. Какие две парадигмы лежат в основе создания современных ИС, что их объединяет и в чем существует их различие?

2. Дайте определение и поясните понятия искусственного интеллекта и интеллектуальной информационной системы? В чем эти понятия расходятся?

3. Укажите основные блоки обобщенной структурной схемы экспертной системы и поясните их назначение.

4. В чем заключаются преимущества и недостатки экспертных систем по сравнению с человеком – экспертом?

5. Поясните цикл работы экспертной системы?

6. Что такое нейронная сеть? В чем состоит парадигма «ученика»?

7. Укажите знаменательные даты в истории создания ИС.

2 Основные направления, функции и классификация ИИС

Области применения ИИС. Интеллектуальные информационные системы проникают во все сферы жизни, поэтому трудно провести строгую классификацию направлений, по которым ведутся активные и многочисленные исследования в области ИИ. Рассмотрим некоторые из них.

1. Разработка интеллектуальных информационных систем или систем, основанных на знаниях. Это одно из главных направлений ИИ. Основной целью построения таких систем являются выявление, исследование и применение знаний высококвалифицированных экспертов для решения сложных задач, возникающих на практике. При построении систем, основанных на знаниях (СОЗ), используются знания, накопленные экспертами в виде конкретных правил решения тех или иных задач. Это направление преследует цель имитации человеческого искусства анализа неструктурированных и слабоструктурированных проблем. В данной области исследований осуществляется разработка моделей представления, извлечения и структурирования знаний, а также изучаются проблемы создания баз знаний (БЗ), образующих ядро СОЗ. Частным случаем СОЗ являются экспертные системы (ЭС).

2. Разработка естественно-языковых интерфейсов и машинный перевод. Проблемы компьютерной лингвистики и машинного перевода разрабатываются в ИИ с 1950-х гг. Системы машинного перевода с одного естественного языка на другой обеспечивают быстроту и систематичность доступа к информации, оперативность и единообразие перевода больших потоков, как правило, научно-технических текстов. Системы машинного перевода строятся как интеллектуальные системы, поскольку в их основе лежат БЗ в определенной предметной области и сложные модели, обеспечивающие дополнительную трансляцию «исходный язык оригинала - язык смысла - язык перевода». Они базируются на структурно-логическом подходе, включающем последовательный анализ и синтез естественно-языковых сообщений. Кроме того, в них осуществляется ассоциативный поиск аналогичных фрагментов текста и их переводов в специальных базах данных (БД). Данное направление охватывает также исследования методов и

разработку систем, обеспечивающих реализацию процесса общения человека с компьютером на естественном языке.

3. Генерация и распознавание речи. Системы речевого общения создаются в целях повышения скорости ввода информации в ЭВМ, разгрузки зрения и рук, а также для реализации речевого общения на значительном расстоянии.

4. Обработка визуальной информации. В этом научном направлении решаются задачи обработки, анализа и синтеза изображений. Задача обработки изображений связана с трансформированием графических образов, результатом которого являются новые изображения. В задаче анализа исходные изображения преобразуются в данные другого типа, например, в текстовые описания. При синтезе изображений на вход системы поступает алгоритм построения изображения, а выходными данными являются графические объекты.

5. Обучение и самообучение. Эта актуальная область ИИ включает модели, методы и алгоритмы, ориентированные на автоматическое накопление и формирование знаний с использованием процедур анализа и обобщения данных. К данному направлению относятся не так давно появившиеся системы добычи данных (Data-mining) и системы поиска закономерностей в компьютерных базах данных (Knowledge Discovery).

6. Распознавание образов. Это одно из самых ранних направлений ИИ, в котором распознавание объектов осуществляется на основании применения специального математического аппарата, обеспечивающего отнесение объектов к классам, а классы описываются совокупностями определенных значений признаков.

7. Игры и машинное творчество. Машинное творчество охватывает сочинение компьютерной музыки, стихов, интеллектуальные системы для изобретения новых объектов. Создание интеллектуальных компьютерных игр является одним из самых развитых коммерческих направлений в сфере разработки программного обеспечения. Кроме того, компьютерные игры предоставляют мощный арсенал разнообразных средств, используемых для обучения.

8. Программное обеспечение систем ИИ. Инструментальные средства для разработки интеллектуальных систем включают в себя:

- специальные языки программирования, ориентированные на обработку символьной информации (LISP, SMALLTALK, РЕФАЛ);
- языки логического программирования **(**PROLOG**);**
- языки представления знаний (OPS 5, KRL, FRL);
- интегрированные программные среды, содержащие арсенал инструментальных средств создания систем ИИ (КЕ, ARTS, GURU, G2);
- оболочки экспертных систем (BUILD, EMYCIN, EXSYS Professional, Эксперт), которые позволяют создавать прикладные ЭС, не прибегая к программированию.

9. Новые архитектуры компьютеров. Это направление связано с созданием компьютеров не фон-неймановской архитектуры, ориентированных на обработку символьной информации. Известны удачные промышленные решения параллельных и векторных компьютеров, однако в настоящее время они имеют весьма высокую стоимость, а также недостаточную совместимость с существующими вычислительными средствами.

10. Интеллектуальные роботы. Создание интеллектуальных роботов составляет конечную цель робототехники. В настоящее время в основном используются программируемые манипуляторы с жесткой схемой управления, названные роботами первого поколения. Несмотря на очевидные успехи отдельных разработок, эра интеллектуальных автономных роботов пока не наступила. Основными сдерживающими факторами в разработке автономных роботов являются нерешенные проблемы в области интерпретации знаний, машинного зрения, адекватного хранения и обработки трехмерной визуальной информации.

Признаки классификации ИИС. Интеллектуальная информационная система основана на концепции использования базы знаний для генерации алгоритмов решения прикладных задач различных классов в зависимости от конкретных информационных потребностей пользователей.

Для ИИС характерны следующие признаки:

- развитые коммуникативные способности;
- умение решать сложные плохо формализуемые задачи;
- способность к самообучению;
- адаптивность.

Каждому из перечисленных признаков условно соответствует свой класс ИИС. Различные системы могут обладать одним или несколькими признаками интеллектуальности с различной степенью проявления.

Средства ИИ могут использоваться для реализации различных функций, выполняемых ИИС. На рисунке 1 приведена классификация ИИС, признаками которой являются следующие интеллектуальные функции:

- коммуникативные способности - способ взаимодействия конечного пользователя с системой;
- решение сложных плохо формализуемых задач, которые требуют построения оригинального алгоритма решения в зависимости от конкретной ситуации, характеризующейся неопределенностью и динамичностью исходных данных и знаний;
- способность к самообучению - умение системы автоматически извлекать знания из накопленного опыта и применять их для решения задач;
- адаптивность - способность системы к развитию в соответствии с объективными изменениями области знаний.

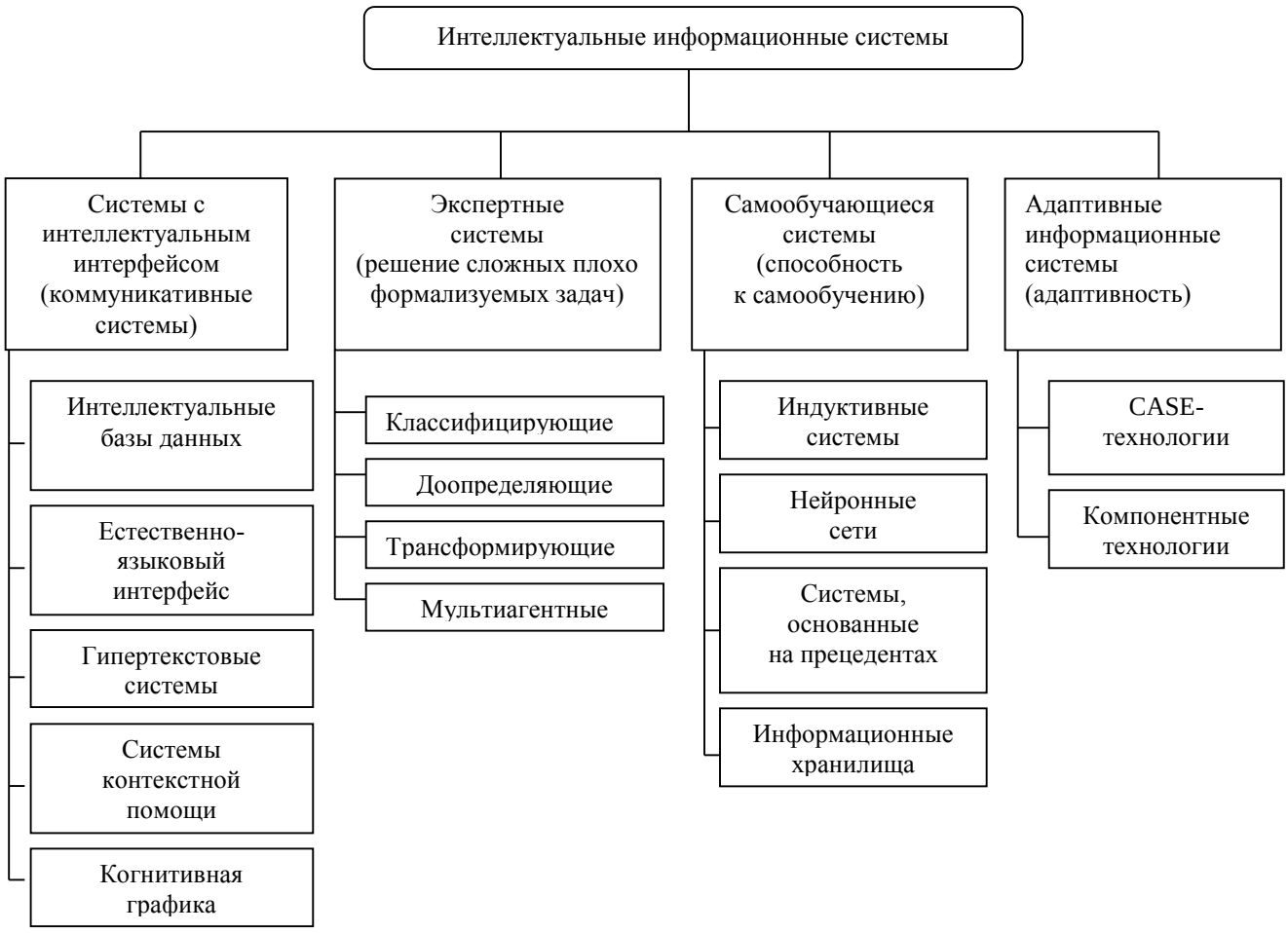


Рисунок 1 - Классификация интеллектуальных систем

Системы с интеллектуальным интерфейсом. Применение ИИС для усиления коммуникативных способностей информационных систем привело к появлению систем с интеллектуальным интерфейсом.

Среди них можно выделить следующие типы:

Интеллектуальные базы данных позволяют в отличие от традиционных БД обеспечивать выборку необходимой информации, не присутствующей в явном виде, а выводимой из совокупности хранимых данных.

1. Естественно-языковой (ЕЯ) интерфейс применяется для доступа к интеллектуальным базам данных, контекстного поиска документальной текстовой информации, голосового ввода команд в системах управления, машинного перевода с иностранных языков. Для реализации ЕЯ - интерфейса необходимо решить проблемы морфологического, синтаксического и семантического анализа, а также задачу синтеза высказываний на естественном языке. При морфологическом анализе осуществляются распознавание и проверка правильности написания слов в словаре. Синтаксический контроль предполагает разложение входных сообщений на отдельные компоненты, проверку соответствия грамматическим правилам внутреннего представления знаний и выявление недостающих частей.

Семантический анализ обеспечивает установление смысловой правильности синтаксических конструкций. В отличие от анализа синтез высказываний заключается в преобразовании цифрового представления информации в представление на естественном языке.

2. Гипертекстовые системы используются для реализации поиска по ключевым словам в базах данных с текстовой информацией. Для более полного отражения различных смысловых отношений терминов требуется сложная семантическая организация ключевых слов. Решение этих задач осуществляется с помощью интеллектуальных гипертекстовых систем, в которых механизм поиска сначала работает с базой знаний ключевых слов, а затем - с самим текстом. Аналогичным образом проводится поиск мультимедийной информации, включающей кроме текста графическую информацию, аудио - и видео образы.

3. Системы контекстной помощи относятся к классу систем распространения знаний. Такие системы являются, как правило, приложениями к документации. Системы контекстной помощи - частный случай гипертекстовых и ЕЯ-систем. В них пользователь описывает проблему, а система на основе дополнительного диалога конкретизирует ее и выполняет поиск относящихся к ситуации рекомендаций. В обычных гипертекстовых системах, наоборот, компьютерные приложения навязывают пользователю схему поиска требуемой информации.

4. Системы когнитивной графики ориентированы на общение с пользователем ИИС посредством графических образов, которые генерируются в соответствии с изменениями параметров моделируемых или наблюдаемых процессов. Когнитивная графика позволяет в наглядном и выразительном виде представить множество параметров, характеризующих изучаемое явление, освобождает пользователя от анализа тривиальных ситуаций, способствует быстрому освоению программных средств и повышению конкурентоспособности разрабатываемых ИИС. Применение когнитивной графики особенно актуально в системах мониторинга и оперативного управления, в обучающих и тренажерных системах, в оперативных системах принятия решений, работающих в режиме реального времени.

Экспертные системы. Экспертные системы как самостоятельное направление в искусственном интеллекте сформировалось в конце 1970-х гг. Группа по экспертным системам при Комитете British Computer Society определила ЭС как «воплощение в ЭВМ компоненты опыта эксперта, основанной на знаниях, в такой форме, что машина может дать интеллектуальный совет или принять решение относительно обрабатываемой функции». Одним из важных свойств ЭС является способность объяснить ход своих рассуждений понятным для пользователя образом.

Область исследования ЭС называют «инженерией знаний». Этот термин был введен Е. Фейгенбаумом и в его трактовке означает «привнесение принципов и инструментария из области искусственного интеллекта в

решение трудных прикладных проблем, требующих знаний экспертов». Другими словами, ЭС применяются для решения неформализованных проблем, к которым относят задачи, обладающие одной (или несколькими) из следующих характеристик:

- задачи не могут быть представлены в числовой форме;
- исходные данные и знания о предметной области обладают неоднозначностью, неточностью, противоречивостью;
- цели нельзя выразить с помощью четко определенной целевой функции;
- не существует однозначного алгоритмического решения задачи;
- алгоритмическое решение существует, но его нельзя использовать по причине большой размерности пространства решений и ограничений на ресурсы (времени, памяти).

Главное отличие ЭС и систем искусственного интеллекта от систем обработки данных состоит в том, что в них используется символьный, а не числовой способ представления данных, а в качестве методов обработки информации применяются процедуры логического вывода и эвристического поиска решений.

ЭС охватывают самые разные предметные области (рис. 2), среди которых лидируют бизнес, производство, медицина, проектирование и системы управления. Во многих случаях ЭС являются инструментом, усиливающим интеллектуальные способности эксперта. Кроме того, ЭС может выступать в роли:

- консультанта для неопытных или непрофессиональных пользователей;
- ассистента эксперта-человека в процессах анализа вариантов решений;
- партнера эксперта в процессе решения задач, требующих привлечения знаний из разных предметных областей.

Для классификации ЭС используются следующие признаки:

- способ формирования решения;
- способ учета временного признака;
- вид используемых данных и знаний;
- число используемых источников знаний.
- партнера эксперта в процессе решения задач, требующих привлечения знаний из разных предметных областей.

Для классификации ЭС используются следующие признаки:

- способ формирования решения;
- способ учета временного признака;
- вид используемых данных и знаний;
- число используемых источников знаний.

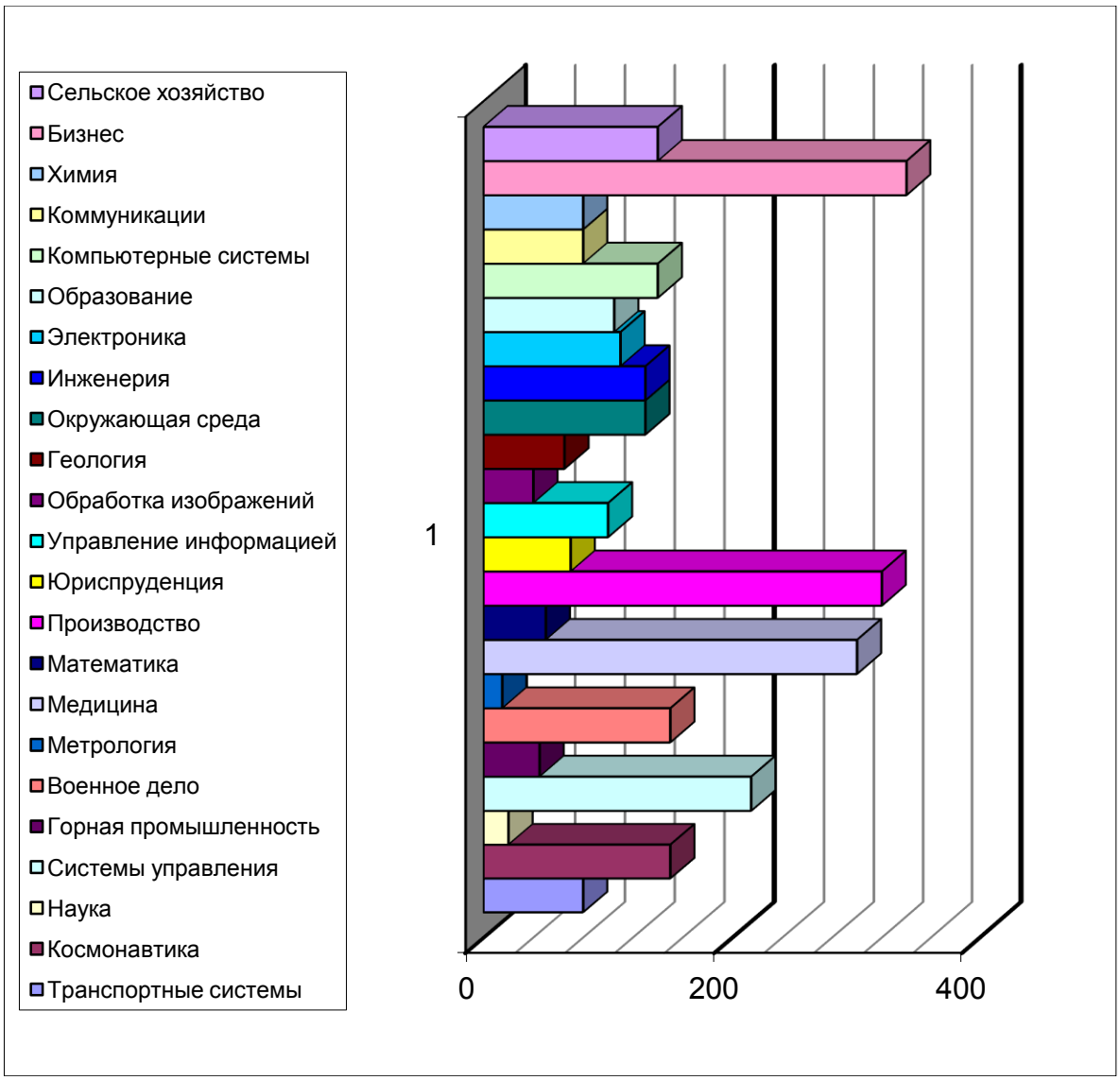


Рисунок 2 - Области применения экспертных систем

По способу формирования решения ЭС можно разделить на анализирующие и синтезирующие. В системах первого типа осуществляется выбор решения из множества известных решений на основе анализа знаний, в системах второго типа решение синтезируется из отдельных фрагментов знаний.

В зависимости от способа учета временного признака ЭС делят на статические и динамические. Статические ЭС предназначены для решения задач с неизменяемыми в процессе решения данными и знаниями, а динамические ЭС допускают такие изменения.

По видам используемых данных и знаний различают ЭС с детерминированными и неопределенными знаниями. Под неопределенностью знаний и данных понимаются их неполнота, ненадежность, нечеткость.

ЭС могут создаваться с использованием одного или нескольких источников знаний.

В соответствии с перечисленными признаками можно выделить четыре основных класса ЭС (рис. 3): классифицирующие, доопределяющие, трансформирующие и мультиагентные.

<table>
<tr><td colspan="2"></td><td>Анализ</td><td>Синтез</td><td></td></tr>
<tr><td rowspan="2">Детерминированность знаний</td><td rowspan="2">Классифици-рующие</td><td rowspan="2">Трансфор-мирующие</td><td rowspan="2">Один источник</td></tr>
<tr></tr>
<tr><td rowspan="2">Неопределённость знаний</td><td rowspan="2">Доопределяющие</td><td rowspan="2">Мультиагентные</td><td rowspan="2">Несколько источников знаний</td></tr>
<tr></tr>
<tr><td colspan="2">Статика</td><td>Динамика</td><td></td></tr>
</table>

Рисунок 3 - Основные классы экспертных систем

Классифицирующие ЭС решают задачи распознавания ситуаций. Основным методом формирования решений в таких системах является дедуктивный логический вывод.

Доопределяющие ЭС используются для решения задач с не полностью определенными данными и знаниями. В таких ЭС возникают задачи интерпретации нечетких знаний и выбора альтернативных направлений поиска в пространстве возможных решений. В качестве методов обработки неопределенных знаний могут использоваться байесовский вероятностный подход, коэффициенты уверенности, нечеткая логика.

Трансформирующие ЭС относятся к синтезирующим динамическим экспертным системам, в которых предполагается повторяющееся преобразование знаний в процессе решения задач. В ЭС данного класса используются различные способы обработки знаний:

- генерация и проверка гипотез;
- логика предположений и умолчаний (когда по неполным данным формируются представления об объектах определенного класса, которые впоследствии адаптируются к конкретным условиям изменяющихся ситуаций);
- использование метазнаний, более общих закономерностей для устранения неопределенностей в ситуациях.

Мультиагентные системы - это динамические ЭС, основанные на интеграции нескольких разнородных источников знаний. Эти источники обмениваются между собой получаемыми результатами в ходе решения задач. Системы данного класса имеют следующие возможности:

- реализация альтернативных рассуждений на основе использования различных источников знаний и механизма устранения противоречий;
- распределенное решение проблем, декомпозируемых на параллельно решаемые подзадачи с самостоятельными источниками знаний;

- применение различных стратегий вывода заключений в зависимости от типа решаемой проблемы;
- обработка больших массивов информации из баз данных;
- использование математических моделей и внешних процедур для имитации развития ситуаций.

Самообучающиеся системы. Самообучающиеся интеллектуальные системы основаны на методах автоматической классификации ситуаций из реальной практики, или на методах обучения на примерах. Примеры реальных ситуаций составляют так называемую обучающую выборку, которая формируется в течение определенного исторического периода. Элементы обучающей выборки описываются множеством классификационных признаков. Стратегия «обучения с учителем» предполагает задание специалистом для каждого примера значений признаков, показывающих его принадлежность к определенному классу ситуаций. При обучении «без учителя» система должна самостоятельно выделять классы ситуаций по степени близости значений классификационных признаков. В процессе обучения проводится автоматическое построение обобщающих правил или функций, описывающих принадлежность ситуаций к классам, которыми система впоследствии будет пользоваться при интерпретации незнакомых ситуаций. Из обобщающих правил, в свою очередь, автоматически формируется база знаний, которая периодически корректируется по мере накопления информации об анализируемых ситуациях.

Построенные в соответствии с этими принципами самообучающиеся системы имеют следующие недостатки:

- относительно низкую адекватность баз знаний возникающим реальным проблемам из-за неполноты и/или зашумленности обучающей выборки;
- низкую степень объяснимости полученных результатов;
- поверхностное описание проблемной области и узкую направленность применения из-за ограничений в размерности признакового пространства.

Индуктивные системы позволяют обобщать примеры на основе принципа индукции «от частного к общему». Процедура обобщения сводится к классификации примеров по значимым признакам. Алгоритм классификации примеров включает следующие основные шаги.

1. Выбор классификационного признака из заданного множества.
2. Разбиение множества примеров на подмножества по значению выбранного признака.
3. Проверка принадлежности каждого подмножества примеров одному из классов.
4. Проверка окончания процесса классификации. Если какое-то подмножество примеров принадлежит одному подклассу, то есть у всех примеров этого подмножества совпадает значение классификационного признака, то процесс классификации заканчивается.

5. Для подмножеств примеров с несовпадающими значениями классификационных признаков процесс распознавания продолжается, начиная с первого шага. При этом каждое подмножество примеров становится классифицируемым множеством.

Нейронные сети представляют собой классический пример технологии, основанной на примерах. Нейронные сети - обобщенное название группы математических алгоритмов, обладающих способностью обучаться на примерах, «узнавая» впоследствии черты встреченных образцов и ситуаций. Благодаря этой способности нейронные сети используются при решении задач обработки сигналов и изображений, распознавания образов, а также для прогнозирования.

Нейронная сеть - это кибернетическая модель нервной системы, которая представляет собой совокупность большого числа сравнительно простых элементов – нейронов, топология соединения которых зависит от типа сети. Чтобы создать нейронную сеть для решения какой-либо конкретной задачи, следует выбрать способ соединения нейронов друг с другом и подобрать значения параметров межнейронных соединений.

В системах, основанных на прецедентах, БЗ содержит описания конкретных ситуаций (прецеденты). Поиск решения осуществляется на основе аналогий и включает следующие этапы:

- получение информации о текущей проблеме;
- сопоставление полученной информации со значениями признаков прецедентов из базы знаний;
- выбор прецедента из базы знаний, наиболее близкого к рассматриваемой проблеме;
- адаптация выбранного прецедента к текущей проблеме;
- проверка корректности каждого полученного решения;
- занесение детальной информации о полученном решении в БЗ.

Прецеденты описываются множеством признаков, по которым строятся индексы быстрого поиска. Однако в системах, основанных на прецедентах, в отличие от индуктивных систем допускается нечеткий поиск с получением множества допустимых альтернатив, каждая из которых оценивается некоторым коэффициентом уверенности. Наиболее эффективные решения адаптируются к реальным ситуациям с помощью специальных алгоритмов. Системы, основанные на прецедентах, применяются для распространения знаний и в системах контекстной помощи.

Информационные хранилища отличаются от интеллектуальных баз данных тем, что представляют собой хранилища значимой информации, регулярно извлекаемой из оперативных баз данных. Хранилище данных - это предметно-ориентированное, интегрированное, привязанное ко времени, неизменяемое собрание данных, применяемых для поддержки процессов принятия управленческих решений. Предметная ориентация означает, что данные объединены в категории и хранятся в соответствии с теми областями, которые они описывают, а не с приложениями, которые их используют. В

хранилище данные интегрируются в целях удовлетворения требований предприятия в целом, а не отдельной функции бизнеса. Привязанность данных ко времени выражает их «историчность», то есть атрибут времени всегда явно присутствует в структурах хранилища данных. Неизменяемость означает, что, попав однажды в хранилище, данные уже не изменяются в отличие от оперативных систем, где данные присутствуют только в последней версии, поэтому постоянно меняются.

Технологии извлечения знаний из хранилищ данных основаны на методах статистического анализа и моделирования, ориентированных на поиск моделей и отношений, скрытых в совокупности данных. Для извлечения значимой информации из хранилищ данных имеются специальные методы (OLAP-анализа, Data Mining или Knowledge Discovery), основанные на применении методов математической статистики, нейронных сетей, индуктивных методов построения деревьев решений и других. Технология OLAP (On-Line Analytical Processing - оперативный анализ данных) предоставляет пользователю средства для формирования и проверки гипотез о свойствах данных или отношениях между ними на основе разнообразных запросов к базе данных. Они применяются на ранних стадиях процесса извлечения знаний, помогая аналитику сфокусировать внимание на важных переменных. Средства Data Mining отличаются от OLAP тем, что кроме проверки предполагаемых зависимостей они способны самостоятельно (без участия пользователя) генерировать гипотезы о закономерностях, существующих в данных, и строить модели, позволяющие количественно оценить степень взаимного влияния исследуемых факторов на основе имеющейся информации.

Адаптивные информационные системы. Потребность в адаптивных информационных системах возникает в тех случаях, когда поддерживаемые ими проблемные области постоянно развиваются. В связи с этим адаптивные системы должны удовлетворять ряду специфических требований, а именно:

- адекватно отражать знания проблемной области в каждый момент времени;
- быть пригодными для легкой и быстрой реконструкции при изменении проблемной среды.

Адаптивные свойства информационных систем обеспечиваются за счет интеллектуализации их архитектуры. Ядром таких систем является постоянно развиваемая модель проблемной области, поддерживаемая в специальной базе знаний - репозитории. Ядро системы управляет процессами генерации или переконфигурирования программного обеспечения. В процессе разработки адаптивных информационных систем применяется оригинальное или типовое проектирование. Оригинальное проектирование предполагает разработку информационной системы с «чистого листа» на основе сформулированных требований. Реализация этого подхода основана на использовании систем автоматизированного проектирования, или CASE-технологий (Designer2000, SilverRun, Natural Light Storm и др.). При типовом проектировании

осуществляется адаптация типовых разработок к особенностям проблемной области. Для реализации этого подхода применяются инструментальные средства компонентного (сборочного) проектирования информационных систем (R/3, BAAN IV, Prodis и др.). Главное отличие подходов состоит в том, что при использовании CASE-технологии на основе репозитория при изменении проблемной области каждый раз выполняется генерация программного обеспечения, а при использовании сборочной технологии - конфигурирование программ и только в редких случаях - их переработка.

3 Технологии разработки экспертных систем

Особенности разработки ИИС. Технология создания интеллектуального программного обеспечения существенно отличается от разработки традиционных программ с использованием известных алгоритмических языков (табл. 2). Рассмотрим отработанные в настоящее время элементы технологии создания ИИС на примере разработки экспертных систем. Этот выбор обусловлен тем, что ЭС получили весьма широкое распространение во многих сферах человеческой деятельности, а технологии их создания имеют универсальный характер и не требуют аппаратных реализаций.

Таблица 2 - Отличия систем искусственного интеллекта от обычных программных систем

Характеристика	Программирование в системах искусственного интеллекта	Традиционное программирование
Тип обработки	Символьный	Числовой
Метод	Эвристический поиск	Точный алгоритм
Задание шагов решения	Неявное	Явное
Искомое решение	Удовлетворительное	Оптимальное
Управление и данные	Смешаные	Разделены
Знания	Неточные	Точные
Модификации	Частые	Редкие

В самых первых ЭС не учитывалось изменение знаний, используемых в процессе решения конкретной задачи. Их назвали статическими ЭС. Типичная статическая ЭС содержит следующие основные компоненты (рис. 4): базу знаний; рабочую память, называемую также базой данных; решатель (интерпретатор); систему объяснений; компоненты приобретения знаний; интерфейс с пользователем

База знаний ЭС предназначена для хранения долгосрочных данных, описывающих рассматриваемую область, и правил, описывающих целесообразные преобразования данных этой области.

База данных (рабочая память) служит для хранения текущих данных решаемой задачи.

Решатель (интерпретатор) формирует последовательность применения правил и осуществляет их обработку, используя данные из рабочей памяти и знания из БЗ.

Система объяснений показывает, каким образом система получила решение задачи, и какие знания при этом использовались. Это облегчает тестирование системы и повышает доверие пользователя к полученному результату.

Компоненты приобретения знаний необходимы для заполнения ЭС знаниями в диалоге с пользователем-экспертом, а также для добавления и модификации заложенных в систему знаний.

К разработке ЭС привлекаются специалисты из разных предметных областей, а именно:

- эксперты той проблемной области, к которой относятся задачи, решаемые ЭС;
- инженеры по знаниям, являющиеся специалистами по разработке ИИС;
- программисты, осуществляющие реализацию ЭС.

Эксперты поставляют знания в ЭС и оценивают правильность получаемых результатов. Инженеры по знаниям помогают экспертам выявить и структурировать знания, необходимые для работы ЭС, выполняют работу по представлению знаний, выбирают методы обработки знаний, проводят выбор инструментальных средств реализации ЭС, наиболее пригодных для решения поставленных задач. Программисты разрабатывают программное обеспечение ЭС и осуществляют его сопряжение со средой, в которой оно будет использоваться. В целом за разработку экспертной системы следует браться организации, где накоплен опыт по автоматизации рутинных процедур обработки информации, например: информационный поиск, графика, сложные расчеты, обработка текстов и т.д.

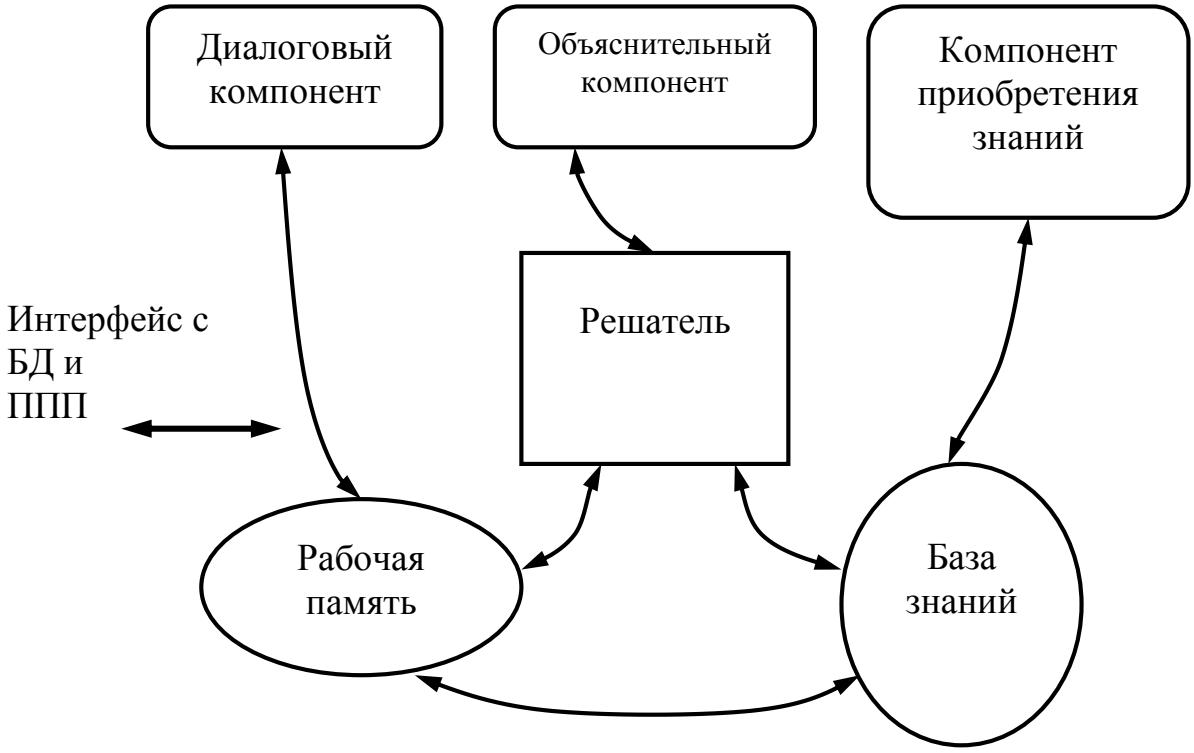


Рисунок 4 - Структура статической экспертной системы

Любая ЭС должна иметь, по крайней мере, два режима работы. В режиме приобретения знаний эксперт наполняет систему знаниями, которые впоследствии позволят ЭС самостоятельно (без помощи эксперта) решать определенные задачи из конкретной проблемной области. Эксперт описывает проблемную область в виде совокупности данных и правил. Данные определяют объекты, их характеристики и значения, существующие в области экспертизы. Правила определяют взаимные связи, существующие между данными, и способы манипулирования данными, характерные для рассматриваемого класса задач. В режиме консультации пользователь ЭС сообщает системе конкретные данные о решаемой задаче и стремится получить с ее помощью результат. Пользователи-неспециалисты обращаются к ЭС за результатом, не умея получить его самостоятельно, пользователи-специалисты используют ЭС для ускорения и облегчения процесса получения результата. Следует подчеркнуть, что термин «пользователь» является многозначным, так как использовать ЭС могут и эксперт, и инженер по знаниям, и программист. Поэтому, когда хотят подчеркнуть, что речь идет о том, для кого делалась ЭС, используют термин «конечный пользователь». В режиме консультации входные данные о задаче поступают в рабочую память. Решатель на основе входных данных из рабочей памяти и правил из БЗ формирует решение. В отличие от традиционных программ компьютерной обработки данных ЭС при решении задачи не только исполняет предписанную последовательность операций, но и сама формирует ее.

Существует широкий класс приложений, в которых требуется учитывать изменения, происходящие в окружающем мире за время исполнения приложения. Для таких задач необходимо применять динамические ЭС. В структуру динамической ЭС (рис. 5) вводятся два компонента: подсистема моделирования внешнего мира и подсистема связи с внешним окружением.

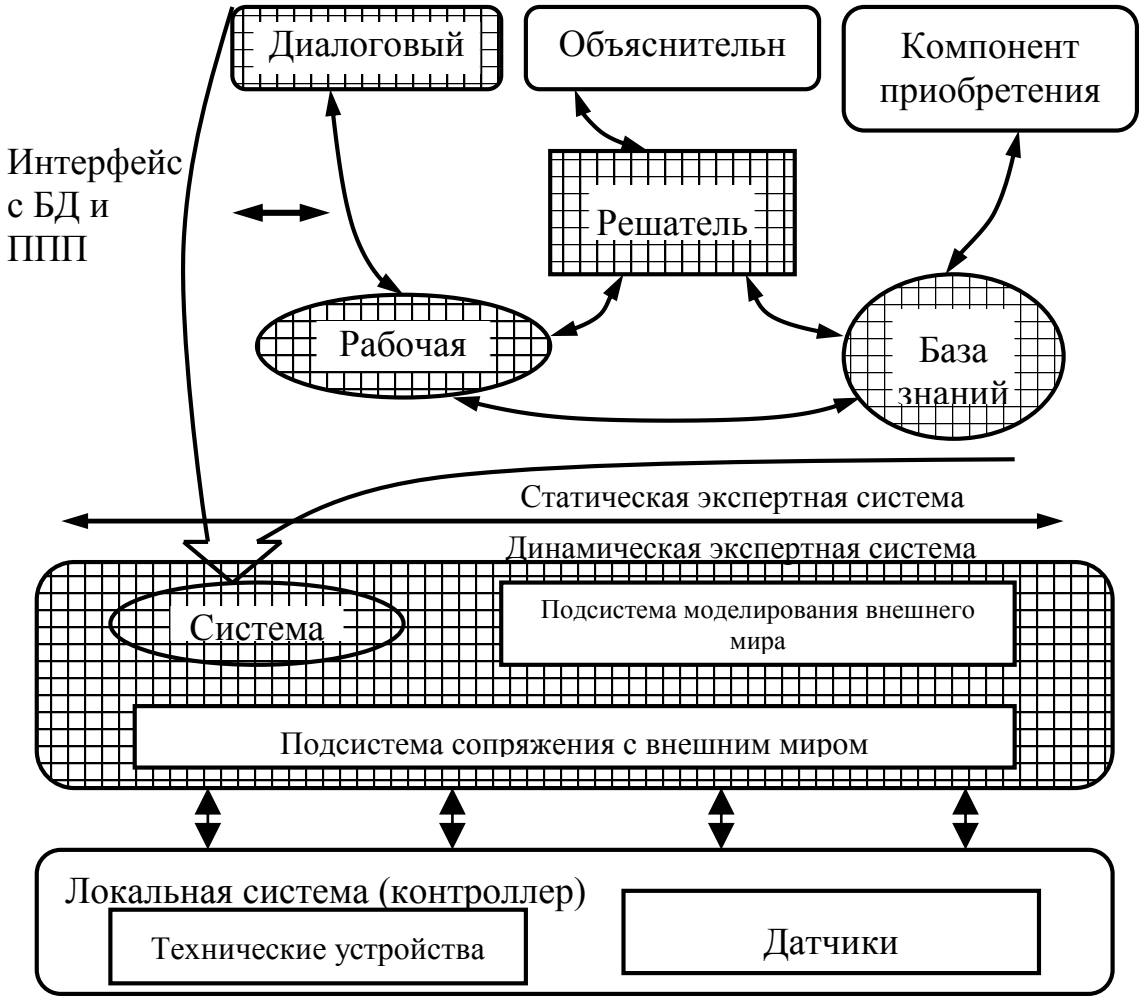


Рисунок 5 - Структура динамической экспертной системы

Подсистема моделирования внешнего мира необходима для прогнозирования, анализа и адекватной оценки состояния внешней среды. Изменения окружения решаемой задачи требуют изменения, хранимых в ЭС знаний, с тем чтобы отразить временную логику происходящих в реальном мире событий. Компонента связи с внешним миром актуальна для автономных интеллектуальных систем (роботов), а также для интеллектуальных систем управления. Связь с внешним миром осуществляется через систему датчиков и контроллеров.

Этапы разработки ЭС. Процесс разработки экспертной системы носит эволюционный характер – от прототипных версий программ к конечному продукту. Прототипная система является усеченной версией экспертной системы, спроектированной для проверки правильности кодирования фактов,

связей и стратегии рассуждения эксперта. Объем прототипа – это несколько десятков правил, фреймов или примеров.

Промышленная технология создания интеллектуальных систем включает следующие этапы:

- исследование выполнимости проекта;
- разработку общей концепции системы;
- разработку и тестирование серии прототипов;
- разработку и испытание головного образца;
- разработку и проверку расширенных версий системы;
- привязку системы к реальной рабочей среде.

Проектирование ЭС основано на трех главных принципах:

1. Мощность экспертной системы обусловлена мощностью БЗ и возможностями ее пополнения и только затем - используемыми методами (процедурами) обработки информации.
2. Знания, позволяющие эксперту (или экспертной системе) получить качественные и эффективные решения задач, являются в основном эвристическими, эмпирическими, неопределенными, правдоподобными.
3. Неформальный характер решаемых задач и используемых знаний делает необходимым обеспечение активного диалога пользователя с ЭС в процессе ее работы.

Перед тем как приступить к разработке ЭС, инженер по знаниям должен рассмотреть вопрос, следует ли разрабатывать ЭС для данного приложения. Положительное решение принимается тогда, когда разработка ЭС возможна, оправданна и методы инженерии знаний соответствуют решаемой задаче. Чтобы разработка ЭС была возможной для данного приложения, необходимо выполнение, по крайней мере, следующих требований:

- существуют эксперты в данной области, которые решают задачу значительно лучше, чем начинающие специалисты;
- эксперты сходятся в оценке предлагаемого решения, так как в противном случае будет невозможно оценить качество разработанной ЭС;
- эксперты способны вербализовать (выразить на естественном языке) и объяснить используемые ими методы, иначе трудно рассчитывать на то, что знания экспертов будут «извлечены» и заложены в ЭС;
- решение задачи требует только рассуждений, а не действий;
- задача не должна быть слишком трудной (то есть ее решение должно занимать у эксперта несколько часов или дней, а не недель или лет);
- задача хотя и не должна быть выражена в формальном виде, но все же должна относиться к достаточно «понятной» и структурированной области, то есть должна существовать возможность выделения основных понятий, отношений и способов получения решения задачи;
- решение задачи не должно в значительной степени опираться на «здравый смысл» (то есть широкий спектр общих сведений о мире и о способе его функционирования, которые знает и умеет использовать любой

нормальный человек), так как подобные знания пока не удается в достаточном количестве заложить в системы искусственного интеллекта.

Приложение соответствует методам ЭС, если решаемая задача обладает совокупностью следующих характеристик:

- задача может быть естественным образом решена посредством манипулирования символами (с помощью символических рассуждений), а не манипулирования числами, как принято в математических методах и в традиционном программировании;
- задача должна иметь эвристическую, а не алгоритмическую природу, то есть ее решение должно требовать применения эвристических правил. Для задач, которые могут быть гарантированно решены (при соблюдении заданных ограничений) с помощью формальных процедур, существуют более эффективные подходы, чем технологии ЭС.

При разработке ЭС, как правило, используется концепция быстрого прототипа, суть которой заключается в том, что разработчики не пытаются сразу построить конечный продукт. На начальном этапе они создают прототип (возможно, не единственный) ЭС, удовлетворяющий двум противоречивым требованиям: умение решать типичные задачи конкретного приложения и незначительные время и трудоемкость его разработки. При выполнении этих условий становится возможным параллельно вести процесс накопления и отладки знаний, осуществляемый экспертом, и процесс выбора (разработки) программных средств, выполняемый инженером по знаниям и программистами. Для удовлетворения указанным требованиям при создании прототипа используются разнообразные инструментальные средства, ускоряющие процесс проектирования.

Традиционная технология реализации ЭС включает шесть основных этапов (рис. 6): идентификацию, концептуализацию, формализацию, выполнение, тестирование, опытную эксплуатацию.

На этапе идентификации определяются задачи, подлежащие решению, цели разработки, эксперты и типы пользователей.

На этапе концептуализации проводится содержательный анализ проблемной области, выявляются используемые понятия и их взаимосвязи, определяются методы решения задач.

На этапе формализации выбираются инструментальные средства и способы представления всех видов знаний, формализуются основные понятия, определяются способы интерпретации знаний, моделируется работа системы, оценивается адекватность системы зафиксированных понятий, методов решения, средств представления и манипулирования знаниями рассматриваемой предметной области.

На этапе выполнения осуществляется заполнение базы знаний. В связи с тем, что основой ЭС являются знания, данный этап является одним из самых важных и самых трудоемких. Процесс приобретения знаний разделяют на извлечение знаний в Диалоге с экспертами; организацию знаний, обеспечивающую эффективную работу системы, и представление знаний в

виде, «понятном» ЭС. Процесс приобретения знаний осуществляется инженером по знаниям на основе анализа деятельности эксперта по решению реальных задач.

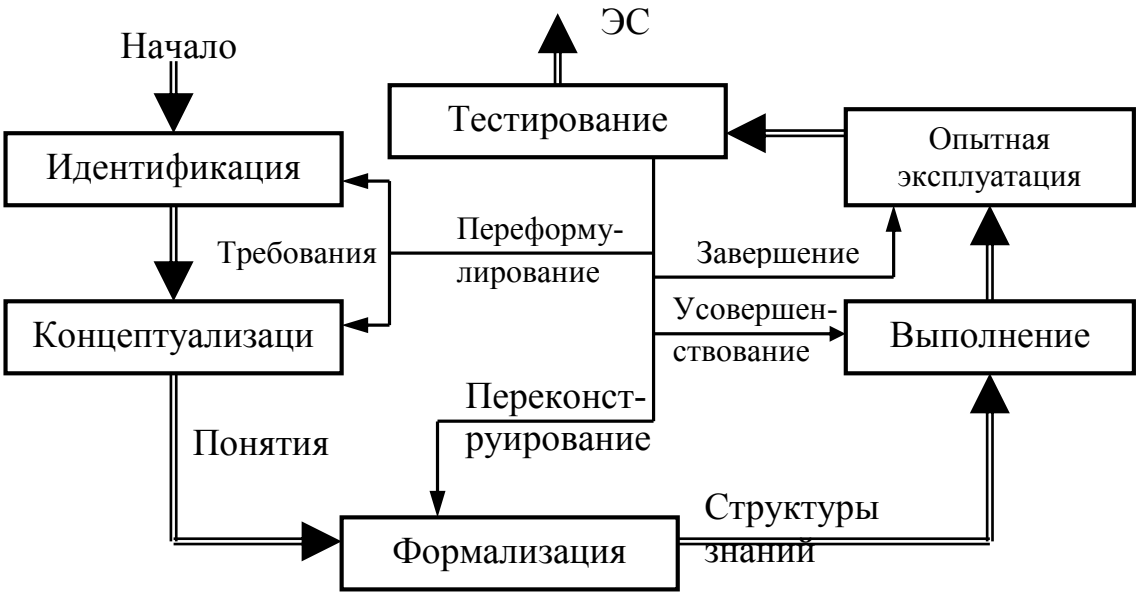


Рисунок 6 - Этапы разработки экспертных систем

На этапе тестирования эксперт и инженер по знаниям в интерактивном режиме с использованием диалоговых и объяснительных средств проверяют компетентность ЭС. Процесс тестирования продолжается до тех пор, пока эксперт не решит, что система достигла требуемого уровня компетентности.

На этапе опытной эксплуатации проверяется пригодность ЭС для конечных пользователей. Полученные результаты могут показать необходимость существенной модификации ЭС.

Процесс создания ЭС не сводится к строгой последовательности перечисленных выше этапов. В ходе разработки приходится неоднократно возвращаться на более ранние этапы и пересматривать принятые там решения.

Инструментальные средства различаются в зависимости от того, какую технологию разработки ЭС они допускают. Можно выделить, по крайней мере, четыре подхода к разработке ЭС:

- подход, базирующийся на поверхностных знаниях;
- структурный подход;
- подход, основанный на глубинных знаниях;
- смешанный подход, опирающийся на использование поверхностных и глубинных знаний.

Поверхностный подход применяется для сложных задач, которые не могут быть точно описаны. Его сущность состоит в получении от экспертов фрагментов знаний, релевантных решаемой задаче. При этом не предпринимается попыток систематического или глубинного изучения области, что предопределяет использование поиска в пространстве состояний

в качестве универсального механизма вывода. Обычно в ЭС, использующих данный подход, в качестве способа представления знаний выбираются правила. Условие каждого правила определяет образец некоторой ситуации, в которой правило может быть выполнено. Поиск решения состоит в выполнении тех правил, образцы которых сопоставляются с текущими данными. При этом предполагается, что в процессе поиска решения последовательность формируемых таким образом ситуаций не оборвется до получения решения, т.е. не возникнет неизвестной ситуации, которая не соответствует ни одному правилу. Данный подход с успехом применяется к широкому классу приложений, но оказывается неэффективным в тех случаях, когда задача может структурироваться или для ее решения может использоваться некоторая модель.

Структурный подход к построению ЭС предусматривает структуризацию знаний проблемной области. Его появление обусловлено тем, что для ряда приложений применение техники поверхностных знаний не обеспечивает решения задачи. Структурный подход к построению ЭС во многом похож на структурное программирование. Однако применительно к ЭС речь идет не о том, чтобы структурирование задачи было доведено до точного алгоритма (как в традиционном программировании), а предполагается, что часть задачи решается с помощью эвристического поиска. Структурный подход в различных приложениях целесообразно сочетать с поверхностным или глубинным знанием.

При глубинном подходе компетентность ЭС базируется на модели той проблемной среды, в которой она работает. Модель может быть определена различными способами, то есть декларативно или процедурно. Экспертные системы, разработанные с применением глубинных знаний, при возникновении неизвестной ситуации способны самостоятельно определить, какие действия следует выполнить, с помощью некоторых общих принципов, справедливых для данной области экспертизы. Глубинный подход требует явного описания структуры и взаимоотношений между различными сущностями проблемной области. В этом подходе необходимо использовать инструментальные средства, обладающие возможностями моделирования: объекты с присоединенными процедурами, иерархическое наследование свойств, активные знания (программирование, управляемое данными), механизм передачи сообщений объектам (объектно-ориентированное программирование) и т. п.

Смешанный подход в общем случае может сочетать поверхностный, структурный и глубинный подходы. Например, поверхностный подход может применяться для поиска адекватных знаний, которые затем используются некоторой глубинной моделью.

Классификационные признаки ЭС. В основе классификации экспертных систем лежат следующие параметры: тип приложения, стадия существования, масштаб, тип проблемной среды, тип решаемой задачи.

Тип приложения характеризуется следующими признаками.

1. Возможность взаимодействия приложения с другими программными средствами:

- изолированное приложение - экспертная система, не способная взаимодействовать с другими программными системами (например, с базами данных, электронными таблицами, пакетами прикладных программ, контроллерами, датчиками и т.п.);
- интегрированное приложение - экспертная система и другие программные системы, с которыми она взаимодействует в ходе работы. Большинство современных ЭС, используемых для решения практически значимых задач, являются интегрированными.

2. Возможность исполнять приложение на разнородной аппаратуре и переносить его на различные платформы:

- закрытые приложения — исполняются только в программной среде данной фирмы и могут быть перенесены на другие платформы только путем перепрограммирования приложения;
- открытые приложения — ориентированы на исполнение в разнородном программно-аппаратном окружении и могут быть перенесены на другие платформы без перепрограммирования.

3. Архитектура приложения:

- централизованное приложение - реализуется на базе центральной ЭВМ, с которой связаны терминалы;
- распределенное приложение - обычно используется архитектура клиент-сервер.

Стадия существования характеризует степень завершенности разработки ЭС. В нее входят:

- исследовательский прототип - решает представительный класс задач проблемной области, но может быть неустойчив в работе и не полностью проверен. При наличии развитых инструментальных средств при разработке исследовательского прототипа требуется примерно 2-4 месяца. База знаний исследовательского прототипа обычно содержит небольшое число исполняемых утверждений;
- действующий прототип - надежно решает любые задачи проблемной области, но при решении сложных задач может потребовать чрезмерно много времени и (или) памяти. Доведение системы от начала разработки до стадии действующего прототипа требует примерно 6 - 9 месяцев, при этом количество исполняемых утверждений в базе знаний увеличивается по сравнению с исследовательским прототипом;
- промышленная система - обеспечивает высокое качество решения всех задач при минимуме времени и памяти. Обычно процесс преобразования действующего прототипа в промышленную систему состоит в расширении базы знаний и ее тщательной отладке. Доведение ЭС от начала разработки до стадии промышленной системы с применением развитых инструментальных средств требует не менее 12-18 месяцев;

- коммерческая система - пригодна не только для использования разработчиком, но и для продажи различным потребителям. Доведение системы до коммерческой стадии требует примерно 1,5-2 года. Приведенные здесь сроки справедливы для ЭС средней сложности.

Масштаб ЭС характеризует сложность решаемых задач и связан с типом используемой ЭВМ.

По этому признаку различают:

- малые ЭС - предназначены для первичного обучения и исследования возможности применения технологии ЭС для рассматриваемого класса задач. Системы такого типа могут быть реализованы на персональных компьютерах;
- средние ЭС - охватывают весь спектр необходимых приложений и обычно интегрированы с базами данных, электронными таблицами и т.д. Системы такого масштаба чаще всего реализуются на рабочих станциях;
- большие ЭС - имеют доступ к мощным базам данных и реализуются на рабочих станциях или на специализированных компьютерах;
- символьные ЭС - создаются с исследовательскими целями и реализуются на специализированных компьютерах, ориентированных на обработку символьных данных.

Понятие проблемной среды включает описание предметной области (множество сущностей, описывающих множество объектов, их характеристик и отношений между объектами) и решаемых в ней задач. Другими словами, проблемная среда включает структуры данных и решаемые с ними задачи, представленные в виде исполняемых утверждений (правил, процедур, формул и др.). В связи с этим проблемная среда определяется характеристиками соответствующей предметной области и характеристиками типов решаемых в ней задач.

Характеристики предметной области.

1. Тип предметной области:

- статический - входные данные не изменяются за время сеанса работы приложения, значения других (не входных) данных изменяются только самой экспертной системой;
- динамический - входные данные, поступающие из внешних источников, изменяются во времени, значения других данных изменяются ЭС или подсистемой моделирования внешнего окружения.

2. Способ описания сущностей предметной области:

- совокупность атрибутов и их значений (фиксированный состав сущностей);
- совокупность классов (объектов) и их экземпляров (изменяемый состав сущностей).

3. Способ организации сущностей в БЗ:

- неструктурированная БЗ;

- структурирование сущностей в БЗ по различным иерархиям, («частное - общее», «часть - целое», «род - вид»), что обеспечивает наследование свойств сущностей.

Структурирование БЗ способствует:

- ограничению круга сущностей, которые должны рассматриваться механизмом вывода, и сокращению количества перебираемых вариантов в процессе выбора решения;
- обеспечению наследования свойств сущностей, т.е. передачи свойств вышерасположенных в иерархии сущностей нижерасположенным сущностям, что значительно упрощает процесс приобретения и использования знаний.

Характеристики типов, решаемых в проблемной области задач.

1. Тип решаемых задач:

- задачи анализа или синтеза. В задаче анализа задана модель сущности и требуется определить неизвестные характеристики модели. В задаче синтеза задаются условия, которым должны удовлетворять характеристики «неизвестной» модели сущности, и требуется построить модель этой сущности. Решение задачи синтеза обычно включает задачу анализа как составную часть;
- статические или динамические задачи. Если задачи, решаемые ЭС, явно не учитывают фактор времени и/или не изменяют в процессе своего решения знания об окружающем мире, то ЭС решает статические задачи, в противном случае речь идет о решении динамических задач. Учитывая значимость времени в динамических проблемных средах, многие специалисты называют их приложениями, работающими в реальном времени. Обычно выделяют следующие системы реального времени: псевдореального времени, «мягкого» реального времени и «жесткого» реального времени. Системы псевдореального времени, как следует из названия, не являются системами реального времени, однако они, в отличие от статических систем, получают и обрабатывают данные, поступающие из внешних источников. Системы псевдореального времени решают задачу быстрее, чем происходят значимые изменения информации об окружающем мире.

2. Общность исполняемых утверждений:

- частные исполняемые утверждения, содержащие ссылки на конкретные сущности (объекты);
- общие исполняемые утверждения, относящиеся к любым сущностям заданного типа (вне зависимости от их числа и имени). Использование общих утверждений позволяет значительно лаконичнее представлять знания. Однако поскольку общие утверждения не содержат явных ссылок на конкретные сущности, для их использования каждый раз требуется определять те сущности, к которым они должны применяться.

Не все сочетания перечисленных выше параметров, характеризующих проблемную среду, встречаются на практике. Наиболее распространены следующие типы проблемных сред:

- статическая предметная область:

представление сущностей в виде совокупности атрибутов и их значений, неизменяемый состав сущностей, БЗ не структурирована, решаются статические задачи анализа, используются только частные исполняемые утверждения;

представление сущностей объектами, изменяемый состав сущностей, БЗ структурирована, решаются статические задачи анализа и синтеза, используются общие и частные исполняемые утверждения;

- динамическая предметная область:

представление сущностей совокупностью атрибутов и их значений, неизменяемый состав сущностей, БЗ не структурирована, решаются динамические задачи анализа, используются частные исполняемые утверждения;

представление сущностей в виде объектов, изменяемый состав сущностей, БЗ структурирована, решаются динамические задачи анализа и синтеза, используются общие и частные исполняемые утверждения.

В ЭС различают следующие типы решения задач:

- интерпретация данных - процесс определения смысла данных, результаты которого должны быть согласованными и корректными. Экспертные системы, как правило, проводят многовариантный анализ данных;
- диагностика - процесс соотнесения объекта с некоторым классом объектов и/или обнаружение неисправностей в системе (отклонений параметров системы от нормативных значений);
- мониторинг - непрерывная интерпретация данных в реальном масштабе времени и сигнализация о выходе тех или иных параметров за допустимые пределы;
- проектирование - создание ранее не существовавшего объекта и подготовка спецификаций на создание объектов с заранее определенными свойствами. Степень новизны может быть разной и определяется видом знаний, заложенных в ЭС, и методами их обработки. Для организации эффективного проектирования требуется формировать не только сами проектные решения, но и мотивы их принятия. ЭС, решающие задачи проектирования, реализуют процедуры вывода решения и объяснения полученных результатов;
- прогнозирование - предсказание последствий некоторых событий или явлений на основе анализа имеющихся данных. Прогнозирующие ЭС логически выводят вероятные следствия из заданных ситуаций. В прогнозирующих ЭС в большинстве случаев используются динамические модели, в которых значения параметров «подгоняются» под заданную ситуацию. Выводимые из этих моделей следствия составляют основу для прогнозов с вероятностными оценками;
- планирование - построение планов действий объектов, способных выполнять некоторые функции. Работа ЭС по планированию основана на

моделях поведения реальных объектов, которые позволяют проводить логический вывод последствий планируемой деятельности;

- обучение - использование компьютера для обучения каким-либо дисциплине или предмету. Экспертные системы обучения выполняют такие функции, как диагностика ошибок, подсказывание правильных решений; аккумулирование знаний о гипотетическом «ученике» и его характерных ошибках; диагностирование слабости в познаниях обучаемых и нахождение соответствующих средств для их ликвидации. Системы обучения способны планировать акт общения с учеником в зависимости от успехов ученика для передачи необходимых знаний;
- управление - функция организованной системы, поддерживающая определенный режим ее деятельности. Экспертные системы данного типа предназначены для управления поведением сложных систем в соответствии с заданными спецификациями;
- поддержка принятия решений - совокупность процедур, обеспечивающая лицо, принимающее решения, необходимой информацией и рекомендациями, облегчающими процесс принятия решения. Такого рода ЭС оказывают помощь специалистам в выборе и/или генерации наиболее рациональной альтернативы из множества возможных при принятии ответственных решений.

Задачи интерпретации данных, диагностики, поддержки принятия решений относятся к задачам анализа, задачи проектирования, планирования и управления - к задачам синтеза. К комбинированному типу задач относятся обучение, мониторинг и прогнозирование.

Характеристика инструментальных средств разработки ИИС. Трудоемкость разработки ИИС в значительной степени зависит от используемых инструментальных средств (ИС). Инструментальные средства для разработки интеллектуальных приложений можно классифицировать по следующим основным параметрам:

- уровень используемого языка;
- парадигмы программирования и механизмы реализации;
- способ представления знаний;
- механизмы вывода и моделирования;
- средства приобретения знаний;
- технологии разработки приложений.

Мощность и универсальность языка программирования определяет трудоемкость разработки ЭС:

1. традиционные (в том числе объектно-ориентированные) языки программирования типа С, С++ (как правило, они используются не для создания ЭС, а для создания инструментальных средств);
2. специальные языки программирования (например, язык LISP**,** ориентированный на обработку списков; язык логического программирования PROLOG**;** язык рекурсивных функций РЕФАЛ и др.). Их недостатком является

слабая приспособленность к объединению с программами, написанными на языках традиционного программирования;

3. инструментальные средства, содержащие многие, но не все компоненты ЭС. Такое программное обеспечение предназначено для разработчиков, владеющих технологиями программирования умеющих интегрировать разнородные компоненты в программный комплекс.

4. оболочки ЭС общего назначения, содержащие все программные компоненты, но не имеющие знаний о конкретных предметных средах. Средства этого типа и последующего не требуют от разработчика приложения знания программирования. Примерами являются ЭКО, Leonardo, Nexpert Object, Kappa, EXSYS, GURU, ART, KEE и др. В последнее время все реже употребляется термин «оболочка», его заменяют более широким термином «среда разработки». Если хотят подчеркнуть, что средство используется не только на стадии разработки приложения, но и на стадиях использования и сопровождения, то употребляют термин «полная среда» (complete environment). Для поддержания всего цикла создания и сопровождения программ используются интегрированные инструментальные системы типа Work Bench, например KEATS, VITAL. Основными компонентами системы KEATS являются: ACQUIST - средства фрагментирования текстовых источников знаний, позволяющие разбивать текст или протокол беседы с экспертом на множество взаимосвязанных, аннотированных фрагментов и создавать понятия (концепты); FLIK - язык представления знаний средствами фреймовой модели; GIS - графический интерфейс, используемый для создания гипертекстов и концептуальных моделей, а также для проектирования фреймовых систем; ERI - интерпретатор правил, реализующий процедуры прямого и обратного вывода; TRI — инструмент визуализации логического вывода, демонстрирующий последовательность выполнения правил; Tables - интерфейс манипулирования таблицами, используемыми для хранения знаний в БЗ; CS — язык описания и распространения ограничений; TMS - немонотонная система поддержания истинности. При использовании инструментария данного типа могут возникнуть следующие трудности:

- управляющие стратегии, заложенные в механизм вывода, могут не соответствовать методам решения, которые использует эксперт, взаимодействующий с данной системой, что может привести к неэффективным, а возможно, и неправильным решениям;
- способ представления знаний, используемый в инструментарии, мало подходит для описания знаний конкретной предметной области.

5. Проблемно/предметно-ориентированные оболочки и среды (не требуют знания программирования):

- проблемно-ориентированные средства, предназначенные для решения задач определенного класса (задачи поиска, управления, планирования, прогнозирования и др.) и содержат соответствующие этому классу функциональные модули;

- предметно-ориентированные средства, использующие знания о типах предметных областей, что сокращает время разработки БЗ.

При использовании оболочек и сред разработчик приложения полностью освобождается от программирования, его основные трудозатраты связаны с формированием базы знаний.

Способы реализации механизма исполняемых утверждений часто называют парадигмами программирования. К основным парадигмам относят следующие:

- процедурное программирование;
- программирование, ориентированное на данные;
- программирование, ориентированное на правила;
- объектно-ориентированное программирование.

Парадигма процедурного программирования является самой распространенной среди существующих языков программирования (например, С и Паскаль). В процедурной парадигме активная роль отводится процедурам, а не данным; причем любая процедура активизируется вызовом. Подобные способы задания поведения удобны для описаний детерминированной последовательности действий одного процесса или нескольких взаимосвязанных процессов.

При использовании программирования, ориентированного на данные, активная роль принадлежит данным, а не процедурам. Здесь со структурами активных данных связывают некоторые действия (процедуры), которые активизируются тогда, когда осуществляется обращение к этим данным.

В парадигме, ориентированной на правила, поведение определяется множеством правил вида «условие-действие». Условие задает образ данных, при возникновении которого действие правила может быть выполнено. Правила в данной парадигме играют такую же роль, как и операторы в процедурной парадигме. Однако если в процедурной парадигме поведение задается детерминированной последовательностью операторов, не зависящей от значений обрабатываемых данных, то в парадигме, ориентированной на правила, поведение не задается заранее предписанной последовательностью правил, а формируется на основе значений Данных, которые в текущий момент обрабатываются программой. Подход, ориентированный на правила, удобен для описания поведения, гибко и разнообразно реагирующего на большое многообразие состояний данных.

Парадигма объектного программирования в отличие от процедурной парадигмы не разделяет программу на процедуры и данные. Здесь программа организуется вокруг сущностей, называемых объектами, которые включают локальные процедуры (методы) и локальные данные (переменные). Поведение (функционирование) в этой парадигме организуется путем пересылки сообщений между объектами. Объект, получив сообщение, осуществляет его локальную интерпретацию, основываясь на локальных процедурах и данных. Такой подход позволяет описывать сложные системы наиболее естественным образом, что особенно удобно для интегрированных ЭС.

Наличие многих способов представления знаний вызвано стремлением представить различные типы проблемных сред с наибольшей эффективностью. Обычно способ представления знаний в ЭС характеризуют моделью представления знаний. Типичными моделями представления знаний являются правила (продукции), фреймы (или объекты), семантические сети, логические формулы. Инструментальные средства, имеющие в своем составе более одной модели представления знаний, называют гибридными. Большинство современных средств, как правило, использует объектно-ориентированную парадигму, объединенную с парадигмой, ориентированной на правила.

В статических ЭС единственным агентом, изменяющим информацию, является механизм вывода экспертной системы. В динамических ЭС изменение данных происходит не только вследствие функционирования механизма исполняемых утверждений, но также в связи с изменениями окружения задачи, которые моделируются специальной подсистемой или поступают извне.

Механизмы вывода в различных средах могут отличаться способами реализации следующих процедур.

1. Структура процесса получения решения:

- построение дерева вывода на основе обучающей выборки (индуктивные методы приобретения знаний) и выбор маршрута на дереве вывода в режиме решения задачи;
- компиляция сети вывода из специфических правил в режиме приобретения знаний и поиск решения на сети вывода в режиме решения задачи;
- генерация сети вывода и поиск решения в режиме решения задачи, при этом генерация сети вывода осуществляется в ходе выполнения операции сопоставления, определяющей пары «правило — совокупность данных», на которых условия этого правила удовлетворяются;
- в режиме решения задач ЭС осуществляет выработку правдоподобных предположений (при отсутствии достаточной информации для решения); выполнение рассуждений по обоснованию (опровержению) предположений; генерацию альтернативных сетей вывода; поиск решения в сетях вывода.

2. Поиск (выбор) решения:

- направление поиска - от данных к цели, от целей к данным, двунаправленный поиск;
- порядок перебора вершин в сети вывода

а) «поиск в ширину», при котором сначала обрабатываются все вершины, непосредственно связанные с текущей обрабатываемой вершиной G;

б) «поиск в глубину», когда сначала раскрывается одна наиболее значимая вершина – G_1, связанная с текущей G, затем вершина G_1 делается текущей, и для нее раскрывается одна наиболее значимая вершина G_2 и т. д.

3. Процесс генерации предположений и сети вывода:

- режим - генерация в режиме приобретения знаний, генерация в режиме решения задачи;
- полнота генерируемой сети вывода - операция сопоставления, применяемая ко всем правилам и ко всем типам указанных в правилах сущностей в каждом цикле работы механизма вывода; используются различные средства для сокращения количества правил и (или) сущностей, участвующих в операции сопоставления; например, применяется алгоритм сопоставления или используются знания более общего характера (метазнания).

Механизм вывода для динамических проблемных сред дополнительно содержит: планировщик, управляющий деятельностью ЭС в соответствии с приоритетами; средства, гарантирующие получение лучшего решения в условиях ограниченности ресурсов; систему поддержания истинности значений переменных, изменяющихся во времени. В динамических инструментальных средствах могут быть реализованы следующие варианты подсистемы моделирования:

- система моделирования отсутствует;
- существует система моделирования общего назначения, являющаяся частью инструментальной среды; |
- существует специализированная система моделирования, являющаяся внешней по отношению к программному обеспечению, на котором реализуется ЭС.

В инструментальных системах средства приобретения знаний характеризуются следующими признаками:

1. Уровень языка приобретения знаний: формальный язык; ограниченный естественный язык; язык пиктограмм и изображений; ЕЯ и язык изображений.
2. Тип приобретаемых знаний: данные в виде таблиц, содержащих значения входных и выходных атрибутов, по которым индуктивными методами строится дерево вывода; специализированные правила; общие и специализированные правила.
3. Тип приобретаемых данных: атрибуты и значения; объекты; классы структурированных объектов и их экземпляры, получающие значения атрибутов путем наследования.

4 Состав и организация данных и знаний в экспертных системах

Отличия знаний от данных. Характерным признаком интеллектуальных систем является наличие знаний, необходимых для решения задач конкретной предметной области. При этом возникает естественный вопрос, что такое знания и чем они отличаются от обычных данных, обрабатываемых ЭВМ. Данными называют информацию

фактического характера, описывающую объекты, процессы и явления предметной области, а также их свойства. В процессах компьютерной обработки данные проходят следующие этапы преобразований:

- исходная форма существования данных (результаты наблюдений и измерений, таблицы, справочники, диаграммы, графики и т.д.);
- представление на специальных языках описания данных, предназначенных для ввода и обработки исходных данных в ЭВМ;
- базы данных на машинных носителях информации.

Знания являются более сложной категорией информации по сравнению с данными. Знания описывают не только отдельные факты, но и взаимосвязи между ними, поэтому знания иногда называют структурированными данными. Знания могут быть получены на основе обработки эмпирических данных. Они представляют собой результат мыслительной деятельности человека, направленной на обобщение его опыта, полученного в результате практической деятельности.

Для того чтобы наделить ИИС знаниями, их необходимо представить в определенной форме. Существуют два способа наделения знаниями программных систем:

- первый способ предусматривает поместить знания в программу, написанную на обычном языке программирования. Такая система будет представлять собой единый программный код, в котором знания не вынесены в отдельную категорию. В этом случае трудно оценить роль знаний и понять, каким образом они используются в процессе решения задач. Трудным делом является модификация и сопровождение подобных программ, а проблема пополнения знаний может стать неразрешимой;
- второй способ базируется на концепции баз данных и заключается в вынесении знаний в отдельную категорию, то есть знания представляются в определенном формате и помещаются в БЗ. База знаний легко пополняется и модифицируется. Она является автономной частью интеллектуальной системы, хотя механизм логического вывода, реализованный в логическом блоке, а также средства ведения диалога накладывают определенные ограничения на структуру БЗ и операции с нею. В современных ИИС принят этот способ.

При разработке ИИС сначала осуществляются накопление и представление знаний, причем на этом этапе обязательно участие человека, а затем знания представляются определенными структурами данных, удобными для хранения и обработки в ЭВМ. Знания в ИИС существуют в следующих формах:

- исходные знания (правила, выведенные на основе практического опыта, математические и эмпирические зависимости, отражающие взаимные связи между фактами; закономерности и тенденции, описывающие изменение фактов с течением времени; функции, диаграммы, графы);

- описание исходных знаний средствами выбранной модели представления знаний (множество логических формул или продукционных правил, семантическая сеть, иерархии фреймов);
- представление знаний структурами данных, которые предназначены для хранения и обработки в ЭВМ;
- базы знаний на машинных носителях информации.

В области ИИ даются конкретные определения знаний. «Знания - это закономерности предметной области (принципы, связи, законы), полученные в результате практической деятельности и профессионального опыта, позволяющие специалистам ставить и решать задачи в этой области». «Знания - это хорошо структурированные данные или данные о данных, или метаданные». «Знания - формализованная информация, на которую ссылаются или используют в процессе логического вывода».

Существует множество классификаций знаний. Как правило, с помощью классификаций систематизируют знания конкретных предметных областей. По своей природе знания можно разделить на декларативные и процедурные знания. Декларативные знания - это описания фактов и явлений. Процедурные знания - это описания действий, которые возможны при манипулировании фактами и явлениями для достижения намеченных целей.

Для описания знаний на абстрактном уровне разработаны специальные языки описания знаний. Эти языки также делятся на языки процедурного типа и декларативного. Все языки описания знаний, ориентированные на использование традиционных компьютеров фон-неймановской архитектуры, являются языками процедурного типа. Разработка языков декларативного типа, удобных для представления знаний, является актуальной проблемой.

По способу приобретения знания можно разделить на факты и эвристику (правила, которые позволяют сделать выбор при отсутствии точных теоретических обоснований). Первая категория знаний обычно указывает на хорошо известные в данной предметной области обстоятельства. Вторая категория знаний основана на собственном опыте эксперта, работающего в конкретной предметной области, накопленном в результате многолетней практики.

По типу представления знания делятся на факты и правила. Факты - это знания типа «А - это А», такие знания характерны для баз данных и сетевых моделей. Правила, или продукции, - это знания типа «если А, то В».

Кроме фактов и правил существуют еще метазнания - знания о знаниях. Они необходимы для управления БЗ и для эффективной организации процедур логического вывода.

Форма представления знаний оказывает существенное влияние на характеристики ИИС. Базы знаний являются моделями человеческих знаний. Однако все знания, которые привлекает человек в процессе решения сложных задач, смоделировать невозможно. Поэтому в интеллектуальных системах требуется четко разделить знания на те, которые предназначены для обработки компьютером, и знания, используемые человеком. Очевидно, что для решения

сложных задач БЗ должна иметь достаточно большой объем, иначе неизбежно возникают проблемы управления такой базой. Поэтому при выборе модели представления знаний следует учитывать такие факторы, как однородность представления и простота понимания. Однородность представления приводит к упрощению механизма управления знаниями. Простота понимания важна для пользователей интеллектуальных систем и экспертов, чьи знания закладываются в ИИС. Если форма представления знаний будет трудна для понимания, то усложняются процессы приобретения и интерпретации знаний.

Организация данных в рабочей памяти ЭС. Рабочая память (РП) экспертных систем предназначена для хранения данных. Данные в рабочей памяти делятся на уровни по типам данных. Выделение уровней усложняет структуру экспертной системы, однако делает ее более эффективной. Например, можно выделить уровень планов, уровень агенды (упорядоченного списка правил, готовых к выполнению), уровень данных предметной области, уровень решений. В рабочей памяти современных экспертных систем рассматриваются как изолированные, так и связанные данные. В первом случае рабочая память состоит из множества простых элементов, а втором - из одного или нескольких (при нескольких уровнях в РП) сложных элементов (например, объектов). При этом сложный элемент соответствует множеству простых элементов, объединенных в единую сущность. Теоретически оба подхода обеспечивают полноту, но использование изолированных элементов в сложных предметных областях приводит к потере эффективности. Данные в РП в простейшем случае являются константами и (или) переменными. При этом переменные могут рассматриваться как характеристики некоторого объекта, а константы - как значения соответствующих характеристик. Если в РП требуется анализировать одновременно несколько различных объектов, описывающих текущую проблемную ситуацию, то необходимо указывать, к каким объектам относятся рассматриваемые характеристики. Одним из способов решения этой задачи является явное указание того, к какому объекту относится характеристика. Если РП состоит из сложных элементов, то связь между отдельными объектами указывается явно, например заданием семантических отношений. При этом каждый объект может иметь свою внутреннюю структуру. Для ускорения поиска и сопоставления данных в РП используются не только логические, но и ассоциативные связи.

Показателем интеллектуальности системы с точки зрения представления знаний считается способность системы использовать в нужный момент необходимые (релевантные) знания. Системы, не имеющие средств для определения релевантных знаний, неизбежно сталкиваются с проблемой «комбинаторного взрыва». Можно утверждать, что эта проблема является одной из основных причин, ограничивающих сферу применения экспертных систем. В проблеме доступа к знаниям можно выделить три аспекта: связность знаний и данных, механизм доступа к знаниям и способ сопоставления.

Организация знаний в БЗ. Связность знаний является основным способом, обеспечивающим ускорение поиска релевантных знаний. Для этого

знания следует организовывать вокруг наиболее важных объектов (сущностей) предметной области. Все знания, характеризующие некоторую сущность, связываются и представляются в виде отдельного объекта. При подобной организации знаний, если системе потребовалась информация о некоторой сущности, то она ищет объект, описывающий эту сущность, а затем уже внутри объекта отыскивает информацию о данной сущности.

В объектах целесообразно выделять два типа связок между элементами: внешние и внутренние. Внутренние связки объединяют элементы в единый объект и предназначены для выражения структуры объекта. Внешние связки отражают взаимозависимости, существующие между объектами в области экспертизы.

Внешние связки бывают логические и ассоциативные. Логические связки выражают семантические отношения между элементами знаний. Ассоциативные связки предназначены для обеспечения взаимосвязей, способствующих ускорению процесса поиска релевантных знаний. Основной проблемой при работе с большой базой знаний является проблема поиска знаний, релевантных решаемой задаче. Очевидно, что упорядочение и структурирование знаний могут значительно ускорить процесс поиска. Нахождение желаемых объектов в общем случае уместно рассматривать как двухэтапный процесс. На первом этапе, соответствующем процессу выбора по ассоциативным связкам, совершается предварительный выбор в базе знаний потенциальных кандидатов в качестве желаемых объектов. На втором этапе путем выполнения операции сопоставления потенциальных кандидатов с описаниями кандидатов осуществляется окончательный выбор искомых объектов. При организации подобного механизма доступа возникают определенные трудности с выбором критерия пригодности кандидата, организации работы в конфликтных ситуациях и др. Операция сопоставления может использоваться не только как средство выбора нужного объекта из множества кандидатов, но и для классификации, подтверждения, декомпозиции и коррекции. Для идентификации неизвестного объекта он может быть сопоставлен с некоторыми известными образцами. Это позволит классифицировать неизвестный объект как такой известный образец, при сопоставлении с которым были получены лучшие результаты. Если осуществлять сопоставление некоторого известного объекта с неизвестным описанием, то в случае успешного сопоставления будет осуществлена частичная декомпозиция описания.

Операции сопоставления весьма разнообразны. Обычно выделяют следующие их формы: синтаксическое, параметрическое, семантическое и принуждаемое сопоставления.

В случае синтаксического сопоставления соотносят формы (образцы), а не содержание объектов, Успешным является сопоставление, в результате которого образцы оказываются идентичными. Обычно считается, что переменная одного образца может быть идентична любой константе (или выражению) другого образца. Иногда на переменные, входящие в образец,

накладывают требования, определяющие тип констант, с которыми они могут сопоставляться Результат синтаксического сопоставления является бинарным: образцы сопоставляются или не сопоставляются.

В параметрическом сопоставлении вводится параметр, определяющий степень сопоставления.

В случае семантического сопоставления соотносятся не образцы объектов, а их функции.

В случае принуждаемого сопоставления один сопоставляемый образец рассматривается с точки зрения другого. В отличие от других типов сопоставления здесь всегда может быть получен положительный результат. Вопрос состоит в силе принуждения и использовании специальных процедур, связываемых с объектами.

Модели представления знаний в ЭС. Выбор модели представления знания часто сводят к обсуждению баланса между декларативным (ДП) и процедурным представлением (ПП). Различие между ДП и ПП можно выразить различием между вопросами «знать, что?» и «знать, как?». Процедурное представление основано на предпосылке, что интеллектуальная деятельность есть знание проблемной среды, вложенное в программы, то есть знание о том, как можно использовать те или иные сущности. Декларативное представление основано на предпосылке, что знание неких сущностей «знать, что?» не имеет глубоких связей с процедурами, используемыми для обработки этих сущностей. При использовании ДП считается, что интеллектуальность базируется на некотором универсальном множестве процедур, обрабатывающих факты любого типа, и на множестве специфических фактов, описывающих частную область знаний. Основное достоинство ДП по сравнению с ПП заключается в том, что в ДП нет необходимости указывать способ использования конкретных фрагментов знания. Простые утверждения могут использоваться несколькими способами, и может оказаться неудобным фиксировать эти способы заранее. Указанное свойство обеспечивает гибкость и экономичность ДП, так как позволяет по-разному использовать одни и те же факты. В ДП знание рассматривается как множество независимых или слабо зависимых фактов, что позволяет осуществлять модификацию знаний и обучение простым добавлением или устранением утверждений. Для ПП проблема модификации значительно сложнее, так как здесь необходимо учитывать, каким образом используется данное утверждение. Однако известно, что существует значительное количество сущностей, которые удобно представить в виде процедур и весьма трудно - в чисто декларативном представлении. Желание использовать достоинства ДП и ПП привело к разработке формализмов, использующих смешанное представление, то есть декларативное представление с присоединенными процедурами (например, фрейм-представление или сети с присоединенными процедурами) или процедурное представление в виде модулей с декларативными образцами. В наиболее совершенном виде эта проблема реализована в объектно-ориентированном подходе.

Модели представления знаний обычно делят на логические (формальные) и эвристические (формализованные) модели. В логических моделях, как правило, используется исчисление предикатов первого порядка (то, что в суждении высказывается о предмете суждения), дополненное рядом эвристических стратегий. Эти методы являются системами дедуктивного типа, то есть в них используется модель получения вывода из заданной системы посылок с помощью фиксированной системы правил вывода. Дальнейшим развитием предикатных систем являются системы индуктивного типа, в которых правила вывода порождаются ЭС на основе обработки конечного числа обучающих примеров. В логических моделях представления знаний отношения, существующие между отдельными единицами знаний, выражаются только с помощью синтаксических правил используемой формальной системы. В отличие от формальных моделей эвристические модели имеют разнообразный набор средств, передающих специфические особенности той или иной проблемной области. Именно поэтому эвристические модели превосходят логические как по возможности адекватно представить проблемную среду, так и по эффективности используемых правил вывода. К эвристическим моделям, используемым в экспертных системах, можно отнести сетевые, фреймовые, продукционные и объектно-ориентированные модели. Следует отметить, что продукционные модели, используемые для представления знаний в экспертных системах, отличаются от формальных продукционных систем тем, что они используют более сложные конструкции правил, а также содержат эвристическую информацию о специфике проблемной среды, выражаемую часто в виде семантических структур.

К типичным моделям представления знаний относятся:

- продукционные модели;
- семантические сети;
- фреймы;
- формальные логические модели (в курсе не рассматриваются).

В свою очередь это множество классов можно разбить на две большие группы:

- модульные (продукционные модели и формальные логические модели) используются для представления поверхностных знаний. Поверхностные знания - знания о видимых взаимосвязях между отдельными событиями и фактами в предметной области;
- сетевые (семантические сети и фреймы) используются для представления глубинных знаний. Глубинные знания - абстракции, аналогии, схемы, отображающие структуру и природу процессов, протекающих в предметной области. Эти знания объясняют явления и могут использоваться для прогнозирования поведения объектов.

Таким образом, модульные языки оперируют отдельными не связанными элементами знаний, к ним относятся правила и аксиомы

предметной области. Сетевые языки дают возможность связывать эти элементы или фрагменты знаний через отношения в семантические сети или сети фреймов. Наиболее распространенным языком представления знаний (ЯПЗ) в ИИС являются продукционные модели. Продукционная модель часто применяется в экспертных системах благодаря своей наглядности, высокой модульности, легкостью внесения дополнений и изменений, простотой механизма логического вывода. Имеется большое число программных средств, реализующих продукционный подход, например, язык высокого уровня CLIPS.

Продукционная модель. Продукционная модель, основанная на правилах, позволяет представить знания в виде предложений типа «если (условие), то (действие)». Под «условием» (антецедентом) понимается некоторое предложение - образец, по которому осуществляется поиск в базе знаний. Под «действием» (консеквентом) понимаются действия, выполняемые при успешном исходе поиска. При этом действия могут быть промежуточными, выступающими далее как условия, и целевыми, завершающими работу системы. Из антецедентов и консеквентов формируются пары атрибут – значение, которые хранятся в рабочей памяти продукционной системы.

Пример правила: если «двигатель не заводится» и «стартер двигателя не работает», то «неполадки в системе электропитания стартера» В этом правиле пары атрибут- значение будут:

двигатель – не заводится;
стартер двигателя – не работает.

Истинность пары атрибут-значение устанавливается в процессе решения конкретной задачи к некоторому текущему моменту времени. В процессе решения задачи содержимое рабочей памяти изменяется. Это происходит по мере срабатывания правил. Правило срабатывает, если при сопоставлении фактов, содержащихся в рабочей памяти, с антецедентом анализируемого правила имеет место совпадение, при этом заключение сработанного правила заносится в рабочую память. В процессе логического вывода объем фактов в рабочей памяти, как правило, увеличивается. Объем фактов в рабочей памяти может уменьшиться в том случае, если действие какого-нибудь правила состоит в удалении фактов из рабочей памяти. В процессе логического вывода каждое правило из базы правил может сработать только один раз. При описании реальных знаний конкретной предметной области может оказаться недостаточным представление фактов с помощью пар атрибут-значение. Более широкие возможности имеет способ описания с помощью триплетов объект-атрибут-значение. В этом случае отдельная сущность предметной области рассматривается как объект, а данные, хранящиеся в рабочей памяти, показывают значения, которые принимают атрибуты этого объекта.

Примеры триплетов:
собака - кличка - Граф;
собака - порода - ризеншнауцер;

собака - окрас - черный.

Одним из преимуществ такого представления знаний является уточнение контекста, в котором применяются правила. Например, правило, относящееся к объекту «собака», должно быть применимо для собак с любыми кличками, всех пород и окрасок. С введением триплетов правила из базы правил могут срабатывать более одного раза в процессе одного логического вывода, поскольку одно правило может применяться к различным экземплярам объекта, но не более одного раза к каждому экземпляру.

Существуют два типа продукционных систем - с прямыми и обратными выводами. Прямые выводы реализуют стратегию «от фактов к заключениям». При обратных выводах выдвигаются гипотезы вероятных заключений, которые могут быть подтверждены или опровергнуты на основании фактов, поступающих в рабочую память. Существуют также системы с двунаправленными выводами.

Основные достоинства продукционных систем связаны с простотой представления знаний и организации логического вывода. К недостаткам систем продукций можно отнести следующие:

- отличие от структур знаний, свойственных человеку;
- неясность взаимных отношений правил;
- сложность оценки целостного образа знаний;
- низкая эффективность обработки знаний.

При разработке небольших систем, состоящих из нескольких десятков правил, проявляются в основном положительные стороны систем продукций, однако при увеличении объема знаний более заметными становятся слабые стороны.

Фреймовая модель. Фреймовая модель представления знаний основана на теории фреймов М. Минского, которая представляет собой систематизированную психологическую модель памяти человека и его сознания. Эта теория имеет весьма абстрактный характер, поэтому только на ее основе невозможно создание конкретных языков представления знаний. Фрейм имеет глубокое психологическое обоснование. Основным преимуществом фреймов как модели представления знаний является то, что она отражает концептуальную основу организации памяти человека, а также ее гибкость и наглядность. Под фреймом понимается абстрактный образ представления объекта или ситуации. Различают фреймы-образцы (прототипы), хранящиеся в базе знаний, и фреймы-экземпляры, которые создаются для отображения фактических ситуаций на основе поступающих данных. Структура фрейма может быть представлена как список свойств (табл. 3).

Приведенные в таблице слоты - это незаполненные значения некоторых атрибутов объекта или ситуации. Дополнительные столбцы «способ получения значения» и «присоединенная процедура» предназначены для описания слотом его значения, и возможного присоединения к тому или иному слоту специальных процедур.

Таблица 3 Структура фрейма

Имя фрейма			
Имя слота	Значение слота	Способ получения значения	Присоединенная процедура

Фрейм имеет имя, служащее для идентификации описываемого им понятия, и содержит ряд описаний - слотов, с помощью которых определяются основные структурные элементы этого понятия. За слотами следуют шпации, в которые помещают данные, представляющие текущие значения слотов. Слот может содержать не только конкретное значение, но также имя процедуры, позволяющей вычислить это значение по заданному алгоритму. Например, слот с именем возраст может содержать имя процедуры, которая вычисляет возраст человека по дате рождения, записанной в другом слоте, и текущей дате. Процедуры, располагающиеся в слотах, называются связанными или присоединенными процедурами. Вызов связанной процедуры осуществляется при обращении к слоту, в котором она помещена. Заполнителями слота могут быть также правила продукций, используемые для определения конкретного значения. В слоте может содержаться не одно, а несколько значений, то есть в качестве структурных составляющих фреймов могут использоваться данные сложных типов, а именно: массивы, списки, множества, фреймы и т. д. Например, в слоте с именем брат может содержаться список имен, если объект, описываемый данным фреймом, имеет нескольких братьев. Значение слота может представлять собой некоторый диапазон или перечень возможных значений, арифметическое выражение, фрагмент текста и т.д. Совокупность данных предметной области может быть представлена множеством взаимосвязанных фреймов, образующих единую фреймовую систему, в которой объединяются декларативные и процедурные знания. Такая система имеет, как правило, иерархическую структуру, в которой фреймы соединены друг с другом с помощью родовидовых связей. На верхнем уровне иерархии находится фрейм, содержащий наиболее общую информацию, истинную для всех остальных фреймов. Фреймы обладают способностью наследовать значения характеристик своих родителей. Например, фрейм африканский слон наследует от фрейма слон значение характеристики цвет «серый». Значение характеристики в дочернем фрейме может отличаться от родительского, например, значением данного слота для фрейма азиатский слон является цвет «коричневый». Над фреймами можно совершать некоторые теоретико-множественные операции, например, объединение и пересечение. При объединении фреймов в результирующем фрейме будут присутствовать все слоты, которые встречались в исходных фреймах. В слотах, не являющихся общими, будут сохранены исходные

значения. Если в объединяемых фреймах были одноименные слоты, в результирующем фрейме останется один слот с таким именем, значение его определится в результате объединения значений одноименных слотов. При пересечении фреймов в результирующем фрейме будут присутствовать только те слоты, которые имелись во всех исходных фреймах. Вычислить результирующие значения можно двумя способами. Первый способ состоит в том, что в результирующем фрейме присутствуют только те значения, которые совпадали в исходных фреймах. Во втором способе результирующие значения находят путем пересечения значений из исходных фреймов.

Фреймовые системы подразделяются на статические и динамические, последние допускают изменение фреймов в процессе решения задачи.

Пример фрейма: руководитель

Имя слота	Значение слота	Тип значения слота
Имя	Иванов И. И.	Строка символов
Рожден	01.01.1965	Дата
Возраст	age(dama, рожден)	Процедура
Специальность	юрист	Строка символов
Отдел	отдел кадров	Строка символов
Зарплата	8000	Строка символов
Адрес	дом_адрес	Фрейм

В общем случае структура данных фрейма может содержать более широкий набор информации, в который входят следующие атрибуты. Имя фрейма служит для идентификации фрейма в системе и должно быть уникальным. Фрейм представляет собой совокупность слотов, число которых может быть произвольным. Число слотов в каждом фрейме устанавливается проектировщиком системы, при этом часть слотов определяется самой системой (системные слоты) для выполнения специфических функций, примерами которых являются:

слот-указатель родителя данного фрейма (is-a);

слот-указатель дочерних фреймов;

слот для ввода имени пользователя;

слот для ввода даты определения фрейма;

слот для ввода даты изменения фрейма и т.д.

Имя слота должно быть уникальным в пределах фрейма. Обычно имя слота представляет собой идентификатор, который наделен определенной семантикой. В качестве имени слота может выступать произвольный текст.

Например, <Имя слота> = Главный герой романа Ф. М. Достоевского «Идиот», <Значение слота> = Князь Мышкин. Имена системных слотов обычно зарезервированы, в различных системах они могут иметь различные значения. Примеры имен системных слотов: is-a, haspart, relations и т. д. Системные слоты служат для редактирования базы знаний и управления выводом во фреймовой системе.

Указатели наследования показывают, какую информацию об атрибутах слотов из фрейма верхнего уровня наследуют слоты с аналогичными именами в данном фрейме. Указатели наследования характерны для фреймовых систем иерархического типа, основанных на отношениях типа «абстрактное - конкретное». В конкретных системах указатели наследования могут быть организованы различными способами, иметь разные обозначения:

U (Unique) - значение слота не наследуется;

S (Same) - значение слота наследуется;

R (Range**)** - значения слота должны находиться в пределах интервала значений, указанных в одноименном слоте родительского фрейма;

O (Override) - при отсутствии значения в текущем слоте оно наследуется из фрейма верхнего уровня, однако в случае определения значения текущего слота оно может быть уникальным. Этот тип указателя выполняет одновременно функции указателей U и S.

Указатель типа данных показывает тип значения слота. Наиболее употребляемые типы: frame - указатель на фрейм; real - вещественное число; integer - целое число; boolean - логический тип; text - фрагмент текста; list - список; table - таблица; expression — выражение; lisp - связанная процедура.

Значение слота должно соответствовать указанному типу данных и условию наследования.

Демоном называется процедура, автоматически запускаемая при выполнении некоторого условия. Демоны автоматически запускаются при обращении к соответствующему слоту. Типы демонов связаны с условием запуска процедуры. Демон с условием if-needed запускается, если в момент обращения к слоту его значение не было установлено. Демон типа if-added запускается при попытке изменения значения слота. Демон if-removed запускается при попытке удаления значения слота. Возможны также другие типы демонов. Демон является разновидностью связанной процедуры.

В качестве значения слота может пользоваться процедура, называемая служебной в языке Лисп или методом в языках объектно-ориентированного программирования. Присоединенная процедура запускается по сообщению, преданному из другого фрейма. Демоны и присоединенные процедуры являются процедурными знаниями, объединенными вместе с декларативными знаниями в единую систему. Эти процедурные знания являются средствами управления выводом во фрейме системах, причем с их помощью можно реализовать любой механизм вывода. Представление таких знаний и заполнение ими интеллектуальных систем - весьма нелегкое дело, которое требует дополнительных затрат труда и времени разработчиков ИИС. Поэтому проектирование фреймовых систем выполняется обычно специалистами, имеющими высокий уровень квалификации в области искусственного интеллекта.

Пример сети фреймов приведен на рис. 7. На нём понятие ученик наследует свойства фреймов ребенок и человек, которые находятся на более высоких уровнях иерархии. Если будет задан вопрос «Любят ли ученики

сладкое?», то следует ответ «да», так как этим свойством обладают все дети, что указано во фрейме ребенок**.** Наследование свойств может быть частичным, например «возраст» для учеников не наследуется из фрейма «ребенок», так как явно указан в собственном фрейме.

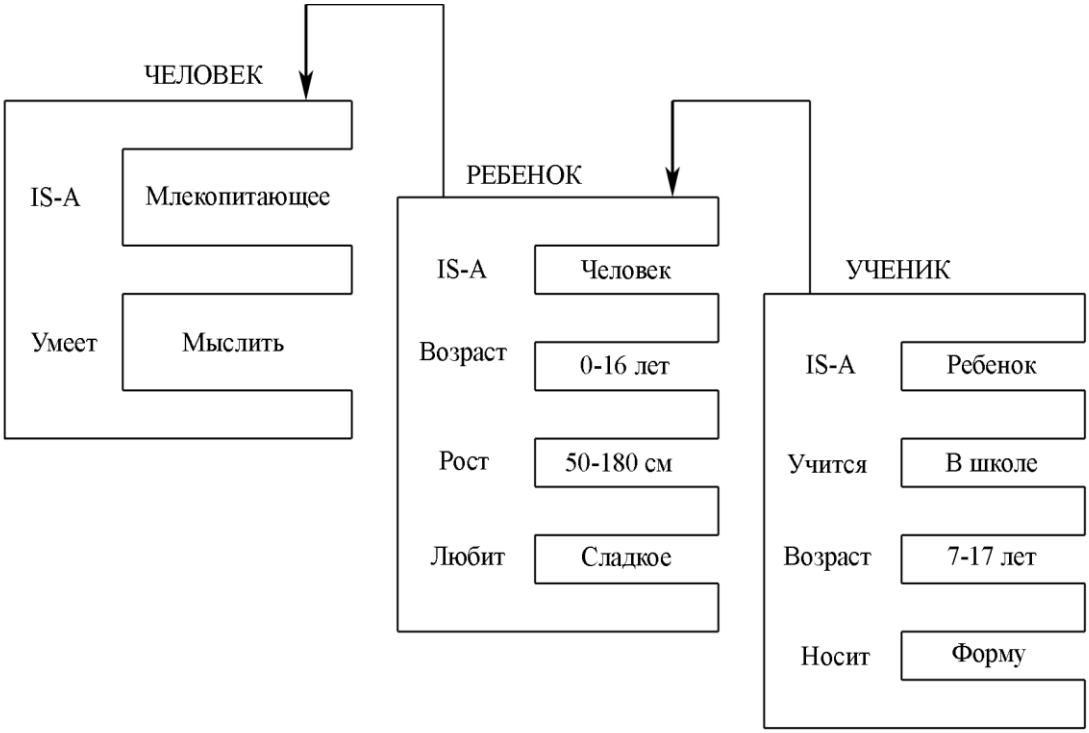


Рисунок 7 - Пример иерархии фреймов

Термин «фреймовый» часто заменяют термином «объектно-ориентированный». Шаблон фрейма можно рассматривать как класс, экземпляр фрейма - как объект. Языки объектно-ориентированного программирования (ООП) предоставляют средства создания классов и объектов, а также средства для описания процедур обработки объектов (методы). Языки ООП не содержащие средств реализации присоединенных процедур не позволяют организовать гибкий механизм логического вывода, поэтому разработанные на них программы либо представляют собой объектно-ориентированные базы данных, либо требуют интеграции с другими средствами обработки знании.

Модель семантические сети. Термин семантическая означает смысловая, а семантика - это наука, устанавливающая отношения между символами и объектами, которые они обозначают, то есть наука, определяющая смысл знаков. Семантическая сеть - это ориентированный граф, вершины которого это понятия, а дуги - отношения между ними.

В семантических сетях используются следующие отношения:

- элемент класса;
- атрибутные связи;
- значение свойства;

- пример элемента класса;
- связи типа «часть-целое»;
- функциональные связи, определяемые глаголами «производит», «влияет»;
- количественные (больше, меньше, равно …);
- пространственные (далеко от, близко от, за, под, над …);
- временные (раньше, позже, в течение…);
- логические связи (и, или, не) и др.

Минимальный состав отношений в семантической сети - это элемент класса, атрибутные связи и значение свойства.

Проблема поиска решения в базе знаний типа семантической сети сводится к задаче поиска фрагмента сети или подсети, соответствующей поставленному вопросу. На рисунке 8 изображен пример семантической сети.

Для реализации семантических сетей в экспертных системах существуют специальные сетевые языки. Систематизация отношений конкретной семантической сети зависит от специфики знаний предметной области и является сложной задачей. Особого внимания заслуживают общезначимые отношения, присутствующие во многих предметных областях. Именно на таких отношениях основана концепция семантической сети. В семантических сетях, так же как при фреймовом представлении знаний, декларативные и процедурные знания не разделены, следовательно, база знаний не отделена от механизма вывода. Процедура логического вывода обычно представляет совокупность процедур обработки сети. Семантические сети получили широкое применение в экспертных системах.

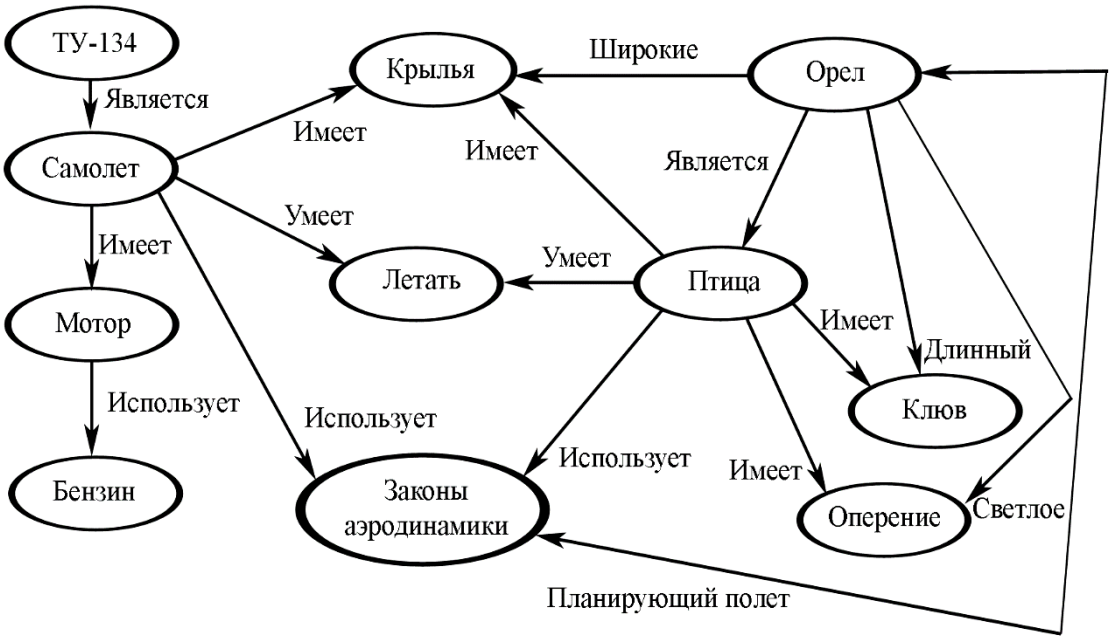


Рисунок 8 - Семантическая сеть, показывающая взаимоотношения птицы и самолета

5 Способы реализации логического вывода в ЭС с классическими моделями представления знаний

Прямой и обратный вывод в ЭС продукционного типа. Любая экспертная система продукционного типа должна содержать три основные компоненты: базу правил, рабочую память и механизм вывода. База правил (БП) - формализованные с помощью правил продукций знания о конкретной предметной области. Рабочая память (РП) - область памяти, в которой хранится множество фактов, описывающих текущую ситуацию, и все пары атрибут-значение, которые были установлены к определенному моменту. Содержимое РП в процессе решения задачи обычно изменяется, увеличиваясь в объеме по мере применения правил. Другими словами, РП - это динамическая часть базы знаний, содержимое которой зависит от окружения решаемой задачи. В простейших ЭС хранимые в РП факты не изменяются в процессе решения задачи, однако существуют системы, в которых допускается изменение и удаление фактов из РП. Это системы с немонотонным выводом, работающие в условиях неполноты информации.
Механизм вывода выполняет две основные функции:

- просмотр существующих в рабочей памяти фактов и правил из БП, а также добавление в РП новых фактов;
- определение порядка просмотра и применения правил. Порядок может быть прямым или обратным.

Прямой порядок - от фактов к заключениям. В экспертных системах с прямыми выводами по известным фактам отыскивается заключение, которое из этих фактов следует. Если такое заключение удается найти, оно заносится в рабочую память. Прямые выводы часто применяются в системах диагностики, их называют выводами, управляемыми данными.

Обратный порядок вывода - от заключений к фактам. В системах с обратным выводом вначале выдвигается некоторая гипотеза о конечном суждении, а затем механизм вывода пытается найти в рабочей памяти факты, которые могли бы подтвердить или опровергнуть выдвинутую гипотезу. Процесс отыскания необходимых фактов может включать достаточно большое число шагов, при этом возможно выдвижение новых гипотез (целей). Обратные выводы управляются целями.

Для выполнения указанных функций механизм вывода включает компоненту вывода и управляющую компоненту. Действие компоненты вывода основано на применении правила логического вывода Modus Ponendo Ponens. Суть применения этого правила в продукционных системах состоит в следующем. Если в РП присутствует истинный факт А и в базе правил существует правило вида «если А, то В», то факт В признается истинным и заносится в рабочую память. Такой вывод легко реализуется на ЭВМ, однако при этом часто возникают проблемы, связанные с распознаванием значений слов, а также с тем, что факты могут иметь внутреннюю структуру, и между элементами этой структуры возможны различного рода связи. Например,

пусть имеется факт А - «автомобиль Иванова - белый» и правило «если автомобиль - белый, то автомобиль легко заметить ночью». Человек легко выведет заключение «автомобиль Иванова легко заметить ночью», но это не под силу ЭС чисто продукционного типа. Она не сможет сформировать такое заключение, потому что А не совпадает точно с антецедентом правила. Кроме того, невысокая интеллектуальная мощность продукционных систем обусловлена тем, что человек выводит заключения, имея в своем распоряжении все свои знания, то есть БЗ огромного объема, в то время как ЭС способны вывести сравнительно небольшое количество заключений, используя заданное множество правил. Из сказанного можно сделать вывод о том, что компонента вывода в ЭС должна быть организована так, чтобы быть способной функционировать в условиях недостатка информации.

Управляющая компонента определяет порядок применения правил, а также устанавливает, имеются ли еще факты, которые могут быть изменены в случае продолжения работы (при немонотонном выводе). Механизм вывода работает циклически, при этом в одном цикле может сработать только одно правило. Схема цикла приведена на рисунке 9.

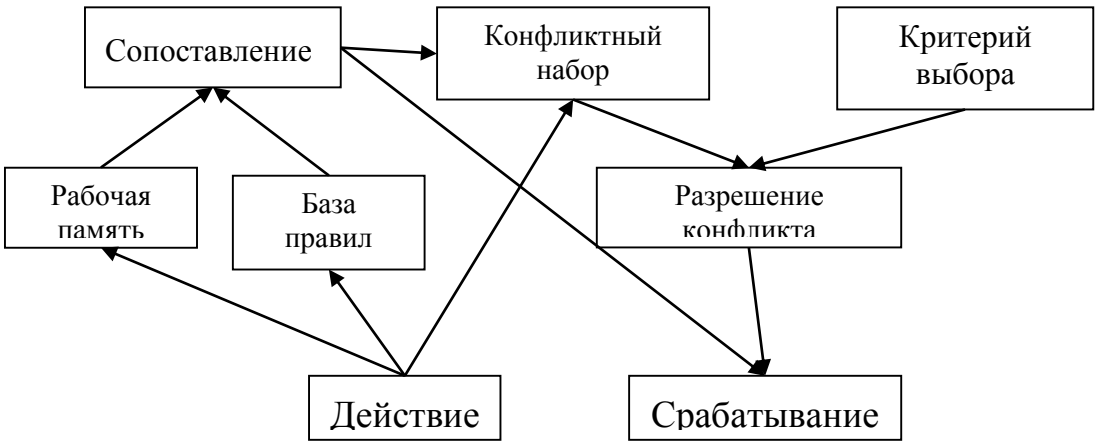


Рисунок 9 - Цикл работы вывода

В цикле выполняются следующие основные операции:

- сопоставление - образец (антецедент) правила сравнивается с имеющимися в РП фактами;
- разрешение конфликтного набора - выбор одного из нескольких правил в том случае, если их можно применить одновременно;
- срабатывание правила - в случае совпадения образца некоторого правила из базы правил с фактами, имеющимися в рабочей памяти, происходит срабатывание правила, при этом оно отмечается в БП;
- действие - изменение содержимого РП путем добавления туда заключения сработавшего правила. Если в заключении содержится

директива на выполнение некоторой процедуры, последняя выполняется.

Поскольку механизм вывода работает циклически, следует знать о способах завершения цикла. Традиционными способами являются либо исчерпание всех правил из БП, либо выполнение некоторого условия, которому удовлетворяет содержимое рабочей памяти (например, появление в ней какого-то образца), либо комбинация этих способов. Особенностью ЭС является то, что они не располагают процедурами, которые могли бы построить в пространстве состояний сразу весь путь решения задачи. Траектория поиска решения полностью определяется данными, получаемыми от пользователя в процессе логического вывода.

Рассмотрим простейшие примеры прямого и обратного вывода в системах продукционного типа.

Пример прямого вывода. Пусть в БП имеются следующие правила:

Правило 1. «если двигатель не заводится, и фары не горят, то сел аккумулятор».

Правило 2. «если указатель бензина находится на нуле, то двигатель не заводится».

Предположим, что в рабочую память от пользователя ЭС поступили факты: фары не горят и указатель бензина находится на нуле. Рассмотрим основные шаги алгоритма прямого вывода.

1. Сопоставление фактов из РП с образцами правил из БП. Правило 1 не может сработать, а правило 2 срабатывает, так как "образец, совпадающий с его антецедентом, присутствует в РП.
2. Действие сработавшего правила 2. В РП заносится заключение этого правила, то есть образец: двигатель не заводится.
3. Второй цикл сопоставления фактов в РП с образцами правил. Теперь срабатывает правило 1, так как совпадение условий в его антецеденте становится истинной.
4. Действие правила 1, которое заключается в выдаче пользователю окончательного диагноза - сел аккумулятор.
5. Конец работы (БП исчерпана).

Пример прямого вывода с конфликтным набором. Теперь допустим, что в БП кроме правила 1 и правила 2 присутствует правило 3: «если указатель бензина находится на нуле, то нет бензина». В РП находятся те же факты, что в предыдущем примере. В результате сопоставления в первом же цикле возможно применение двух правил - правила 2 и правила 3, то есть возникает конфликтный набор и встает задача выбора: какое из этих правил применить первым. Если выберем правило 2, то в РП добавится факт «двигатель не заводится» и на следующем шаге опять возникнет конфликтный набор, так как можно будет применить правило 1 и правило 3. Если будет выбрано правило 1, то к заключению «сел аккумулятор» придем за два шага. При любом другом выборе порядка применения правил к этому же заключению приходим за три шага. Если завершение цикла работы ЭС наступает после просмотра всех

правил, то число шагов будет равно трем, причем порядок применения правил не будет иметь какого-либо значения.

Пример обратного вывода**.** Предположим, что в БП имеется два правила (правило 1 и правило 2), а в РП - те же факты, что в предыдущих примерах с прямым выводом.

Алгоритм обратного вывода содержит следующие шаги.

1. Выдвигается гипотеза окончательного диагноза - сел аккумулятор.
2. Отыскивается правило, заключение которого соответствует выдвинутой гипотезе, в нашем примере - это правило 1.
3. Исследуется возможность применения правила 1, то есть решается вопрос о том, может ли оно сработать. Для этого в рабочей памяти должны присутствовать факты, совпадающие с образцом этого правила. В рассматриваемом примере правило 1 не может сработать из-за отсутствия в РП образца «двигатель не заводится». Этот факт становится новой целью на следующем шаге вывода.
4. Поиск правила, заключение которого соответствует новой цели. Такое правило есть - правило 2.
5. Исследуется возможность применения правила 2 (сопоставление). Оно срабатывает, так как в РП присутствует факт, совпадающий с его образцом.
6. Действие правила 2, состоящее в занесении заключения «двигатель не заводится» в РП.
7. Условная часть правила 1 теперь подтверждена фактами, следовательно, оно срабатывает, и выдвинутая начальная гипотеза подтверждается.
8. Конец работы.

При сравнении этого примера с примером прямого вывода нельзя заметить преимуществ обратных выводов перед прямыми выводами.

Пример обратного вывода с конфликтным набором**.** Предположим, что в БП записаны правило 1, правило 2, правило 3 и правило 4: «если засорился бензонасос, то двигатель не заводится». В РП присутствуют те же самые факты: «фары не горят и указатель бензина находится на нуле». В данном случае алгоритм обратного вывода с конфликтным набором включает следующие шаги.

1. Выдвигается гипотеза сел аккумулятор.
2. Поиск правила, заключение которого совпадает с поставленной целью. Это правило 1.
3. Исследуется возможность применения правила 1. Оно не может сработать, тогда выдвигается новая подцель «двигатель не заводится», соответствующая недостающему образцу.
4. Поиск правил, заключения которых совпадают с новой подцелью. Таких правил два: правило 2 и правило 4. Если выберем правило 2, то дальнейшие шаги совпадают с примером без конфликтного набора. Если выберем правило 4, то оно не сработает, так как в РП нет образца: «засорился

бензонасос». После этого будет применено правило 2, что приведет к успеху, но путь окажется длиннее на один шаг.

Следует обратить внимание на то, что правило 3, не связанное с поставленной целью, вообще не затрагивалось в процессе вывода. Этот факт свидетельствует о более высокой эффективности обратных выводов по сравнению с прямыми выводами. При обратных выводах существует тенденция исключения из рассмотренных правил тех правил, которые не имеют отношения к поставленной цели.

В ЭС продукционного типа все множество знаний обычно хранится в виде древовидной структуры, называемой И-ИЛИ-графом. Фрагменты такой структуры приведены на рисунках 10, 11.

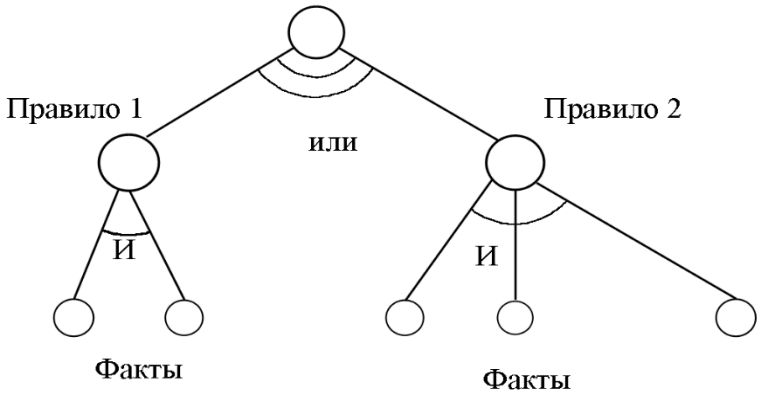


Рисунок10 - Простейший фрагмент структуры И-ИЛИ-графа

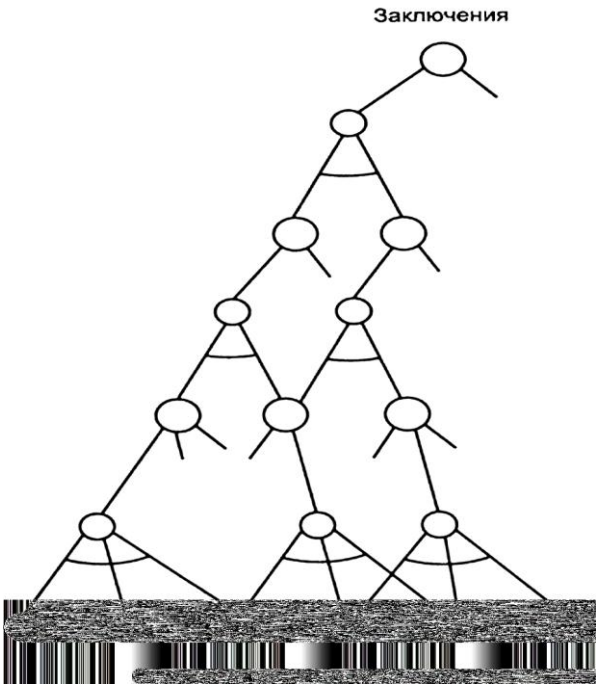


Рисунок 11 - Фрагмент структуры И-ИЛИ – графа продукционной ЭС

Классическая форма продукций предполагает наличие в антецеденте только связки И. На практике классическая форма может быть расширена введением связки ИЛИ в условную часть либо включением в антецедент

вычислений на основании содержимого рабочей памяти и т.п. Если существует множество правил, из которых выводится одно и то же заключение, то, выполнив операцию ИЛИ (дизъюнкцию) над всеми заключениями, полученными с помощью этих правил, можно показать отношение между результатом отдельного вывода и данными, на основании которых делается вывод.

С помощью И-ИЛИ-графа обратный вывод в ЭС продукционного типа можно представить как проблему поиска определенного пути на графе. Выбор одной из связок ИЛИ соответствует разрешению конфликтного набора, при этом не безразличен порядок оценки условий в антецеденте, соединенных связкой. В экспертных системах, имеющих практическую ценность, следует знать, с помощью каких стратегий управления выводом можно минимизировать время решения задач.

Стратегия поиска в глубину. При выборе очередной подцели в процессе обратного вывода предпочтение всегда отдается той, которая соответствует следующему, более детальному уровню описания задачи. Например, система диагностики, сделав на основании известных симптомов предположение о причинах неисправности, будет запрашивать уточняющие признаки и симптомы до тех пор, пока полностью не подтвердит (опровергнет) выдвинутую гипотезу. Пример организации поиска в глубину показан на рис. 12, где цифрами обозначены номера шагов поиска.

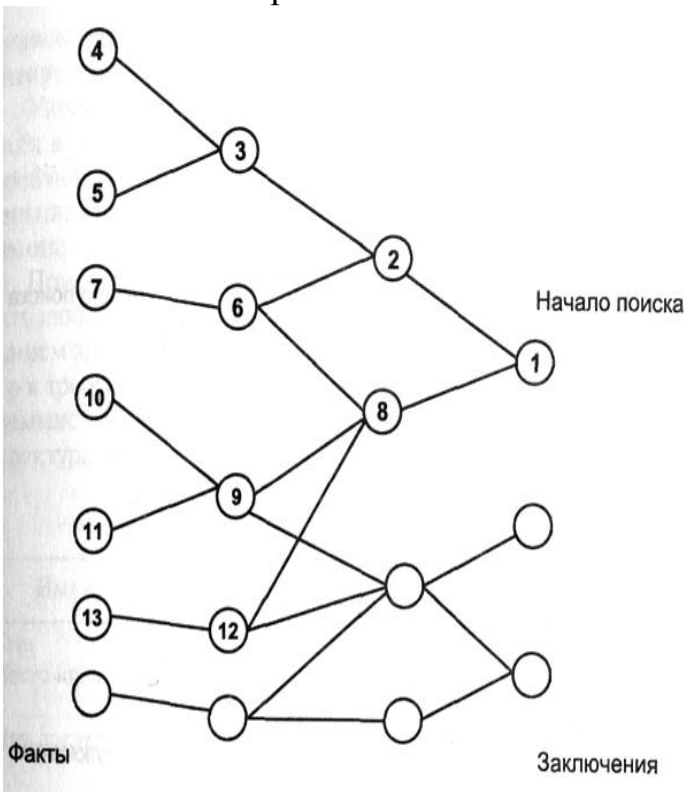


Рисунок 12 - Поиск в глубину при обратном выводе

Стратегия поиска в ширину. При поиске в ширину сначала анализируются все симптомы (факты), находящиеся на одном уровне пространства состояний задачи, даже если они относятся к разным целям

(подцелям), и только после этого происходит переход к поиску симптомов следующего уровня. На рис. 13 показаны шаги поиска в ширину, обозначенные номерами, указанными в вершинах. На рисунке представлена стратегия обратного вывода на том же И-ИЛИ-графе, который приведен и на рис. 12 Алгоритм поиска в глубину более эффективен в отношении времени поиска и обработки знаний, однако он характеризуется более высоким риском потери перспективных решений по сравнению с поиском в ширину.

Разбиение на подзадачи. Декомпозиция дает положительный эффект только для хорошо структурированных областей знаний, так как применение этой стратегии основано на правильном понимании сущности задачи и возможности ее представления в виде системы иерархически связанных целей-подцелей, причем разбиение на подзадачи необходимо выполнить оптимальным способом.

$\alpha - \beta$ - алгоритм.

С помощью этого алгоритма исходная задача сводится к уменьшению пространства состояний путем удаления в нем ветвей, неперспективных для поиска успешного решения, то есть просматриваются только те вершины, в которые можно попасть в результате следующего шага, после чего неперспективные направления исключаются. Например, в БЗ продукционной системы, заполненной знаниями о животном мире, не следует искать животных, не относящихся к млекопитающим, в направлении, берущем начало от вершины, определяющей млекопитающих. Данная стратегия является определенным компромиссом между поиском в ширину и поиском в глубину. Для ее успешной реализации следует располагать дополнительными эвристическими знаниями, которые используются при выборе перспективных направлений.

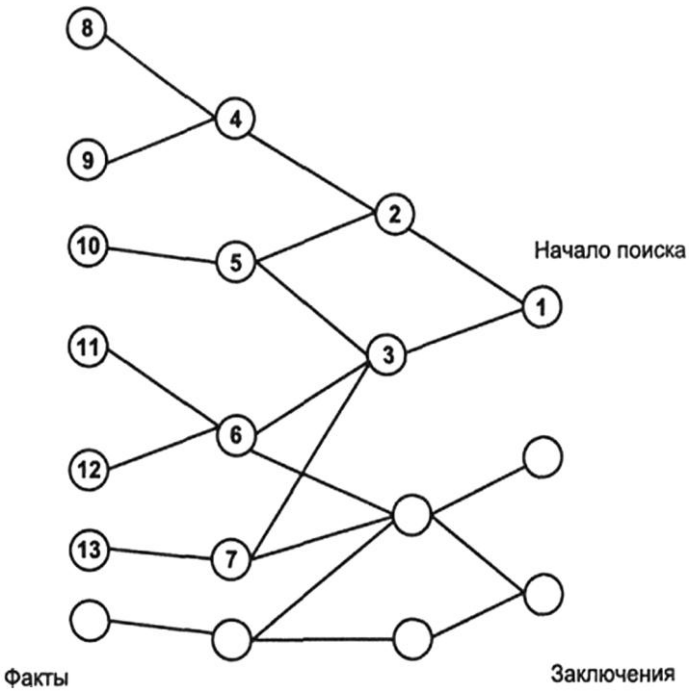


Рисунок 13 - Поиск в ширину при обратном выводе

Обработка знаний в ЭС с фреймовым представлением. В ЭС с фреймовой моделью представления знаний используются три способа управления логическим выводом:

- демоны,
- присоединенные процедуры
- механизм наследования.

Для фреймовых объектно-ориентированных систем механизм наследования является единственным основным механизмом вывода. Управленческие функции механизма наследования заключаются в автоматическом поиске и определении значений слотов фреймов нижележащих уровней по значениям слотов фреймов верхних уровней, а также в запуске присоединенных процедур и демонов.

Рассмотрим обработку знаний в ЭС на примере структуры фрейма «Научная конференция» (табл. 4.).

Таблица 4 Фрейм «Научная конференция»

Имя слота	Значение слота	If- needed	If-added	If-removed
Дата	25.02. 10:00			
Место проведения	Аудитория 6 - 332		Заказ	
Тема доклада	Алгоритм отжига			
Докладчик	Семенов А.И.	Кто?		

Демон «Заказ» - это процедура, которая автоматически запускается при попытке подстановки значения в слот с именем место проведения. Главное назначение этой процедуры состоит в проверке возможности заказа аудитории на нужное время. Такая процедура на языке LISP может выглядеть примерно так:

LISP proc «Заказ» (название конференции, место проведения, дата)
if возможно (место проведения, дата)
then заказать (название конференции, место проведения, дата)
else сообщение («Заказ невозможен», название конференции)
end.

Демон «Кто?» автоматически запускается при обращении к слоту «Докладчик», если значение этого слота не определено. Основное содержание данной процедуры заключается в генерации запроса к пользователю типа «Кто выступает?», получение ответа и его запись в качестве значения слота. Реализация вывода с помощью присоединенных процедур требует наличия механизма обмена информацией между фреймами. В качестве такого механизма обычно используется механизм сообщений. На рисунке 14 схематично показан обмен информацией между фреймами АА и ВВ во время

исполнения присоединенной процедуры CALC, при этом вызывается процедура MEAN, расположенная во фрейме BB.

Фрейм

Имя слота	Тип	Значение
ISA	Frame	A
Объект 1	Integer	125
Объект 2	Real	8.5
……	……	……
Расчет	LISP	CALC

MSG →

← Ответ

Фрейм BB

Имя слота	Тип	Значение
ISA	Frame	B
X	Integer	10, 20, 30
У	Real	8.0, 12.6
……	……	……
Среднее	LISP	MEAN

Рисунок 14 - Обмен информацией между фреймами AA и BB

Допустим, что процедура CALC (result) выполняет расчет, в процессе которого происходит обращение к фрейму BB с использованием команды MSG, реализующей передачу сообщения в другой фрейм.

LISP proc CALC (result) MSG (Cpednee, BB, X) end.

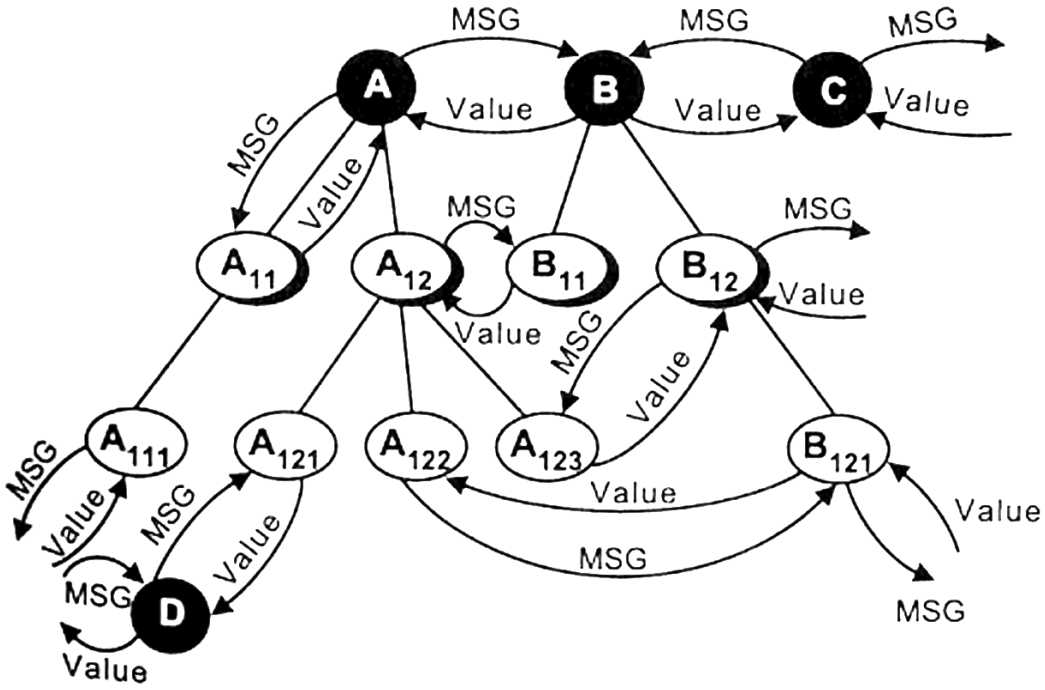
Команда MSG имеет три параметра:

- имя слота, к которому происходит обращение (в данном случае значением слота «Среднее» является присоединенная процедура MEAN);
- имя фрейма, в котором содержится необходимая информация (BB);
- имя слота - параметра, в котором находятся данные для расчета (X).

Запуск процедуры CALC вызовет исполнение следующих действий: передача сообщения во фрейм BB на запуск процедуры MEAN, которая найдет среднее арифметическое чисел, записанных в слоте X; вычисленное значение будет записано в переменную result и передано в CALC как ответ на сообщение MSG.

Таким образом, в ЭС с фреймовым представлением знаний невозможно четко отделить процедурные знания от декларативных знаний. Присоединенные процедуры и демоны одновременно являются знаниями и средствами управления логическим выводом.

На рисунке 15 схематично показаны средства управления выводом во фреймовой системе. Возможность организации выводов любого типа является существенным преимуществом фреймовых систем по сравнению с продукционными системами. Не менее важным достоинством является большее сходство этой модели представления знаний со структурой знаний в памяти человека. Вместе с тем практическая реализация фреймовых систем сопряжена со значительной трудоемкостью, как на этапе проектирования, так и при реализации. Поэтому стоимость промышленных экспертных систем фреймового типа на порядок превосходит стоимость продукционных систем.



Рисунке15 - Средства управления выводом в ЭС фреймового типа

6 Методы приобретения знаний

Основные аспекты процесса извлечения знаний. Извлечением знаний называют процесс получения знаний от экспертов. Извлечение знаний – это сложная и трудоемкая процедура, в результате которой инженеру по знаниям необходимо создать собственную модель предметной области на основе информации, полученной от экспертов. Процесс извлечения знаний рассматривают в трех основных аспектах: психологическом, лингвистическом и гносеологическом (рис. 16).

Психологический аспект самый важный из всех аспектов, так как извлечение знаний происходит в процессе общения инженера по знаниям с экспертами, где психология играет доминирующую роль. Процесс извлечения знаний для интеллектуальных систем необходимо организовать не как однонаправленный процесс передачи сообщений от эксперта аналитику, а как совместный поиск истины. В процессе разговорного общения много

информации теряется, поэтому важной проблемой является увеличение информативности общения аналитика и эксперта за счет использования методик, выработанных в психологии (рис. 17).

Модель общения включает участников общения, средства общения и предмет общения (знания). В соответствии с этими компонентами выделяются три слоя психологических проблем: контактный, процедурный, когнитивный.

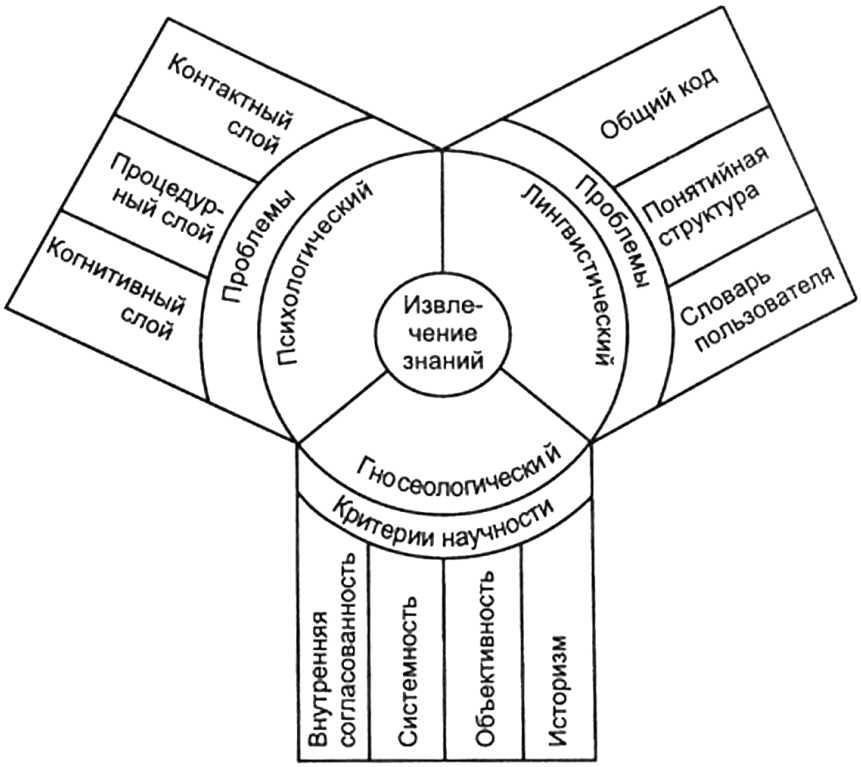


Рисунок 16 - Аспекты извлечения знаний

Степень информативности общения аналитика и эксперта на уровне контактного слоя зависит в основном от пола, возраста, темперамента личности и мотивации участников общения. Установлено, что хорошие результаты дают гетерогенные пары (мужчина/женщина) и соотношение возрастов:

$$5 \le (B_Э - B_A) \le 20,$$

где $B_Э$ — возраст эксперта; B_A - возраст аналитика.

Желательно, чтобы участники процесса общения обладали следующими качествами: доброжелательностью, хорошей памятью, вниманием, наблюдательностью, воображением, впечатлительностью, собранностью, настойчивостью, общительностью и находчивостью.

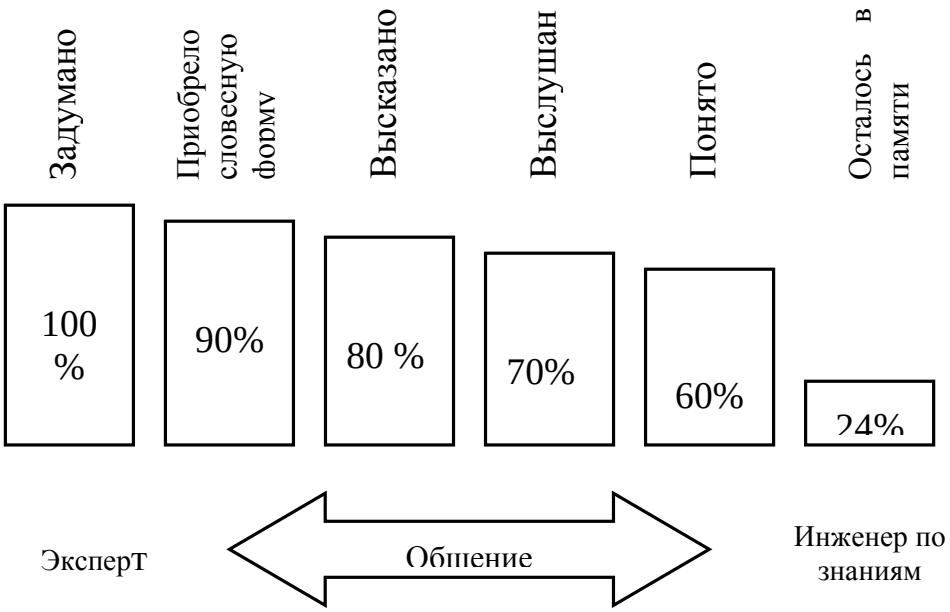


Рисунок 17 - Потери информации при разговорном общении

В рамках контактного слоя наиболее предпочтительными из четырех классических типов темперамента являются сангвиники и холерики.

Параметры процедурного слоя обеспечивают эффективность извлечения знаний. К этим параметрам относятся: ситуация общения (место, время, продолжительность); оборудование (вспомогательные средства, освещенность, мебель); профессиональные приемы (темп, стиль, методы и др.). Для повышения эффективности процесса извлечения знаний инженер по знаниям должен подобрать значимые для эксперта стимулы, так как эксперт передает аналитику один из самых ценных ресурсов - знания.

Когнитивный слой связан с изучением семантического пространства памяти эксперта и с воссозданием его понятийной структуры и модели рассуждений. Желательно, чтобы и аналитики, и эксперты обладали следующими когнитивными характеристиками:

- высокой поленезависимостью, которая подразумевает способность выделять главные аспекты рассматриваемой проблемы и отбрасывать все лишнее, что не относится к поставленной задаче;
- рефлексивностью, характеризующей склонность к рассудительности и самоанализу (в то время как импульсивность характеризуется быстрым, зачастую недостаточно обоснованным принятием решений);
- когнитивной эквивалентностью, определяющей способность человека к различению понятий и разбиению их на классы и подклассы;
- эксперты - устойчивостью представлений, то есть жесткостью структуры восприятия, а аналитики - гибкостью, то есть умением легко приспосабливаться к новой обстановке.

Для эффективного построения ИИС инженер по знаниям должен владеть специальными неформальными методами и математическим аппаратом, позволяющими ему воссоздавать полученные от эксперта знания с помощью различных моделей, например, таких, как продукционная или фреймовая. Не навязывая эксперту какой-либо модели, аналитик должен подобрать средства представления знаний, имеющие максимально высокую семантическую репрезентативность (представительность).

Лингвистический аспект определяется тем, что язык является основным средством общения в процессе извлечения знаний. В области лингвистических проблем наиболее важными являются понятия: общий код, понятийная структура, словарь пользователя.

Общим кодом называют специальный промежуточный язык общения между экспертом и инженером по знаниям. Этот язык включает совокупность общенаучных и специальных понятий из профессиональной литературы, элементов базового языка, неологизмов и др. Общий код позволяет преодолеть языковой барьер в процессе общения инженеров по знаниям с экспертами. Выработка общего кода для партнеров осуществляется в соответствии с информационными потоками, представленными на рис. 18. В дальнейшем общий код преобразуется в понятийную структуру, или семантическую сеть, которая связывает понятия, хранящиеся в памяти человека. Выявление отношений между элементами знаний, представленных понятиями, является одной из самых сложных проблем в процессах извлечения знаний.

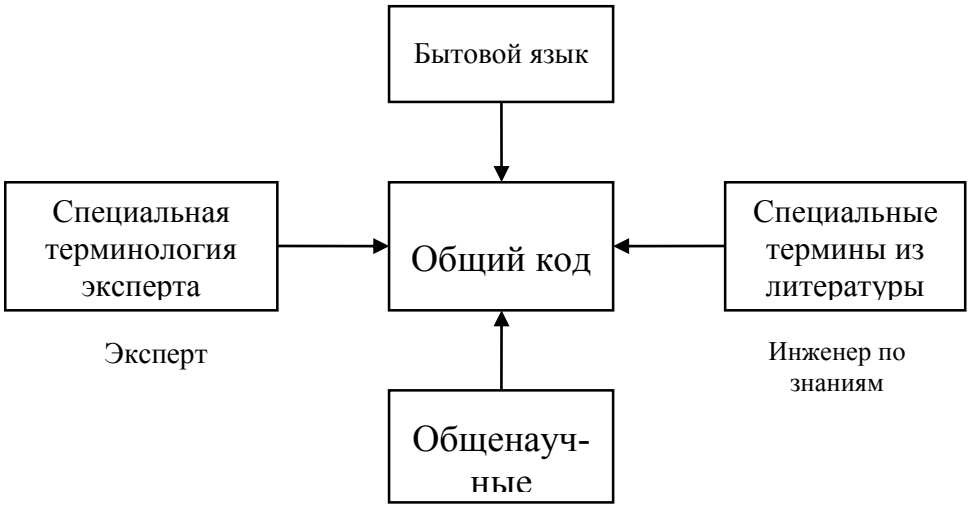


Рисунок 18 - Структура формирования общего кода

Разработка словаря пользователя необходима в связи с тем, что конечный пользователь не обязан владеть профессиональным языком предметной области, который использовался при построении БЗ. Интерфейс пользователя создается, как правило, путем доработки словаря общего кода.

Гносеологический аспект объединяет методологические проблемы получения нового научного знания, так как процесс познания часто

сопровождается появлением новых понятий и теорий. В процессе разработки БЗ эксперты довольно часто впервые формулируют некоторые закономерности на основе накопленного эмпирического опыта. Например, последовательность факт → обобщенный факт → эмпирический закон → теоретический закон называется гносеологической цепочкой. Теория - это не только система обобщения накопленных знаний, но и способ получения нового знания.

В процессе извлечения знаний инженеров по знаниям прежде всего интересуют эмпирические знания экспертов, представляющие собой результаты наблюдений, которые могут оказаться несогласованными. Внутренняя согласованность эмпирических знаний характеризуется понятиями модальности, противоречивости, неполноты. Под модальностью знания понимается возможность его существования в различных категориях. Противоречивость является естественным свойством эмпирических знаний и не всегда может и должна быть устранена. Она может служить отправной точкой в рассуждениях экспертов. Неполнота знаний связана с невозможностью исчерпывающего описания любой предметной области.

На начальных этапах инженер по знаниям, исследуя структуру умозаключений эксперта, может использовать разные теории и подходы для построения формальной модели знаний предметной области. Наиболее известными и часто применяемыми приемами являются математическая логика, теория ассоциаций. Математическая логика формирует критерии, которые гарантируют точность, значимость и непротиворечивость общих понятий, рассуждений и выводов. В теории ассоциаций мышление представляется в виде цепочки идей, связанных общими понятиями. Здесь применяются следующие приемы: ассоциации, приобретенные на основе связей различной природы; привлечение прошлого опыта; метод проб и ошибок со случайным успехом; привычные («автоматические») реакции и др.

Методы извлечения знаний. Многообразие задач, ситуаций и источников знаний обусловило появление большого количества методов извлечения, приобретения и формирования знаний. На первом уровне возможной классификации методов извлечения знаний (рис. 19) выделены два больших класса. Первый класс образуют коммуникативные методы, которые ориентированы на непосредственный контакт инженера по знаниям с экспертом (источником знаний), второй класс - текстологические методы, основанные на приобретении знаний из документов и специальной литературы.

Коммуникативные методы разделяются на пассивные и активные. В пассивных методах ведущую роль играет эксперт, а в активных - инженер по знаниям. При решении конкретных задач, как правило, используются как пассивные, так и активные методы. Активные методы делятся на индивидуальные и групповые.

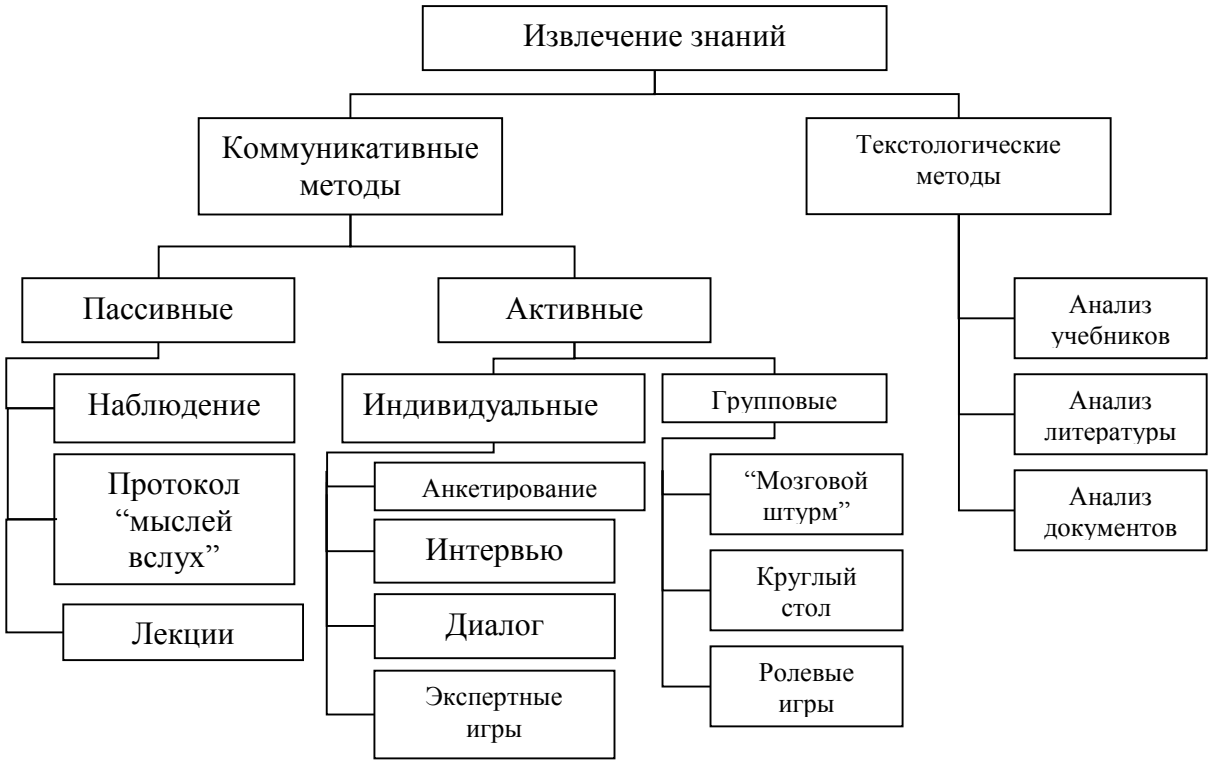


Рисунок 19 - Классификация методов извлечения знаний

В групповых методах знания получают от множества экспертов, а в индивидуальных - от единственного эксперта. Индивидуальные методы получили более широкое применение на практике по сравнению с групповыми.

Пассивные коммуникативные методы включают наблюдение, анализ протоколов «мыслей вслух», процедуры извлечения знаний из лекций.

Метод наблюдения является одним из наиболее применяемых на начальных этапах разработки экспертных систем. Его суть заключается в фиксировании всех действий эксперта, его реплик и объяснений. При этом инженер по знаниям не вмешивается в работу эксперта, а только наблюдает за процессом решения реальных задач, либо за решением проблем, имитирующих реальные задачи. Наблюдения за процессом решения реальных задач позволяют инженеру по знаниям глубже понять предметную область. Однако эксперт в этом случае испытывает большое психологическое напряжение, понимая, что осуществляет не только свою профессиональную деятельность, но и демонстрирует ее инженеру по знаниям. Наблюдение за имитацией процесса снимает это напряжение, но приводит к снижению полноты и качества извлекаемых данных. Наблюдения за имитацией незаменимы в тех случаях, когда наблюдения за реальным процессом невозможны из-за специфики изучаемой предметной области.

Метод анализа протоколов «мыслей вслух» отличается от метода наблюдения тем, что эксперт не только комментирует свои действия, но и

объясняет цепочку своих рассуждений, приводящих к решению. Основной проблемой, возникающей при использовании этого метода, является принципиальная сложность для любого человека словесного описания собственных мыслей и действий. Повысить полноту и качество извлекаемых знаний можно за счет многократного уточняющего протоколирования рассуждений эксперта.

Метод извлечения знаний из лекций предполагает, что эксперт передает свой опыт инженеру по знаниям в форме лекций. При этом инженер по знаниям может заранее сформулировать темы лекций. Если этого не удается сделать, то инженер по знаниям конспектирует лекции и задает вопросы. Качество информации, предоставленной экспертом в ходе лекции, определяется четкостью сформулирован-ной темы, а также способностями лектора в структурировании и изложении своих знаний и рассуждений.

Предметные области отличаются уровнем документированности и структурированности. Для характеристики предметной области по уровню документированности выделяют три класса: хорошо документированные, среднедокументированные и слабодокументированные области (рис. 20). По степени структурированности знаний предметные области могут быть:

- хорошо структурированными (с четкой аксиоматизацией, широким применением математического аппарата, устоявшейся терминологией);
- среднеструктурированными (с определившейся терминологией, развивающейся теорией, явными взаимосвязями между явлениями);
- слабоструктурированными (с размытыми определениями, богатым эмпирическим материалом, скрытыми взаимосвязями).

Активные индивидуальные методы включают методы анкетирования, интервьюирования, свободного диалога и игры с экспертом. Преимуществом методов анкетирования является то, что анкета или вопросник составляются инженером по знаниям заранее и используются для опроса экспертов. Составление анкеты следует проводить с учетом рекомендаций, выработанных в социологии и психологии. Основными требованиями к анкетам являются следующие:

1. Анкета не должна быть монотонной и однообразной, чтобы не вызывать скуку или усталость. Для этого необходимо разнообразить тематику и форму задания вопросов, применить стиль игры;
2. Анкета должна быть приспособлена к языку эксперта;
3. Следует учитывать, что вопросы влияют друг на друга, поэтому важно расположить их в правильной последовательности.
4. В анкете должно содержаться оптимальное число избыточных вопросов, часть которых предназначена для контроля правильности ответов, а другая часть - для снятия напряжения.

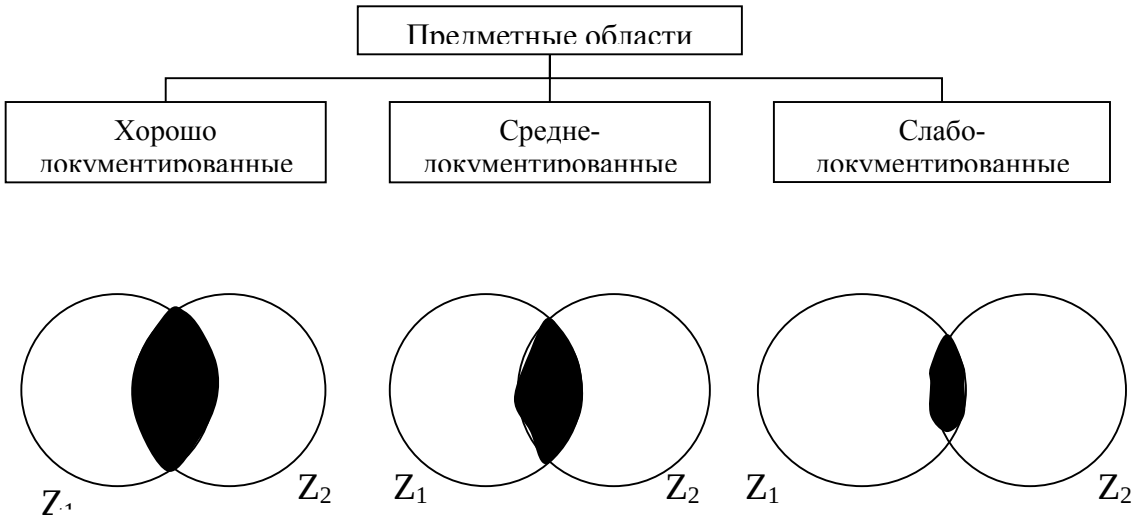


Рисунок 20 - Классификация предметных областей по уровню документированности:

Z_1 – знание эксперта; Z_2 – материализованное в книгах «общее» знание; $Z_{по} - Z_1 \cup Z_2$ – знания предметной области.

Метод интервьюирования отличается от метода анкетирования тем, что позволяет аналитику опускать ряд вопросов в зависимости от ситуации, вставлять новые вопросы в анкету, изменять темы и разнообразить ситуацию общения. Важную роль в методе интервьюирования играют вопросы, классификация которых показана на рисунке 21.

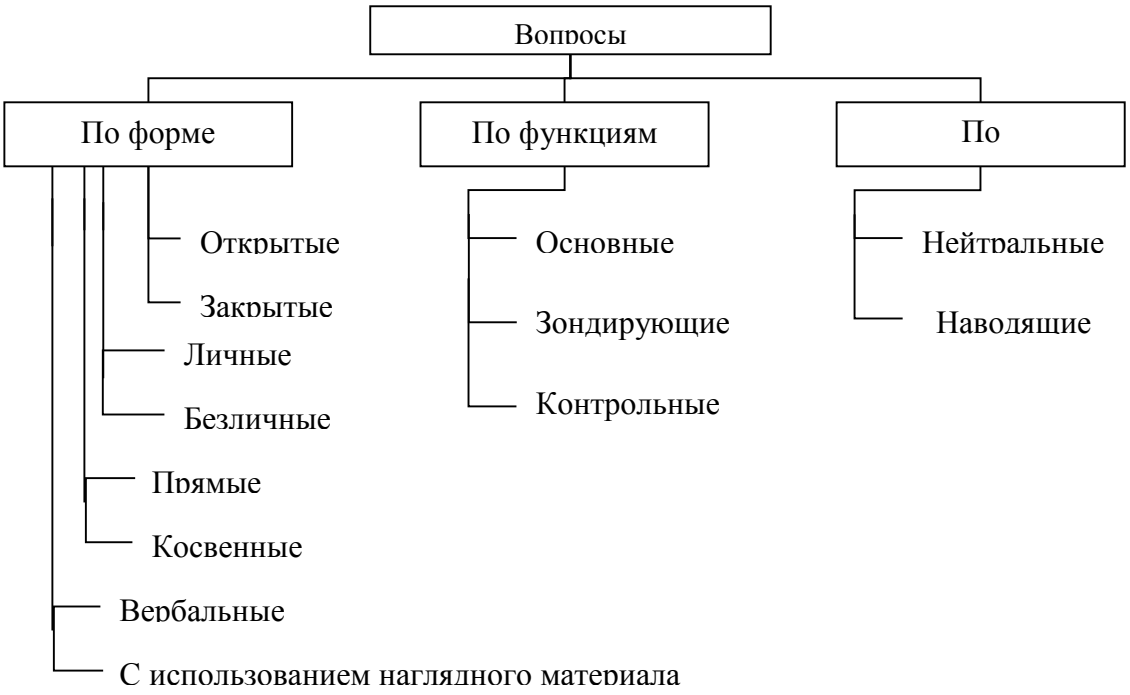


Рисунок 21- Классификация вопросов при интервьюировании

Открытый вопрос называет тему или предмет, оставляя эксперту полную свободу в отношении формы и содержания ответа. Закрытый вопрос предлагает эксперту выбрать ответ из предложенного набора. Личный вопрос непосредственно касается личного опыта эксперта. Безличный вопрос направлен на выявление наиболее распространенных закономерностей предметной области. Прямой вопрос непосредственно указывает на интересующий предмет или тему (используется при «зажатости» эксперта). Косвенный вопрос исподволь затрагивает рассматриваемую проблему. Вербальный вопрос - традиционный устный вопрос. Вопрос использованием наглядного материала позволяет разнообразить интервью и снять усталость эксперта (используются фотографии рисунки, карточки). Основной вопрос направлен на выявление знаний. Зондирующий вопрос направляет рассуждения эксперта в нужную сторону. Контрольный вопрос проверяет достоверность и объективность информации, полученной в интервью ранее. Нейтральный вопрос подчеркивает беспристрастность инженера по знаниям к предмету исследования. Наводящий вопрос ориентирует эксперта принять во внимание позицию инженера по знаниям. Дополнительно в интервью рекомендуется включать следующие вопросы: контактные (снимающие психологический барьер между аналитиком и экспертом), буферные (разграничивающие отдельные темы интервью), оживляющие память экспертов (реконструирующие отдельные случаи из практики), «провоцирующие» (способствующие получению неподготовленных ответов). При использовании метода интервьюирования следует иметь в виду, что его эффективность во многом определяется языком вопросов (понятностью, лаконичностью, терминологией); порядком вопросов (логическая последовательность); уместностью вопросов (этичностью и вежливостью). Прежде чем готовить вопросы, аналитик должен овладеть ключевым набором знаний исследуемой предметной области, поскольку любой вопрос имеет смысл только в контексте.

Метод свободного диалога позволяет извлекать знания в форме беседы с экспертом, поэтому здесь не предусматривается использование жесткого вопросника или плана. В то же время подготовка к свободному диалогу должна проводиться по специальной методике, в которую входит общая, специальная, конкретная и психологическая подготовка. Общая подготовка направлена на повышение научной эрудиции, овладение общей культурой, знакомство с системной методологией. Специальная подготовка сводится к овладению теорией и навыками интервьюирования. Конкретная подготовка предполагает изучение предметной области, подготовку ситуации общения, знакомство с экспертом, тестирование эксперта. Психологическая подготовка включает знакомство с теорией общения и с когнитивной психологией.

Игры с экспертом существенно отличаются от приведений выше индивидуальных активных методов извлечения знании рассматриваются в классе групповых активных методов, где особое место принадлежит ролевым и экспертным методам. Активные групповые методы включают «мозговой

штурм», дискуссии за круглым столом и ролевые игры. Групповые методы позволяют творчески интегрировать знания множества экспертов. Метод «мозгового штурма» - один из наиболее известных и широко применяемых методов генерирования новых идей путем творческого сотрудничества группы специалистов. Являясь в некотором смысле единым мозгом, группа пытается штурмом преодолеть трудности, мешающие разрешить рассматриваемую проблему. В процессе такого штурма участники выдвигают и развивают собственные идеи, стимулируя появление новых и комбинируя их. Для обеспечения максимального эффекта «мозговой штурм» должен подчиняться определенным правилам и основываться на строгом разделении во времени процесса выдвижения идей и процесса их обсуждения и оценки. На первой стадии штурма запрещается осуждать выдвинутые идеи и предложения (считается, что критические замечания уводят к частностям, прерывают творческий процесс, мешают выдвижению идей). Роль аналитика состоит в том, чтобы активизировать творческое мышление участников заседания и обеспечить выдвижение возможно большего числа идей. После выдвижения идей выполняются тщательное их обсуждение, оценка и отбор лучших. На стадии обсуждения участники «мозгового штурма» должны сконцентрироваться на положительных сторонах идей, найти в них рациональные зерна и предложить направления их развития. Выдвигаемые в процессе обсуждения дополнительные идеи могут базироваться на идеях других участников или, наоборот, служить для них фундаментом, катализатором. Значительный эффект дает комбинирование идей при одновременном выявлении преимуществ и недостатков синтезируемых при этом вариантов. Метод «мозгового штурма» эффективен при решении не слишком сложных задач общего организационного характера, когда проблема хорошо знакома всем участникам заседания и по рассматриваемому вопросу имеется достаточная информация. Индивидуальный «мозговой штурм» проводится по тем же правилам, что и коллективный, но выполняется одним экспертом, который одновременно генерирует идеи, дает им объективную оценку и критикует их. Массовый «мозговой штурм» проводится в массовой аудитории (до нескольких десятков человек). Отбор идей проводится на промежуточных этапах. Эксперты группируются по 6 - 8 человек, при этом важно, чтобы непосредственное отношение к задаче имел лишь руководитель группы, а остальные были лишь знакомы с нею (иначе амбиции могут сыграть негативную роль). Штурм проводится в два этапа. На первом этапе оперативные группы осуществляют прямой коллективный «мозговой штурм» При этом желательно, чтобы каждая группа работала над задачей, наиболее близкой по тематике к профилю вошедших в нее специалистов. На втором этапе руководители каждой группы в течение нескольких минут оценивают выдвинутые идеи, отбирают из них наиболее интересные и сообщают их на «пленарном заседании». Двойной «мозговой штурм» органически соединяет в себе процессы генерирования идей и их доброжелательной позитивной

критики. Обратный «мозговой штурм» отличается от прямого тем, что в нем больше внимания уделяется критике высказанных идей.

Метод дискуссии за круглым столом предполагает равноправное обсуждение экспертами поставленной проблемы. Отличительной особенностью метода дискуссии является коллективное рассмотрение предметной области с разных точек зрения и исследование спорных гипотез.

Экспертные игры предназначены для извлечения знаний и базируются на деловых, диагностических и компьютерных играх. По числу участников игры подразделяют на индивидуальные (игры с экспертом) и групповые (ролевые игры в группе). По применению специального оборудования - игры с тренажерами и игры без реквизита. Особый класс представляют собой компьютерные игры. В играх с экспертом инженер по знаниям берет на себя чью-нибудь роль в моделируемой ситуации. Ролевые игры в группе предусматривают участие в игре нескольких специалистов. Участники игры наделяются определенными ролями, а собственно игра проводится по составленному инженером по знаниям сценарию. Компьютерные экспертные игры в настоящее время используются в основном в целях обучения. Они полезны для «разминки» экспертов перед сеансом извлечения знаний.

Текстологические методы включают методы извлечения знаний основанные на изучении текстов учебников, специальной литературы и документов. Простейший алгоритм извлечения знаний из текстов включает следующие шаги: составление основного списка литературы для ознакомления с предметной областью; выбор текста для извлечения знаний; беглое знакомство с текстом, проведение консультаций со специалистами для определения значений незнакомых слов; формирование первой гипотезы о макроструктуре текста; определение смысла прочитанного текста с выпиской ключевых слов; определение связи между ключевыми словами, составление реферата; формирование нового представления знаний на основании макроструктуры текста.

7 Нечеткие знания и способы их обработки

Работа с нечеткостью. При разработке ИИС существует проблема, затрудняющая использование традиционного математического аппарата. Это проблема описания понятий, оперирующих качественными характеристиками объектов. Эти характеристики обычно размыты и не могут быть однозначно интерпретированы, однако содержат важную информацию. Для учета этой информации в ИИС используются методы представления нечетких знаний и механизмы вывода, работающие в их среде. Компонентами нечеткости знаний являются: недетерминированность выводов, многозначность, ненадежность, неполнота, неточность.

Недетерминированность выводов основывается на фундаментальной идее, получившей наименование поиск в пространстве состояний. Недетерминированность означает, что заранее путь решения конкретной

задачи в пространстве ее состояний определить невозможно. Поэтому в большинстве случаев методом проб и ошибок выбирается некоторая цепочка логических заключений, согласующихся с имеющимися знаниями, и в случае если она не приводит к успеху, то организуется перебор с возвратом для поиска другой цепочки и т.д. Такой подход предполагает определение некоторого первоначального пути. Для решения подобных задач рассмотрим классический алгоритм A^*.

В алгоритме A^* используются оценочные функции, построенные на основе априорных оценок стоимости пути до целевого состояния. Для поиска в пространстве состояний используются дерево поиска и методы горизонтального (в ширину) и вертикального (в глубину) поиска на этом дереве. Основные шаги и понятия алгоритма рассмотрим на примере игры в «8», являющейся усеченной версией игры в «15». Целью игры является переход из некоторого начального состояния в конечное состояние, как показано на рис. 22.

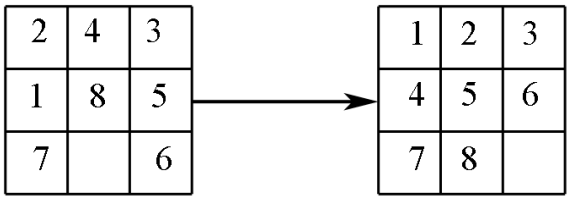


Рисунок 22 - Переход из начального в конечное состояние при игре в «8»

В этой игре в качестве основного объекта удобнее рассматривать не передвигаемые шашки, а перемещение пустого квадрата. При этом можно определить четыре основных оператора, выполняемых над пустым квадратом:

- перемещение пустого квадрата влево;
- перемещение пустого квадрата вверх;
- перемещение пустого квадрата вниз;
- перемещение пустого квадрата вправо.

Оценочная функция f(n) будет формироваться как стоимость оптимального пути к цели из начального состояния через n вершин дерева поиска. Дерево поиска для данного примера показано на рис. 23. Значение оценочной функции в n -й вершине можно представить, как сумму двух составляющих f(n) = g(n) + h(n), где g(n) - стоимость оптимального пути от первой вершины до n -й; h(n) - стоимость оптимального пути от n -й вершины до цели.

Для простоты будем считать, что стоимость перемещения пустого квадрата равна 1. Оптимальным будет путь, имеющий минимальную стоимость. Точное значение f(n) в процессе поиска неизвестно, поэтому введем априорную оценку значения функции: $f^*(n) = g^*(n) + h^*(n)$, где $g^*(n)$

- глубина пройденного пути на дереве поиска от первой до n -ой вершины; $h^*(n)$ - априорное значение h(n).

Основная проблема заключается в определении второй компоненты $h^*(n)$, так как этот путь еще не пройден. В качестве априорной оценки $h^*(n)$ можно, например, взять число шашек, находящихся не на своих местах на n - м шаге поиска. Сформировав, таким образом, оценочную функцию, определим стратегию выбора вершин (применения операторов), в которых значения функции минимальны. Результат поиска показан на рис. 23, где цифры в кружках показывают последовательность переходов из начального состояния в конечное состояние.

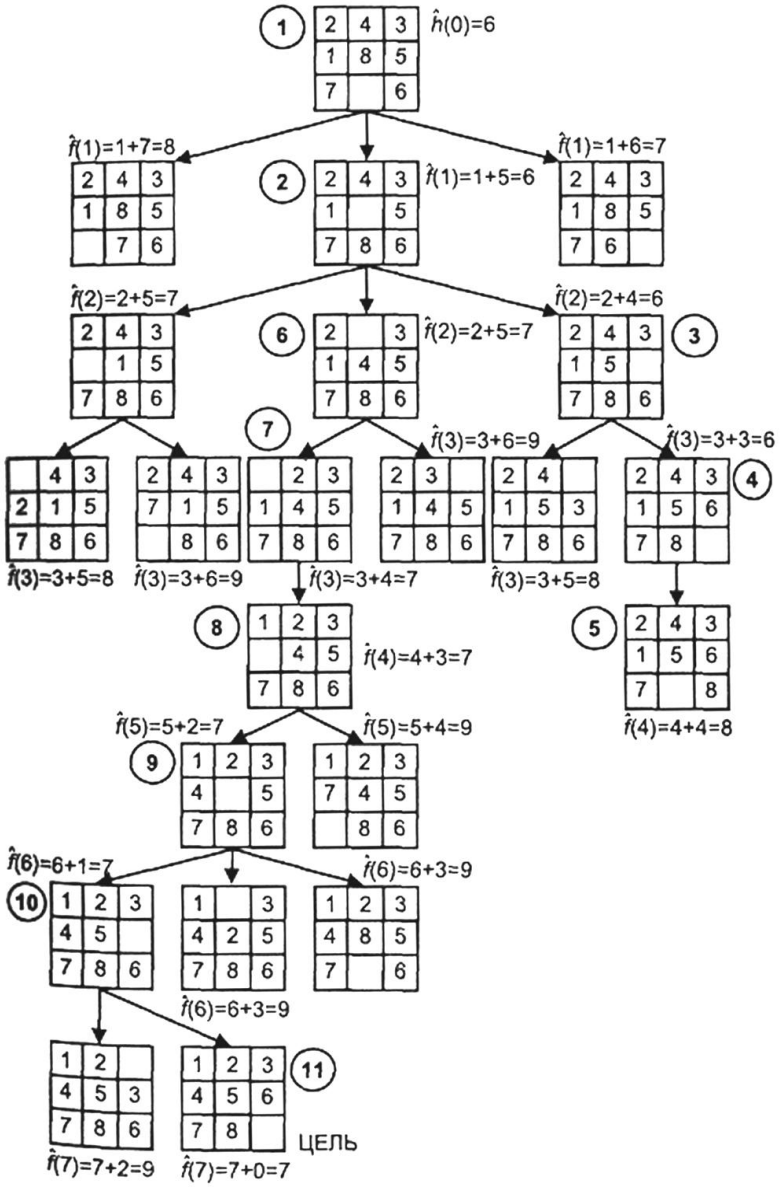


Рисунок 23 - Дерево поиска для игры «8»

Основные шаги алгоритма:

1. Определяются все возможные операторы над пустым квадратом в начальном состоянии и выбирается вариант с наименьшим значением $h^*(n)$;

2. Применяется выбранный оператор и при его использовании получается новое состояние;

3. Создаются вершины следующего уровня иерархии, исходя из анализа всех возможных операторов для перехода в новое состояние;

4. Выбирается состояние с наименьшим значением $h^*(n)$;

5. Перечисленные действия повторяются до тех пор, пока не будет достигнута цель.

При разработке алгоритма на игре в «8» важно, чтобы $h^*(n) \le h(n)$. Если априорная оценка стоимости оптимального пути не превышает истинной стоимости, то нахождение оптимального пути гарантировано. Это условие можно интерпретировать следующим образом: цель поиска не будет достигнута, пока число шашек, находящихся не на своих местах, больше числа перемещений. Если $h^*(n)$ выбрать по-другому, например $h^*(n) = 0$, то будет осуществляться горизонтальный поиск на дереве состояний задачи, при котором раскрываются все вершины нижеследующего уровня.

В области некорректных задач точные знания о проблеме получить невозможно, поэтому приходиться сталкиваться с неточными знаниями, которые не могут быть интерпретированы как полностью истинные или ложные. Для оценки их достоверности также нельзя применить двухбалльную шкалу: логические true/false или 0/1. Существуют знания, достоверность которых выражается некоторой промежуточной цифрой, которая может изменяться от 0 до 1. Для учета нечетких знаний при разработке ИИС используется формальный аппарат нечеткой (fuzzy) алгебры и нечеткой логики, предложенный математиком Л. Заде. Одно из главных понятий в нечеткой логике - это понятие лингвистической переменной (ЛП), значение которой определяется набором словесных (вербальных) характеристик некоторого свойства. Например, ЛП «рост» соответствуют следующие характеристики карликовый, низкий, средний, высокий, очень высокий. Значения лингвистической переменной (ЛП) находится через нечеткие множества (НМ), определяемые на некотором базовом наборе значений или базовой числовой шкале, имеющей размерность. Для определения НМ рассмотрим пример: пусть имеется нечеткое множество T всех высоких людей, входящих во множество людей S. Введем для каждого человека степень его принадлежности множеству T. Функцию принадлежности $\mu(h)$, определяющую в какой степени можно считать высоким человека ростом h сантиметров, представим в виде:

$$\mu(h)=\begin{cases}0 & \text{для } h<150\\ \dfrac{h-150}{60} & \text{для } 150\le h\le 210\\ 1 & \text{для } h>210,\end{cases}$$

где h - рост конкретного человека в сантиметрах.

Если рост человека h =163 см, тогда истинность высказывания, что этот человек высок будет $\mu(h) = 0.21$. Использованная в данном случае функция принадлежности $\mu(h)$ тривиальна. При решении большинства реальных задач подобные функции имеют более сложный вид и содержат большое число аргументов. Методы построения функций принадлежности для нечетких множеств довольно разнообразны. В большинстве случаев функция принадлежности определяет субъективную степень уверенности эксперта в том, что данное конкретное значение базовой шкалы соответствует определяемому НМ. Эту функцию не стоит путать с вероятностью, носящей объективный характер и подчиняющейся другим математическим зависимостям.

Экспертные системы, основанные на нечеткой логике. Правила нечеткого вывода в ЭС описываются в терминах теории НМ. Как правило, они имеют вид: «если цена велика и спрос низкий, то оборот мал». Здесь «цена» и «спрос» используются в качестве входных переменных, «оборот» - как выходное значение. Характеристики «велик», «низкий» и «мал» являются функциями принадлежности НМ. Эти функции определяются на множествах значений «цены», «спроса» и «оборота» соответственно. Нечеткие правила вывода образуют базу правил. В нечеткой экспертной системе все правила работают одновременно, однако степень их влияния на выход может быть различной. Принцип вычисления суперпозиции многих влияний на окончательный результат лежит в основе нечетких экспертных систем. Процесс обработки нечетких правил вывода в экспертной системе состоит из четырех этапов:

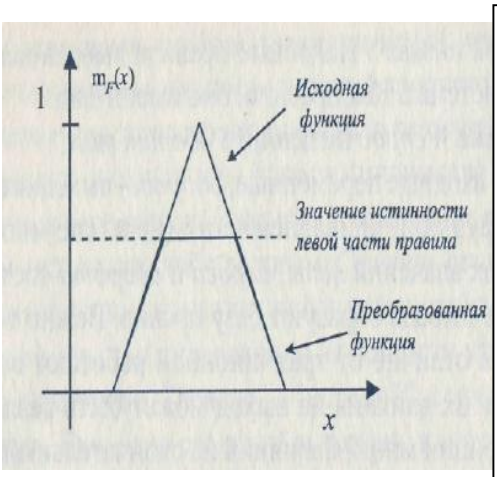
1. Определение степени принадлежности входных значений НМ, указанным в левой части правил вывода;
2. Модификация НМ, указанных в правой части правил вывода в соответствии со значениями истинности, полученными на первом этапе;
3. Объединение (суперпозиция) модифицированных множеств;
4. Скаляризация результата суперпозиции, то есть переход от НМ к скалярным значениям.

Для определения степени истинности левой части каждого правила нечеткая экспертная система вычисляет значения функций принадлежности НМ от соответствующих значений входных переменных. Например, для правила «если цена велика и спрос низкий, то оборот мал» определяется степень вхождения конкретного значения переменной «цена» в нечеткое множество «велика», то есть истинность предиката «цена велика». К

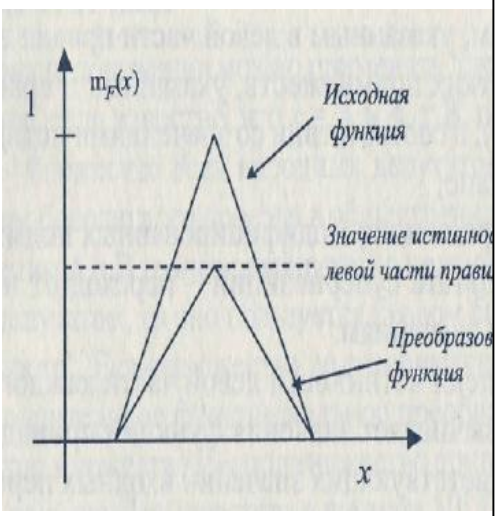
вычисленным значениям истинности могут применяться логические операции. Наиболее часто используются следующие определения операций нечеткой логики:

$$\text{truth}(\textbf{НЕ}\ x) = 1 - \text{truth}(x),$$
$$\text{truth}(x\ \textbf{И}\ y) = \min[\text{truth}(x), \text{truth}(y)],$$
$$\text{truth}(x\ \textbf{ИЛИ}\ y) = \max[\text{truth}(x), \text{truth}(y)],$$

где x и y - высказывания; truth (z) - степень истинности высказывания z.



а)



б)

Полученное значение истинности используется для модификации НМ, указанного в правой части правила. Для выполнения такой модификации используют один из двух методов: «минимума» и «произведения» Метод «минимума» (рис. 24 а) ограничивает функцию принадлежности для множества, указанного в правой части правила, значением истинности левой части. Метод «произведение» (рис. 24 б) использует значение истинности левой части как коэффициент, на который умножаются значения функции принадлежности. Результат выполнения правила - нечеткое множество, то есть происходит ассоциирование переменной и функции принадлежности, указанных в правой части. Выходы всех правил вычисляются нечеткой экспертной системой отдельно, однако в правой части нескольких из них может быть указана одна и та же нечеткая переменная. При определении

Рисунок 24 - Модификация НМ, а – Метод «минимума», б – Метод «произведение»

Процесс обработки нечетких правил вывода поясним на примере.

Правая часть правил:
$$\begin{cases} \text{если цена мала, то спрос велик} \\ \text{если цена велика, то спрос мал} \end{cases}$$

содержит одну и ту же переменную - «спрос». Два нечетких множества, получаемые при выполнении этих правил, должны быть объединены экспертной системой. Традиционно суперпозиция функций принадлежности нечетких множеств $\mu_1(x), \mu_2(x), \ldots, \mu_n(x)$ определяется как:

$$\mu_{sum}(x) = \max[\mu_i(x)] \qquad \forall x,\ i \in [1, n].$$

Графическое представление подобной суперпозиции приведено на рисунке 25.

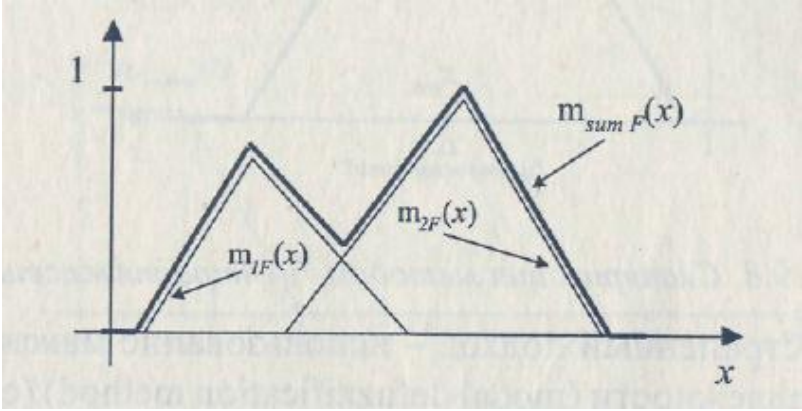


Рисунок 25 - Метод «Max Combination»

Другой метод суперпозиции состоит в суммировании значений всех функций принадлежности. Графическая интерпретация метода приведена на рисунке 26.

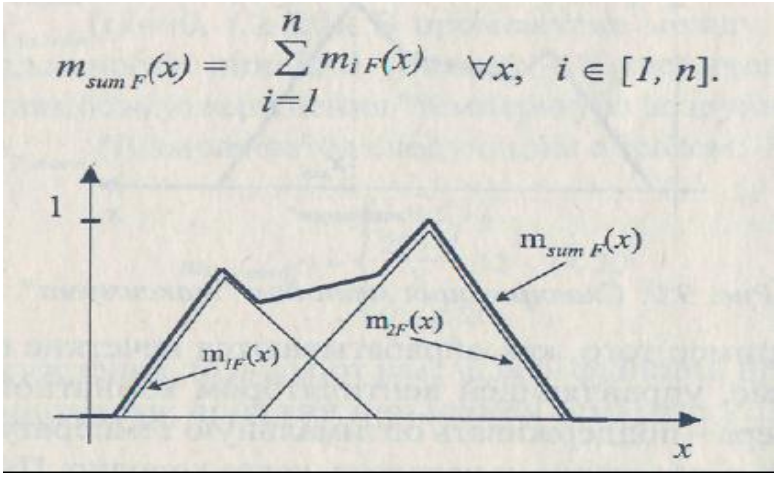


Рисунок 26 - Метод "Sum Combination"

Самым простым, но и наименее часто используемым, является подход, когда суперпозиция не производится. Выбирается одно из правил вывода, результат которого используется в качестве интегрального результата. Конечный этап обработки базы правил вывода - это переход от нечетких значений к конкретным скалярным. Процесс преобразования нечеткого множества в единственное значение называется

скаляризацией. В качестве такого значения часто используется «центр тяжести» функции принадлежности нечеткого множества (рис. 27).

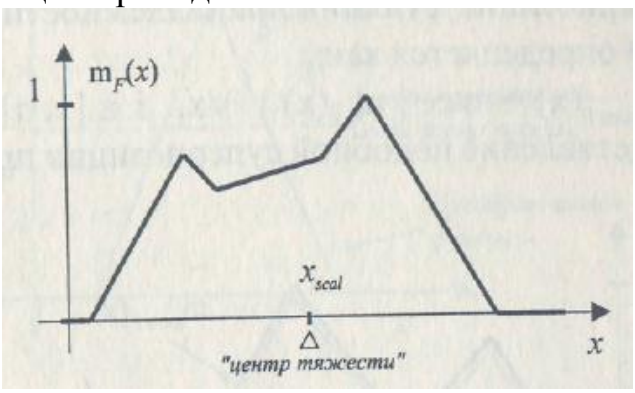


Рисунок 27 - Скаляризация методом «центра тяжести»

Другой распространенный подход - использование максимального значения функции принадлежности (рис. 28). Конкретный выбор методов суперпозиции и скаляризации осуществляется в зависимости от желаемого поведения нечеткой экспертной системы.

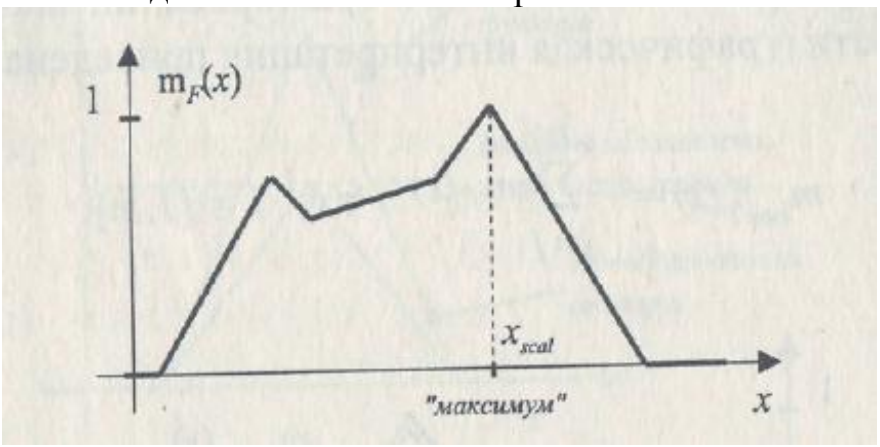


Рисунок 28 - Скаляризация методом «максимума»

Рассмотрим пример того, как обрабатываются нечеткие правила вывода в экспертной системе, управляющей вентилятором комнатного кондиционера. Задача кондиционера - поддерживать оптимальную температуру воздуха в комнате, охлаждая его, когда жарко, и нагревая, когда холодно. Алгоритм работы кондиционера может быть задан следующими правилами:

1. Если температура воздуха в комнате высокая, то скорость вращения вентилятора высокая;
2. Если температура воздуха в комнате средняя, то скорость вращения вентилятора средняя;
3. Если температура воздуха в комнате низкая, то скорость вращения вентилятора низкая.

Для того чтобы система могла обрабатывать эти правила, надо задать функции принадлежности для нечетких множеств, определенных на значениях температуры (t) и скорости вращения вентилятора (v). Пусть

температура воздуха в комнате находится в пределах от 0 °С до 60 °С. В противном случае кондиционер вряд ли поможет, функцию принадлежности для нечеткого множества «низкая», определенную на интервале изменения температуры, можно задать как показано на рис. 29. Если температура меньше 12 °С, то это определенно низкая темпера-тура для комнаты $\mu_1(t)$ = 1 при t < 12 °С. Температуру выше 20 °С никак нельзя назвать низкой, поэтому $\mu_1(t) = 0$ при t > 20 °С. В промежутке между этими значениями функция принадлежности линейно убывает, то есть с увеличением температуры уменьшается истинность утверждения «температура воздуха в комнате низкая».

$$\mu_1(t) = \begin{cases} 1 & \text{при } t \le 12 \\ \dfrac{20 - t}{8} & \text{при } 12 < t < 20. \\ 0 & \text{при } t \ge 20 \end{cases}$$

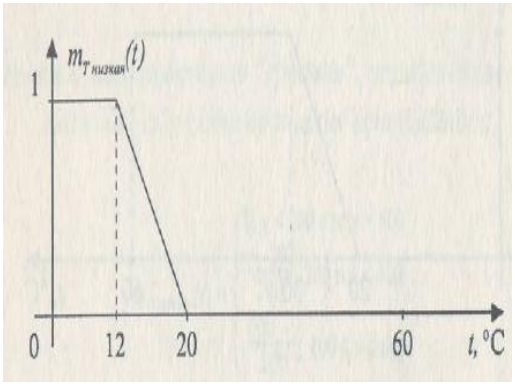


Рисунок 29 - Нечеткое подмножество «низкая», определенное на множестве значений температуры

Сходные рассуждения позволяют задать функции принадлежности для оставшихся множеств: «средняя» (рис.30) и «высокая» (рис. 31).

$$\mu_2(t) = \begin{cases} 0 \quad \text{при } t < 12 \text{ или } t > 30 \\ \dfrac{t - 12}{8} \quad \text{при } 12 \le t < 20 \\ \dfrac{30 - t}{10} \quad \text{при } 20 \le t \le 30 \end{cases} .$$

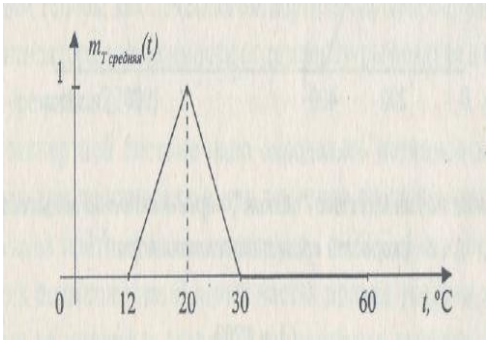


Рисунок 30 - Нечеткое подмножество «средняя», определенное на множестве значений температуры

$$\mu_3(t)=\begin{cases}0 & \text{при } t<20\\ \dfrac{t-20}{10} & \text{при } 20\le t<30.\\ 1 & \text{при } t\ge 30\end{cases}$$

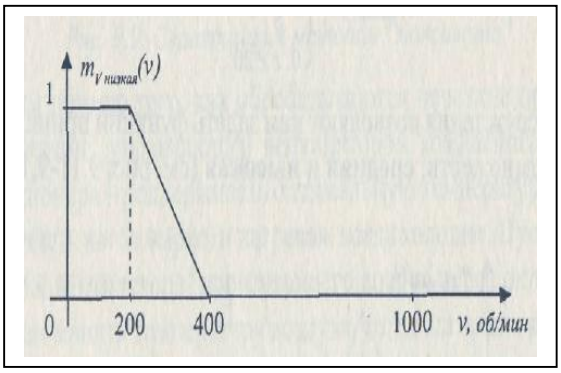


Рисунок 31 - Нечеткое подмножество «высокая», определенное на множества значений температуры

Определим нечеткие подмножества для скорости вращения вентилятора. Пусть скорость может изменяться от 0 до 1000 об/мин. Допустимым будет следующий вариант определения функций принадлежности для нечетких множеств скорости «низкая», «средняя» и «высокая» (рис. 32 - 34).

$$\mu_1(v)=\begin{cases}1 & \text{при } v<200\\ \dfrac{400-v}{200} & \text{при } 200\le v\le 400.\\ 0 & \text{при } v>400\end{cases}$$

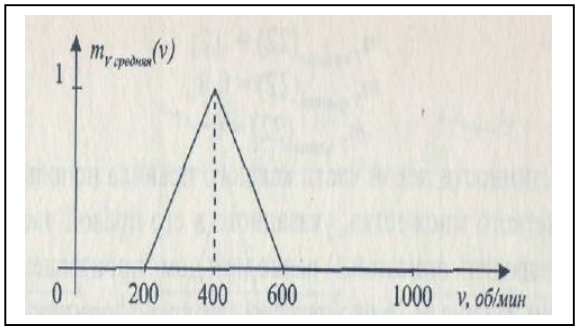


Рисунок 32 - Нечеткое множество «низкая», определенное на множестве значении скорости вращения вентилятора

$$\mu_3(v) = \begin{cases} 0 & \text{при } v < 400 \\ \dfrac{v - 400}{200} & \text{при } 400 \le v < 600 \\ \dfrac{600 - v}{200} & \text{при } v \ge 600 \end{cases}$$

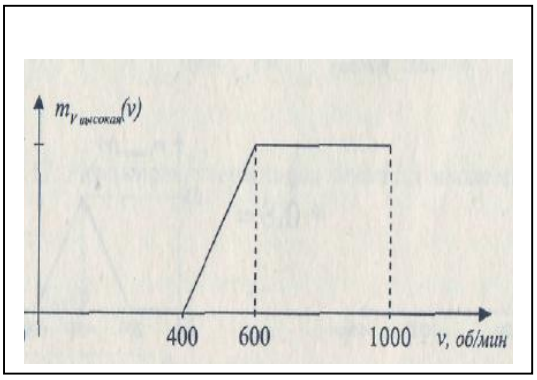


Рисунок 33 - Нечеткое множество «высокая», определенное на множестве значений скорости вращения вентилятора

Рассмотрим теперь, как нечеткая экспертная система определяет скорость вращения вентилятора в зависимости от температуры воздуха в комнате. Пусть эта температура равна 22 °С. Сначала экспертной системе надо определить истинность левых частей правил вывода при подстановке в них текущего значения температуры. Для этого она должна найти степень вхождения t = 22 °С в каждое из указанных слева нечетких множеств. В левых частях правил указаны три множества, заданных на интервале значений температуры: «высокая», «низкая» и «средняя». Степень вхождения находим, вычисляя значение функций принадлежности каждого множеств от

t = 22 °C: $\mu_1(22) = 0.2;\ \mu_2(22) = 0.8;\ \mu_3(22) = 0$. Значения истинности левой части каждого правила используются для модификации нечеткого множества, указанного в его правой части. Модификацию будем производить методом «произведения» На рис. 34 изображено, как трансформируются находящиеся в правых частях правил нечеткие подмножества «высокая», «средняя» и «низкая». Нечеткой экспертной системе необходимо обобщить результаты действия всех правил вывода, то есть произвести суперпозицию полученных нечетких множеств. Воспользуемся методом «Max Combination» (рис. 25). Результат объединения нечетких множеств показан на рис. 36. Осуществим переход от суперпозиции множеств к скалярному значению. Скаляризацию произведем методом «центра тяжести». Иллюстрация того, как получается результат, представлена на рис. 36. Центр тяжести фигуры на рис. 36 находится в точке v = 560.5691. Следовательно, экспертная система при температуре воздуха в комнате равной 22 °C определит до целой части значение скорости вращения вентилятора v = 561 об/мин. При других значениях температуры функция принадлежности обобщенного результата выполнения всех правил, изображенная на рис. 37, будет меняться. Если на вход экспертной системы поступит значение t = 28 °C (рис. 38), то центр тяжести в этом случае будет находится в точке v = 746.6667, что составляет скорость вентилятора 747 об/мин.

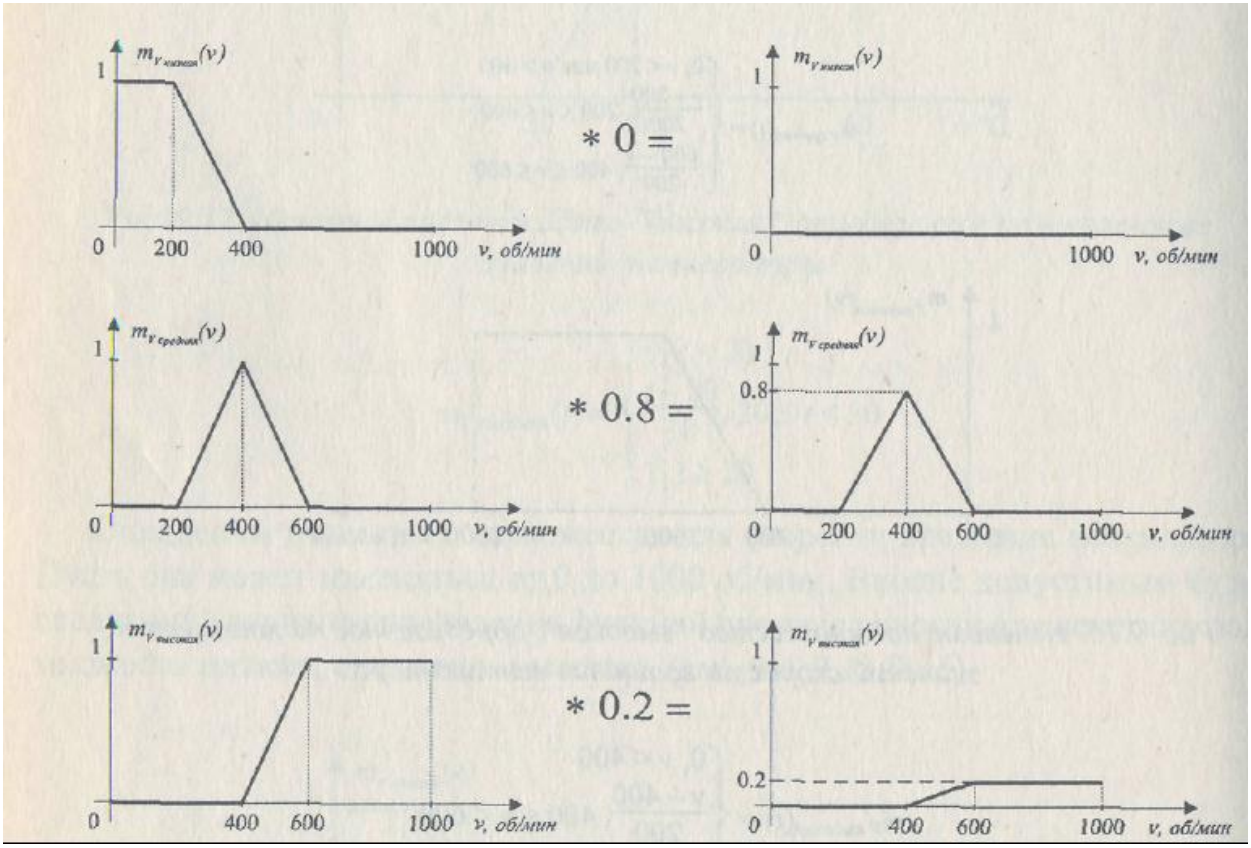


Рисунок 34 - Модификация нечетких подмножеств, определенных на интервале изменения скорости вращения вентилятора

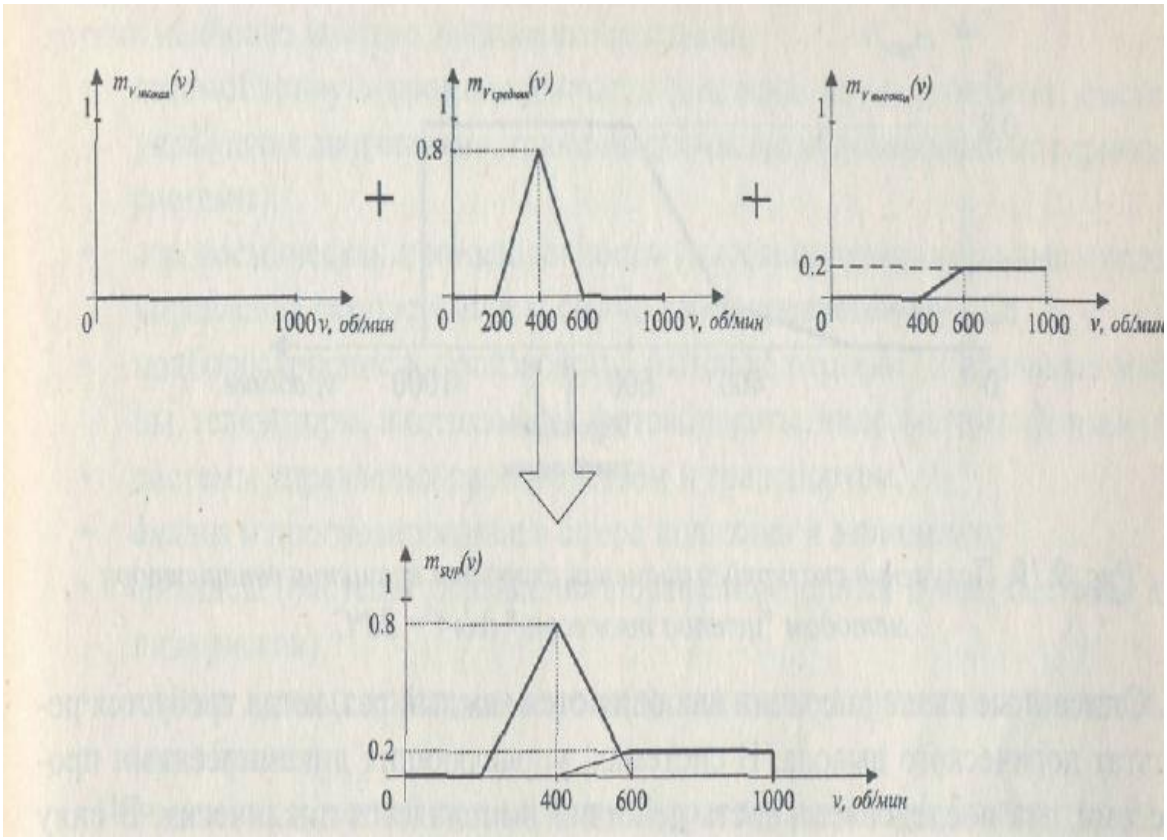


Рисунок 35 - Результат суперпозиции нечетких множеств

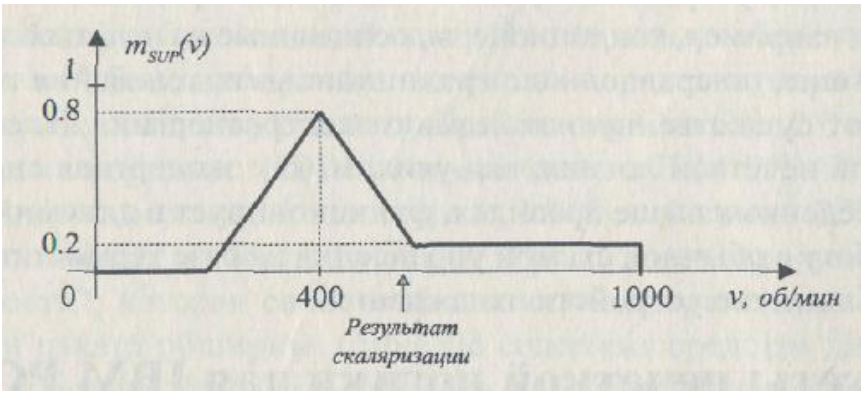


Рисунок 36 - Получение скалярного значения скорости вращения вентилятора методом «центра тяжести» для t = 22 °C

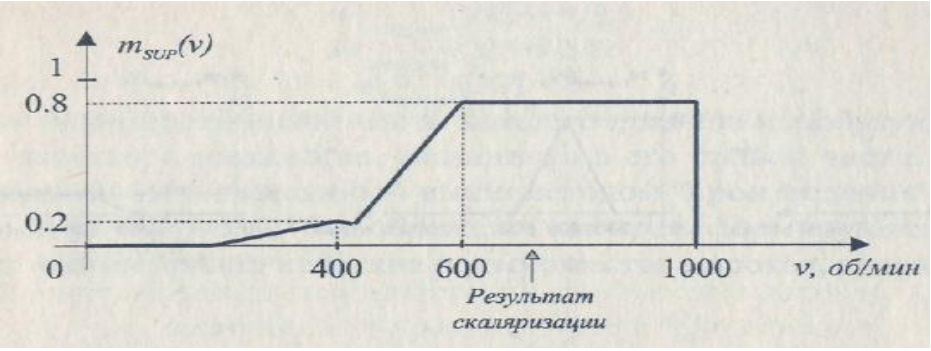


Рисунок 37 - Получение скалярного значения скорости вращения вентилятора методом «центра тяжести» для температуры t = 28 °C

Практика показывает, что даже для управления таким простым устройством как вентилятор применение нечеткой экспертной системы оказывается экономически выгодным. Так, например, кондиционеры, основанные на нечеткой логике, обеспечивают меньшие, по сравнению с традиционными системами, колебания темпера-туры и дают существенную экономию электроэнергии.

Описанные выше операции выполняются каждый раз, когда требуется результат логического вывода. В системах, управляющих динамическими процессами, эта последовательность действий выполняется циклически. Вследствие раздельного вычисления результатов логического вывода, нечеткие экспертные системы эффективно реализуются в параллельных алгоритмах обработки информации и управления.

8 Основные понятия теории искусственных нейронных сетей

Модель искусственного нейрона. Искусственная нейронная сеть (ИНС) - это упрощенная модель биологического мозга, точнее нервной ткани. Нервная система человека, построенная из элементов, называемых нейронами, имеет поразительную сложность. Около 10^{11} нейронов участвуют в примерно 10^{15} передающих связях, имеющих длину метр и более. Каждый нейрон обладает многими качествами, общими с другими элементами тела, но его уникальной способностью является прием, обработка и передача электрохимических сигналов по нервным путям, которые образуют коммуникационную систему мозга. Структура пары типичных биологических нейронов показана на рис. 38. Нейрон состоит из тела (сомы), содержащего ядро, и отростков - дендритов, по которым в нейрон поступают входные сигналы. Один из отростков, ветвящийся на конце, служит для передачи выходных сигналов данного нейрона другим нервным клеткам. Он называется аксоном. Соединение аксона с дендритом другого нейрона называется синапсом. Нейрон возбуждается и передает сигнал через аксон, если число пришедших по дендритам возбуждающих сигналов больше, чем число тормозящих. Несмотря на то, что у этой функциональной схемы нейрона

много усложнений и исключений, большинство искусственных нейронных сетей моделируют лишь эти простые свойства.

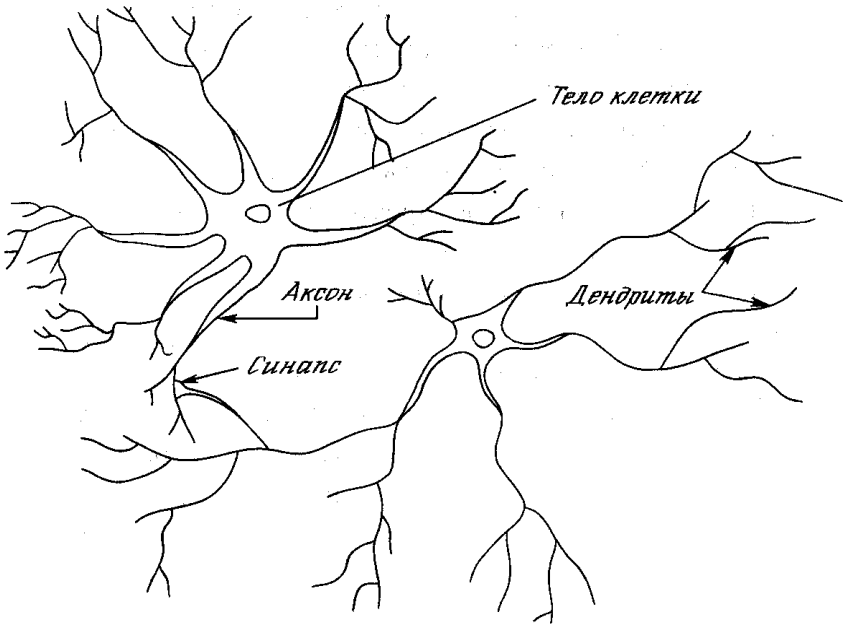


Рисунок 38 - Биологический нейрон

Искусственный нейрон (рис. 39), далее просто нейрон, задается совокупностью своих входов, обозначенных $x_1, x_2, \ldots, x_n$, весами входов $w_1, w_2, \ldots, w_n$, функцией состояния NET и функцией активации F. Функция состояния определяет состояние нейрона в зависимости от значений его входов, весов входов и, возможно, предыдущих состояний. Наиболее часто используются функции состояния, вычисляемые как сумма произведений значений входов на веса соответствующих входов по всем входам $\mathrm{NET} = \sum_{i=1}^{n} x_i w_i$. Суммирующий блок, соответствующий телу биологического нейрона, складывает взвешенные входы алгебраически, создавая выход, который будем называть NET.

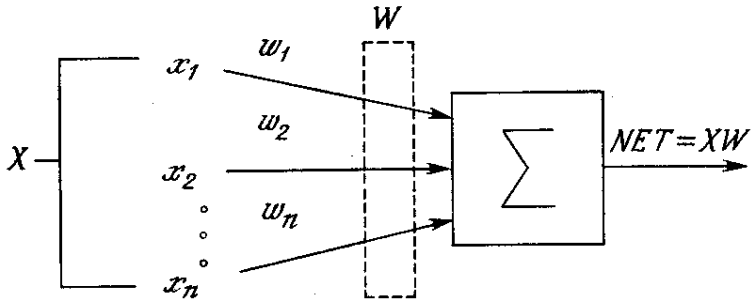


Рисунок 39 - Искусственный нейрон: X - вектор входных NET = XW – вектор состояния нейрона

Сигнал NET преобразуется функцией активации F и дает выходной сигнал нейрона $OUT = F(NET)$.

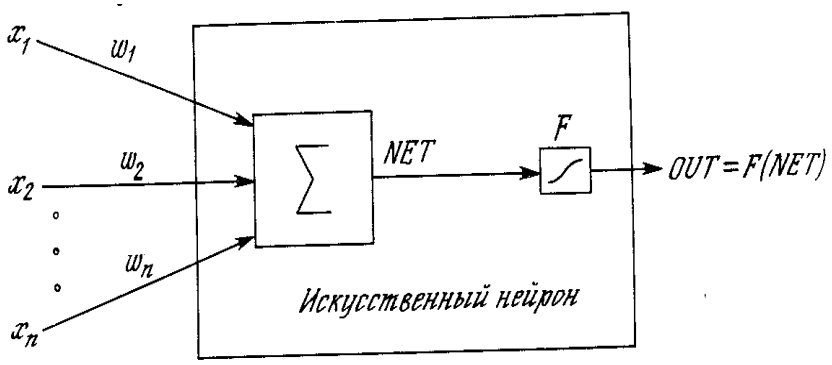


Рисунок 40 - Искусственный нейрон с активационной функцией F

На рис. 40 блок, обозначенный F, принимает сигнал NET и выдает сигнал OUT. Если блок F сужает диапазон изменения величины NET так, что при любых значениях NET значения OUT принадлежат некоторому конечному интервалу, то F называется «сжимающей» функцией. В качестве «сжимающей» функции часто используется сигмоидальная (S-образная) функция, показанная на рис. 41.

Рисунок 41 - Сигмоидальная функция

Эта функция имеет вид: $OUT = 1/(1+e^{-NET})$. По аналогии с электронными системами активационную функцию можно считать нелинейной усилительной характеристикой искусственного нейрона. Коэффициент усиления вычисляется как отношение приращения величины OUT к вызвавшему его небольшому приращению величины NET. Центральная область сигмоидальной функции, имеющая большой коэффициент усиления, решает проблему обработки слабых сигналов, в то время как области с падающим усилением на положительном и отрицательном концах подходят для больших возбуждений. Таким образом, нейрон функционирует с большим усилением в широком диапазоне уровня входного сигнала. Другой широко используемой функцией активации является

гиперболический тангенс OUT = th(x). Подобно сигмоидальной функции гиперболический тангенс является S-образной функцией, но он симметричен относительно начала координат, и в точке NET = 0 значение выходного сигнала OUT равно нулю (рис. 42). В отличие от сигмоидальной функции гиперболический тангенс принимает значения различных знаков, что оказывается выгодным при поиске экстремума в пространстве параметров ИНС.

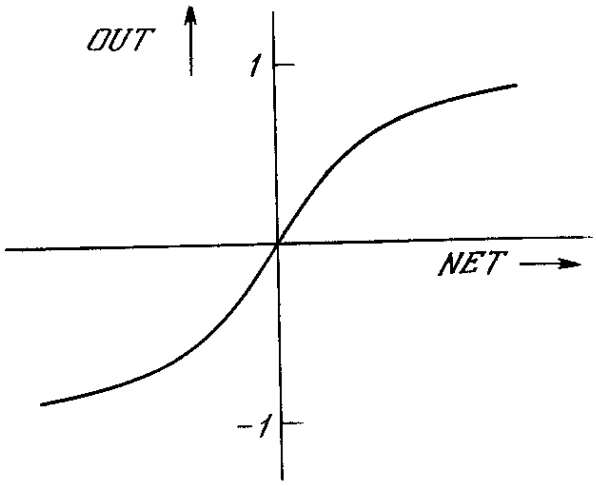


Рисунок 42 - Функция гиперболического тангенса

Простейшими формами функции активации являются линейная и ступенчатая (пороговая) функции. Линейная функция активации дифференцируема и легко вычисляется, что в ряде случаев позволяет уменьшить ошибки выходных сигналов в ИНС. Однако она не универсальна и не обеспечивает решение многих плохо формализованных задач. Ступенчатая функция обычно задается в виде единичной функции с жесткими ограничениями: она равна нулю, если $NET < 0$, и единице, если $NET \geq 0$. Такая функция не обеспечивает достаточной гибкости ИНС при обучении. Рассмотренная простая модель искусственного нейрона игнорирует многие свойства своего биологического двойника. Например, она не принимает во внимание задержки во времени, которые воздействуют на динамику системы. Несмотря на эти ограничения, сети, построенные из этих нейронов, обнаруживают свойства, сильно напоминающие биологическую систему.

Модели нейронных сетей. Нейронная сеть представляет собой совокупность искусственных нейронов, организованных слоями. Реальная нейронная сеть может содержать один или большее количество слоев и соответственно характеризоваться как однослойная или как многослойная, с обратными связями и без них. Многослойные сети отличаются от однослойных тем, что между входными и выходными данными располагаются несколько так называемых скрытых слоев нейронов, добавляющих больше нелинейных связей в модель. Простейшая сеть состоит из группы нейронов, образующих один слой (рис. 43). На рисунке вершины -

круги слева служат лишь для распределения входных сигналов. Они не выполняют каких - либо вычислений, и поэтому не будут считаться слоем. По этой причине они обозначены кругами, чтобы отличать их от вычисляющих нейронов, обозначенных квадратами. Каждый элемент из множества входов X отдельным весом соединен с каждым искусственным нейроном, и каждый нейрон выдает взвешенную сумму входов в сеть. В искусственных и биологических сетях многие соединения могут отсутствовать, поэтому на рисунке все соединения показаны в целях общности. Многослойные сети обладают большими вычислительными возможностями, чем однослойные. Послойная организация нейронов копирует слоистые структуры определенных отделов мозга. Многослойные сети могут образовываться каскадами слоев: выход одного слоя является входом для последующего слоя. Двухслойная сеть со всеми соединениями показана на рис. 44. Если функция активации является линейной, то нейронная сеть также будет линейной. Двухслойная линейная сеть эквивалентна одному слою с весовой матрицей, равной произведению двух весовых матриц, то есть $\mathrm{NET} = (\mathrm{XW}_1)\mathrm{W}_2 = \mathrm{X}(\mathrm{W}_1\mathrm{W}_2)$. Следовательно, любая многослойная линейная сеть может быть заменена эквивалентной однослойной сетью. Для расширения вычислительных возможностей ИНС применяется нелинейная функция активации.

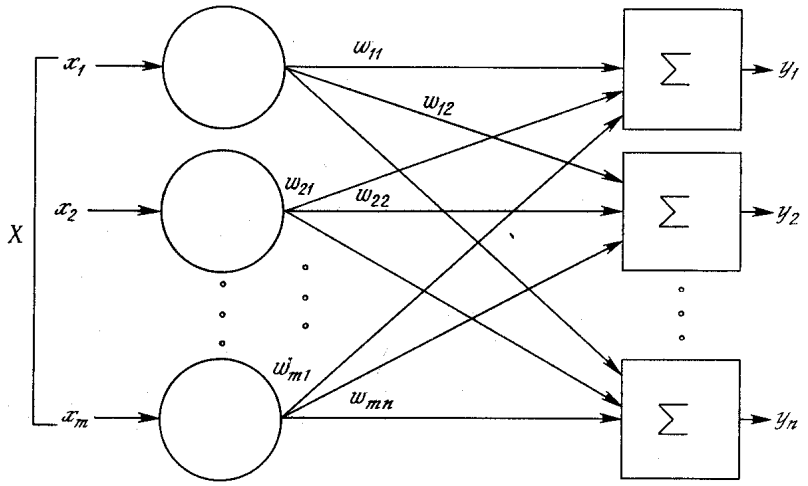


Рисунок 43 - Однослойная нейронная сеть

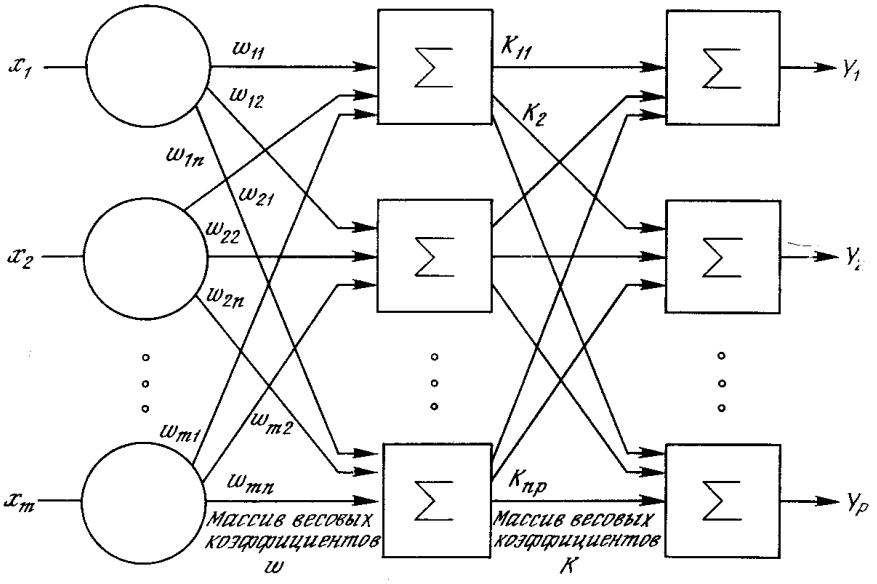


Рисунок 44- Двухслойная нейронная сеть

По архитектуре связей нейронные сети могут быть сгруппированы в два класса: сети прямого распространения, в которых обратные связи отсутствуют (нет соединений, идущих от выходов некоторого слоя к входам этого же слоя или предшествующих слоев) и сети рекуррентного типа, в которых возможны обратные связи. В сетях прямого распространения нет памяти, их выход полностью определяется текущими входами и значениями весов. Сети прямого распространения подразделяются на однослойные перцепротроны (сети) и многослойные перцептроны (сети). Название перцептрон для нейросетей введено Ф. Розенблаттом, разработчиком первой нейросети (1957 г.). Он же доказал сходимость области решений для перцептрона при его обучении. Реккурентные сети могут обладать свойствами, сходными с кратковременной человеческой памятью.

Класс рекуррентных нейронных сетей гораздо обширнее и сложнее по своему устройству. Поведение рекуррентных сетей описывается дифференциальными или разностными уравнениями, как правило, первого порядка. Это гораздо расширяет области применения нейросетей и способы их обучения. Сеть организована так, что каждый нейрон получает входную информацию от других нейронов, возможно, и от самого себя, и от окружающей среды. Этот тип сетей имеет важное значение для моделирования нелинейных динамических систем. Среди рекуррентных сетей можно выделить сети Хопфилда и сети Кохонена. При всем многообразии возможных конфигураций ИНС на практике получили распространение лишь некоторые из них. Классическим моделями нейронных сетей являются:

1. Многослойные перцептроны содержат помимо входного и выходного слоев так называемые скрытые слои. Они представляют собой нейроны, которые не имеют непосредственных входов исходных данных, а связаны только с выходами входного слоя и с входом выходного слоя. Таким образом, скрытые слои дополнительно преобразуют информацию и добавляют

нелинейности в модели. Многослойный перцептрон с сигмоидальными функциями активации способен аппроксимировать любую функциональную зависимость (доказано в виде теоремы), однако при этом не известно ни нужное число слоев, ни нужное количество скрытых нейронов, ни необходимое для обучения сети время. Эти проблемы до сих пор не решены;

2. Сети Хопфилда строятся из N нейронов, связанных каждый с каждым кроме самого себя, причем все нейроны являются выходными. Нейронную сеть можно использовать в качестве ассоциативной памяти, а также для обработки неупорядоченные, упорядоченных во времени или пространстве образцов (рукописные буквы, временные ряды, графики);

3. Сети Кохонена еще называют «самоорганизующимися картами признаков». Сеть рассчитана на самостоятельное обучение. В процессе обучения на вход сети подаются различные образцы. Сеть улавливает особенности их структуры и разделяет образцы на кластеры, а затем уже обученная сеть относит каждый вновь поступающий пример к одному из кластеров, руководствуясь некоторым критерием «близости». Сеть состоит из одного входного и одного выходного слоя. Количество элементов в выходном слое непосредственно определяет, сколько различных кластеров сеть сможет распознать. Каждый из выходных элементов получает на вход весь входной вектор. Как и во всякой нейронной сети, каждой связи приписан некоторый синаптический вес. В большинстве случаев каждый выходной элемент соединен также со своими соседями. Эти внутрислойные связи играют важную роль в процессе обучения, так как корректировка весов происходит только в окрестности того элемента, который наилучшим образом откликается на очередной вход. Выходные элементы соревнуются между собой за право вступить в действие «получить урок». Выигрывает тот из них, чей вектор весов окажется ближе всех к входному вектору.

Структура ИНС, включая определение числа слоев и числа нейронов в каждом слое, формируется до начала обучения, поэтому успешное решение этой проблемы во многом зависит от опыта и искусства аналитика, проводящего исследование.

Обучение нейронных сетей. Главное отличие и преимущество нейросетей перед классическими средствами прогнозирования и классификации заключается в их способности к обучению. На этапе обучения происходит вычисление синаптических коэффициентов в процессе решения нейронной сетью задач, в которых нужный ответ определяется не по правилам, а с помощью примеров, сгруппированных в обучающие множества. Нейросеть на этапе обучения сама выполняет роль эксперта в процессе подготовки данных для построения экспертной системы. Предполагается, что правила находятся в структуре обучающих данных. Такие данные представляют собой ряды примеров с указанием для каждого из них значения выходного параметра, которое было бы желательно получить. Действия, которые при этом происходят, можно назвать обучение с учителем. На вход сети подается вектор исходных данных, а на выходной узел сообщается

желаемое значение результата вычислений. Контролируемое обучение нейросети можно рассматривать как решение оптимизационной задачи. Ее целью является минимизация функции ошибок на данном множестве примеров путем выбора значений весов W. Достижение минимума называется сходимостью процесса обучения. Поскольку ошибка зависит от весов нелинейно, получить решение в аналитической форме невозможно, и поиск глобального минимума осуществляется посредством итерационного процесса, так называемого обучающего алгоритма. В настоящее время используются более сотни разных обучающих алгоритмов, отличающихся друг от друга стратегией оптимизации и критерием ошибок. Обычно в качестве меры погрешности берется средняя квадратичная ошибка

$$E = \sqrt{\frac{\sum_{i=1}^{M}(d_i - y_i)^2}{M}},$$

где M – число примеров в обучаемом множестве.

Среди обучающих алгоритмов наиболее распространенным является алгоритм обратного распространения ошибок. Согласно методу перед началом обучения сети всем межнейронным связям присваиваются небольшие случайные значения весов. Каждый шаг обучающей процедуры состоит из двух фаз. Во время первой фазы входные элементы сети устанавливаются в заданное состояние. Входные сигналы распространяются по сети, порождая некоторый выходной вектор. При этом используются сигмоидальные функции активации. Полученный выходной вектор сравнивается с требуемым (правильным) вектором. Если они совпадают, то весовые коэффициенты связей не изменяются. В противном случае вычисляется разница между фактическими и требуемыми выходными значениями, которая передается последовательно от выходного слоя к входному слою. На основе этой информации проводится модификация весов связей в соответствии с формулой:

$$\Delta W_{ij}(t+1) = \mu \Delta W_{ij}(t) - (1-\mu)\varepsilon \frac{\partial E}{\partial W_{ij}},$$

где $\Delta W_{ij}(t)$ - изменение веса связи на шаге итерации t обучающей пары вход-выход i -го нейрона, получающего входной сигнал, и j –го нейрона, посылающего выход-ной сигнал; μ - коэффициент, задаваемый пользователем в интервале (0,1); ε - величина шага сети или мера точности обучения сети. В современных нейросетевых пакетах пользователь может сам определять, как будет изменяться величина шага сети, очень часто по умолчанию берется линейная или экспоненциальная зависимость величины шага от количества итераций сети. Чем меньше шаг сети, тем больше времени потребуется на обучение сети и тем возможнее станет ее попадание в окрестность локального минимума ошибки.

Общая ошибка функционирования сети определяется по формуле:

$$D = \frac{1}{2}\sum_{p}\sum_{j}(T_{jp} - R_{jp})^2,$$

где T_{jp} и R_{jp} - соответственно желательное и действительное значение выходного сигнала j – го блока для p – й обучающей пары нейронов вход-выход.

Когда величина ошибки достигает приемлемо малого уровня, обучение останавливают, и сеть готова к выполнению возложенных на нее задач. Важно отметить, что вся информация, которую сеть приобретает о задаче, содержится в наборе примеров. Поэтому качество обучения сети зависит от количества примеров в обучающей выборке, а также от того, насколько полно эти примеры описывают задачу. Считается, что для полноценной тренировки требуется хотя бы несколько десятков (а лучше сотен) примеров. Если не для всех примеров обучающей выборки известны правильные ответы, то обучение сети проводится без учителя. В этом случае применение самонастраивающихся сетей Кохонена дает возможность определить внутреннюю структуру поступающих в сеть данных и распределить образцы по категориям.

Несмотря на многочисленные успешные применения алгоритма обратного распространения при обучении ИНС, у него есть недостатки. Больше всего неприятностей приносит неопределенно долгий процесс обучения. В сложных задачах для обучения сети могут потребоваться дни или даже недели, она может и вообще не обучиться. Длительное время обучения может быть результатом неоптимального выбора шага сети ε. Неудачи в обучении сети обычно возникают по двум причинам.

1. Паралич сети. В процессе обучения сети значения весов могут в результате коррекции стать очень большими величинами. Это может привести к тому, что все или большинство нейронов будут функционировать при очень больших значениях OUT, в области, где производная сжимающей функции очень мала. Так как посылаемая обратно в процессе обучения ошибка пропорциональна этой производной, то процесс обучения может практически замереть. В теоретическом отношении эта проблема плохо изучена. Обычно этого избегают уменьшением размера шага, но это увеличивает время обучения. Для предохранения от паралича применяются различные эвристики, но пока что они могут рассматриваться лишь как экспериментальные.

2. Попадание в локальный минимум. Обратное распространение использует разновидность градиентного спуска, то есть осуществляет спуск вниз по поверхности ошибки, непрерывно подстраивая веса в направлении минимума. Поверхность ошибки сложной сети сильно изрезана и состоит из холмов, долин, складок и оврагов в пространстве высокой размерности. Сеть может попасть в локальный минимум (неглубокую долину), когда рядом имеется гораздо более глубокий минимум. В точке локального минимума все направления ведут вверх, и сеть не может из него выбраться.

Способы реализации ИНС. Нейронные сети могут быть реализованы программным или аппаратным способами. Вариантом аппаратной реализации являются нейрокомпьютеры. Большинство современных нейрокомпьютеров представляют собой персональный компьютер, снабженный дополнительной нейроплатой. Повышенный интерес вызывают специализированные нейрокомпьютеры, в которых реализованы принципы архитектуры нейронных сетей. В тех случаях, когда разработка или внедрение аппаратных реализаций нейронных сетей обходится слишком дорого, применяют более дешевые программные реализации. Программу моделирования нейронной сети обычно называют программой-имитатором или нейропакетом, понимая под этим программную оболочку, эмулирующую для пользователя среду нейрокомпьютера на обычном компьютере. В настоящее время на рынке программного обеспечения имеется множество самых разнообразных программ для моделирования нейронных сетей. В них воплощены практически все известные алгоритмы обучения и топологии нейросетей. Общее число фирм, разрабатывающих эти средства, превышает 150. Несмотря на сложность заложенных в нейропакетах методов, использовать их довольно просто. Они позволяют сконструировать, обучить, протестировать и использовать в работе нейронную сеть на основе понимания нескольких базовых теоретических положений, изложенных выше. Первые подобные программные продукты появились на Западе в середине 80 -х годов XX века. Одним из лидеров этого рынка стал нейросетевой пакет BrainMaker американской фирмы California Scientific Software. В настоящее время нейрокомпьютерный сегмент рынка программного обеспечения бурно развивается, и появление новых пакетов – естественный процесс. Современные продукты во многом лучше своих предшественников – улучшен интерфейс пользователя, появились дополни-тельные нейропарадигмы, в пакетах реализованы возможности взаимодействия с другими приложениями посредством механизмов OLE, ActiveX и др. Плата за эти возможности использовать более мощный, чем ранее компьютер, рост требований к объему оперативной и дисковой памяти. Кроме того, новый пакет не гарантирует более качественного решения прикладной задачи пользователя. В большей степени это качество зависит от правильно поставленной задачи, выбора данных. Эффективным реальным инструментом для расчета и проектирования нейронных сетей при решении многочисленных прикладных задач во многих областях техники, в том числе электротехники и электроники, электрооборудования является пакет прикладных программ Neural Network Toolbox (ППП NNT), функционирующий под управлением системы Matlab версии 5.3 и 6.0. Следует также обратить внимание на интерфейс ППП NNT с системой Simulink, что позволяет наглядно отобразить архитектуру сети и выполнить моделирование как статических, так и динамических нейронных сетей. Последнее обстоятельство особенно важно при построении систем управления различными электротехническими объектами, диагностики их состояния и поведения.

9 Генетические алгоритмы

Эволюционные вычисления и традиционные методы оптимизации. Задача управления любым техническим комплексом или системой, включая электротехнические комплексы, является многокритериальной, то есть такой, в которой приходится учитывать большое число факторов. Здесь аналитику приходится оценивать множество сил, влияний интересов и последствий, характеризующих то или иное решение. Зачастую решения требуется принимать в режиме, близком к реальному времени. Принятие правильного решения заключается в выборе такого варианта из числа возможных, в котором с учетом всех разнообразных факторов и противоречивых требований будет оптимизирована некая общая ценность, то есть решение будет в максимальной степени способствовать достижению поставленной цели. В качестве критерия оценки качества принимаемого решения выступает некая целевая функция, аргументами которой являются количественные характеристики, описывающие состояние факторов, влияющих на достижение цели в решаемой задаче. При этом решению, приводящему к наилучшему результату, как правило, соответствует экстремальное значение целевой функции, то есть точка ее максимума или минимума.

В основе традиционных методов оптимизации лежат математические вычисления, позволяющие находить экстремум целевой функции. Ценность каждого такого метода заключается в том, что с помощью него можно найти точку экстремума целевой функции, не перебирая всех возможных комбинаций ее аргументов.

Среди многочисленных подходов можно выделить три основных типа методов поиска оптимальных решений.

- Методы, основанные на математических вычислениях подразделяются на направленные и ненаправленные. Суть ненаправленного метода состоит в том, что локальный экстремум ищется путем решения системы, как правило, нелинейных уравнений. Эта система составляется путем приравнивания градиента целевой функции к нулю (например, метод градиентного спуска или покоординатного спуска). Направленные методы строятся на перемещении от точки к точке в допустимой области, причем направление подобных перемещений связывается с направлением, на которое указывает градиент (например, метод касательных). К недостаткам этих методов можно отнести очень жесткие условия, накладываемые на целевую функцию. Она должна быть дифференцируема на всем пространстве поиска. При формализации современных задач это условие, как правило, выполнить не удается. Кроме того, данные методы находят лишь локальные экстремумы целевой функции, тогда как оптимальному решению соответствует только глобальный экстремум. Следовательно, методы поиска оптимальных решений, основанные на математических вычислениях, применимы лишь в случаях

гладких, всюду дифференцируемых целевых функций, имеющих один экстремум на пространстве поиска.

- Перечислительные методы основываются на том, что пространство поиска любой задачи можно представить в виде совокупности дискретных точек. Даже если пространство поиска непрерывно, то конечная точность представления чисел в компьютере позволяет сделать такое допущение. В этом случае поиск решения будет сводиться к перебору всех точек пространства поиска и вычислению в них целевой функции, в одной из которых она, несомненно, примет экстремальное значение. Недостаток этих методов очевиден. При увеличении числа аргументов целевой функции количество точек пространства значительно увеличивается, что приводит к значительным временным затратам и необходимости применения все более мощной и дорогостоящей вычислительной техники. В то же время размерность решаемых сейчас задач постоянно растет, а время, доступное для принятия решений сокращается. Следовательно, перечислительные методы также применимы для решения все более сужающегося класса задач.
- Методы, использующие элементы случайного поиска в пространстве задачи с сохранением наилучшего полученного результата. Очевидно, что такие методы не гарантируют нахождение оптимальных решений, но могут к ним существенно приблизиться. Для этого в методы вносится свойство детерминированности, на чем основываются эволюционные вычисления.

Эволюционные вычисления используется для описания алгоритмов поиска, основанных на некоторых формализованных принципах естественного эволюционного процесса. Основное преимущество эволюционных вычислений заключается в возможности решения задач с большой размерностью за счет сочетания элементов случайности и детерминированности точно так, как это происходит в природной среде. Детерминированность методов заключается в моделировании природных процессов отбора, размножения и наследования, происходящих по строго определенным правилам. Основным правилом при этом является закон эволюции: «выживает сильнейший», который обеспечивает улучшение находимого решения. Другим важным фактором эффективности эволюционных вычислений является моделирование размножения и наследования. Рассматриваемые варианты решений могут по определенному правилу порождать новые решения, которые будут наследовать лучшие черты своих «предков».

В качестве случайного элемента в методах эволюционных вычислений может использоваться, например, моделирование процесса мутации. В этом случае характеристики того или иного решения могут быть случайно изменены, что приведет к новому направлению в процессе эволюции решений и может ускорить процесс выработки лучшего решения. История эволюционных вычислений началась с разработки ряда независимых моделей эволюционного процесса. Среди этих моделей можно выделить три основные

парадигмы – это генетические алгоритмы, эволюционные стратегии, эволюционное программирование.

Отличительной особенностью генетических алгоритмов является представление любой альтернативы решения в виде битовой строки фиксированной длины, манипуляции с которой производятся в отсутствие всякой связи с ее смысловой интерпретацией, то есть применяется единое универсальное представление любой задачи.

Эволюционные стратегии представляют каждую из альтернатив решения единым массивом численных параметров. По существу, за каждым массивом скрывается аргумент целевой функции. Воздействие на данные массивы осуществляется с учетом их смыслового содержания и направлено на улучшение значений, входящих в них параметров.

Эволюционное программирование основано на представлении альтернатив решений в виде моделей-автоматов, описывающих средствами формальной логики возможные переходы исследуемой системы из некоторого начального состояния в заключительное состояние. Эволюционная программа при этом реализует моделирование процессов естественной эволюции моделей-автоматов, когда в каждый момент времени сохраняется тот «организм», который может наилучшим способом справится с данной задачей.

Эволюционные вычисления не гарантируют обнаружения глобального экстремума целевой функции, однако они демонстрируют эффективность решений ряда практических задач по проектированию, планированию, управлению, прогнозированию во многих областях техники. Отрицательной чертой эволюционных вычислений является то, что они представляют собой скорее подход к решению задач оптимизации, чем алгоритм. Вследствие этого они требуют адаптации к каждому конкретному классу задач путем выбора определенных характеристик и параметров.

Генетические алгоритмы. Генетический алгоритм представляет собой поисковый алгоритм, основанный на природных механизмах селекции и генетики. Он работает с кодами безотносительно их смысловой интерпретации. Поэтому сам код и его структура описываются понятием генотип, а его интерпретация, с точки зрения решаемой задачи, понятием фенотип. Каждый код представляет, по сути, точку пространства поиска. С целью максимально приблизиться к биологическим терминам, экземпляр кода называют хромосомой или особью. В обозначении строки кода будем использовать термин «особь». На каждом шаге работы генетический алгоритм использует несколько точек поиска одновременно. Совокупность этих точек в пространстве поиска является набором особей или популяцией. Количество особей в популяции называют размером популяции. Размер популяции является фиксированным и представляет одну из характеристик генетического алгоритма. Формирование исходной популяции происходит с использованием какого-либо случайного закона, на основе которого выбирается нужное

количество точек поискового пространства. На каждом шаге работы генетический алгоритм обновляет популяцию путем создания новых особей и уничтожения старых. Чтобы отличать популяции на каждом из шагов и сами эти шаги, их называют поколениями и обычно идентифицируют по номеру. Например, популяция, полученная из исходной популяции после первого шага работы алгоритма, будет первым поколением, после следующего шага - вторым, и т.д. В процессе работы алгоритма генерация новых особей происходит на основе моделирования процесса размножения. При этом, естественно, порождающие особи называются родителями, а порожденные соответственно - потомками. Родительская пара, как правило, порождает пару потомков. Непосредственная генерация новых кодовых строк из двух выбранных происходит за счет работы оператора скрещивания, который также называют кроссинговером. При порождении новой популяции оператор скрещивания может применяться не ко всем парам родителей. Часть этих пар может переходить в популяцию следующего поколения непосредственно. Насколько часто будет возникать такая ситуация, зависит от значения вероятности применения оператора скрещивания, которая является одним из параметров генетического алгоритма. Вероятность применения оператора скрещивания обычно выбирается в пределах от 0,9 до 1, чтобы обеспечить появление новых особей, расширяющих пространство поиска. При значении вероятности меньше единицы часто используют особую стратегию - элитизм, которая предполагает переход в популяцию следующего поколения элиты, то есть лучших особей текущей популяции, без всяких изменений. Применение элитизма способствует сохранению общего качества популяции на высоком уровне. При этом элитные особи участвуют еще и в процессе отбора родителей для последующего скрещивания. Количество элитных особей определяется по формуле $K = (1 - P) \cdot N$, где К – количество элитных особей, Р - вероятность применения оператора скрещивания, N – размер популяции. В случае использования элитизма все выбранные родительские пары подвергаются скрещиванию, несмотря на то, что вероятность применения оператора скрещивания меньше единицы, что позволяет сохранить прежний размер популяции.

Моделирование процесса мутации новых особей осуществляется за счет работы оператора мутации. Основным параметром оператора мутации также является вероятность мутации. Поскольку размер популяции фиксирован, то порождение потомков должно сопровождаться уничтожением других особей. Выбор пар родителей из популяции для порождения потомков производит оператор отбора, а выбор особей для уничтожения - оператор редукции. Основным параметром их работы является, как правило, качество особи, которое определяется значением целевой функции в точке пространства поиска, описываемой этой особью. В основе оператора отбора лежит принцип «выживает сильнейший». Выбор особи для размножения производится случайно. Вероятность участия особи в процессе размножения определяется

по формуле $P_i = f_i \Big/ \sum_{j=1}^{N} f_j$, где i – номер особи, P_i - вероятность участия особи в процессе размножения, f_i - значение целевой функции для i -ой особи. Очевидно, что одна особь может быть задействована в нескольких родительских парах.

Таким образом, можно перечислить основные понятия и термины, используемые в области генетических алгоритмов: генотип и фенотип; особь и качество особи; популяция и размер популяции; поколение; родители и потомки.

К характеристикам генетического алгоритма относятся: размер популяции; оператор скрещивания и вероятность его использования; оператор мутации и вероятность мутации; оператор отбора; оператор редукции; критерий останова.

Операторы отбора, скрещивания, мутации и редукции называют еще генетическими операторами.

Критерием останова работы генетического алгоритма может быть одно из трех событий:

- формирование заданного пользователем числа поколений;
- достижение популяции заданного пользователем качества;
- достижение уровня сходимости, при котором дальнейшее улучшение особей в популяции происходит чрезвычайно медленно.

Характеристики генетического алгоритма определяются путем подбора, обеспечивающего поиск лучшего решения задачи за малое время работы. Рассмотрим примеры применения генетических алгоритмов при решении оптимизационных задач.

Пример поиска одномерной функции. Пусть имеется набор натуральных чисел от 0 до 31 и функция $f(x)=x$, определенная на этом наборе чисел. Требуется найти максимальное значение функции.

При решении задачи используем в качестве кода двоичное представление аргументов функции. Это положение представляет собой фенотип решаемой задачи. Сам код будет представлять собой двоичную строку из пяти бит. Это генотип алгоритма. Функция $f(x)=x$ является целевой функцией. При задании характеристик генетического алгоритма будем исходить из следующих правил:

1. Размер начальной популяции N можно выполнять произвольным образом, например подбрасыванием монеты;
2. Вероятность оператора скрещивания $P(CO) \le 1$;
3. Вероятность оператора мутации $P(MO) \ge 0{,}001$.

Пусть процесс мутации заключается в инверсии одного из битов строки, выбираемого случайно по равномерному закону; вероятность оператора мутации равна 0,001; размер популяции $N = 4$ (табл. 5); в связи с простейшей задачей стратегия элитизма в алгоритме не применяется.

Таблица 5 Исходная популяция из четырех особей

№	Код	Значение f_i	Вероятность P_i
1	01011	11	11/43
2	10010	18	18/43
3	00010	2	2/43
4	01100	12	12/43

Предположим, что оператор отбора выбрал для производства потомков две пары строк (1,2) и (2,4). В каждой паре разбиение на подстроки оператором скрещивания происходит независимо (табл. 6).

Таблица 6 Работа оператора скрещивания

№	Родители	Потомки	Значение f_i для потомков
1	0 | 1011	00010	2
2	1 | 0010	11011	27
3	100| 10	10000	16
4	011| 00	01110	14

Пусть оператор мутации, несмотря на низкую вероятность 0,001, изменяет значение кода потомка в третьей строке (табл.6) с 10000 на 10001. Тогда за счет порожденных потомков популяция расширяется до восьми особей. Оператор редукции сокращает популяцию до исходного числа особей, исключая те особи, в которых значения целевой функции минимальны (табл. 7).

Таблица 7 Популяция первого поколения особей

№	Код	Значение f_i	Вероятность P_i
1	10010	18	18/76
2	11011	27	27/76
3	10001	17	17/76
4	01110	14	14/76

На этом шаг работы генетического алгоритма заканчивается. Очевидно, что даже за эту одну итерацию качество популяции значительно возросло. Лучшее решение увеличилось с 18 до 27 при оптимальном решении 31.

Оптимизационная задача по обучению нейронной сети. Обучение нейронных сетей является одной из основных областей применения

генетических алгоритмов. Для построения и обучения нейронной сети зададим набор примеров, который представляет собой совокупность векторов вида (X, Y), где $X = x_1, x_2, ..., x_n$ - значения всех входов нейронной сети, а $Y = y_1, y_2, ..., y_m$ - значения всех выходов нейронной сети, которые должны получаться в процессе ее работы. Структура эталонных векторов задает одновременно количество входных и выходных нейронов, то есть в данном случае n и m соответственно. Целью обучения нейронной сети является достижение такой ситуации, когда при подаче на вход сети любого вектора X из набора примеров на ее выходе получается выходной вектор Y^*, отличающийся от эталонного вектора Y не более чем на заданную заранее и вычисляемую определенным образом величину δ.

Исходными данными для решения задачи являются количество входных и выходных нейронов; набор обучающих примеров. Требуется найти следующие характеристики нейронной сети: количество скрытых слоев; количество нейронов в каждом скрытом слое; значения весов всех входов для каждого скрытого и выходного нейрона; функции активации для каждого скрытого и выходного нейрона. Очевидно, что целевой функцией в данной задаче будет максимальное значение δ, полученное среди всех векторов (X, Y) набора примеров. Подбирая значения характеристик нейронной сети и вычисляя каждый раз значение δ, можно найти такое значение δ_{max}, которое требуется. Чем меньше значение δ_{max}, тем качественнее будет построена нейронная сеть.

Таким образом, задача обучения нейронной сети сводится к задаче поиска оптимального решения. Следует заметить, что даже для простейших нейронных сетей эта задача является многомерной и крайне сложной. Цель найти нейронную сеть, удовлетворяющую условию $\delta_{max} = 0$, для реальных задач недостижима и обычно не ставится. Поэтому поиск оптимального решения превращается в поиск лучшего решения, где с успехом можно применить генетический алгоритм.

Как правило, генетические алгоритмы используются на различных этапах построения и обучения сети в качестве основного или вспомогательного средства. Генетический алгоритм может использоваться на первом этапе работы для поиска общих параметров нейронной сети: количества скрытых слоев и нейронов. Также генетический алгоритм может использоваться на заключительном этапе работы для поиска всех значений весов нейронной сети и функций активации. Причем функции активации, как правило, выбираются из ограниченного набора, а еще чаще подбирается не сама формула функции активации, а один или несколько ее параметров.

Для упрощения возьмем небольшую сеть прямого распространения (рис. 45) и построим для нее обучающий алгоритм. Сеть состоит из шести нейронов: трех входных, двух скрытых и одного выходного.

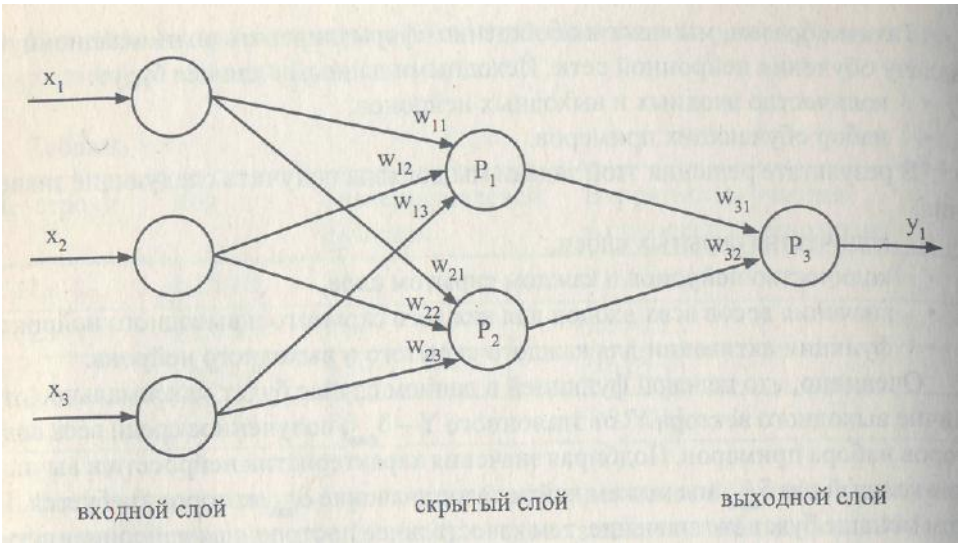


Рисунок 45 - Нейронная сеть

Используем сигмоидальную функцию качестве функции активации $OUT = 1/(1+e^{-NET})$, а для вычисления значения целевой функции выражение

$$\delta_{max} = \max_{j=1}^{k} \frac{\left| y_j^* - y_j \right|}{y_j} \cdot 100\%,$$

где k – число примеров; y_j^* - значение выхода нейронной сети для j –го примера; y_j - эталонное значение выхода нейронной сети для j – го примера.

Для останова работы генетического алгоритма можно указать число поколений, а можно задать условие на значение целевой функции. Например, остановить работу алгоритма, когда значение целевой функции $\delta_{max} \le 0{,}1\%$. При обучении нейронной сети размер популяции должен быть не менее 100 особей. В качестве кода при решении задачи будем использовать массив из 11 действительных чисел (табл. 8), по четыре на каждый скрытый нейрон и три для выходного нейрона, которые представляют собой веса соответствующих входов нейронов и параметры их функций активности. Для элементов хромосомы (особи) – генов – введем ограничения, обусловленные природой нейронных сетей: $-1 \le w_{ij} \le 1$ и $NET_i > 0$. Для оператора скрещивания можно установить вероятность применения 0,95 и использовать элитизм. В качестве оператора мутации будем использовать случайное изменение значений весов и параметра функции активации для каждого нейрона на случайную величину. Вероятность мутации 0,01. Причем одновременно будет изменяться параметр функции только для одного нейрона, и для каждого нейрона будет изменяться один из входных весов. Какие конкретно веса и параметры будут меняться, определяется по равномерному закону. Например, мутация может быть следующей: значение NET_2 увеличивается на 0,1; значение w_{13} уменьшается

на 0,01 и т.д. Исходная информация будет формироваться на основе равномерного распределения для каждого гена хромосомы (особи).

Таблица 8 Структура кода

w_{11}	w_{12}	w_{13}	NET_1	w_{21}	w_{22}	w_{23}	NET_2	w_{31}	w_{32}	NET_3

Рассмотренный процесс показывает, насколько сложной является задача построения и обучения нейросети как с точки зрения ее размерности, так и с точки зрения ее вычислительной сложности. Тем не менее, генетические алгоритмы представляют один из эффективных и изящных путей ее решения.

Эволюционные вычисления, в том числе и генетические алгоритмы, представляют собой подход к решению задачи поиска лучшего решения, а не четко определенный алгоритм. Для решения конкретной задачи, помимо ее формализации, формулировки генотипа и фенотипа, требуется создавать и конкретный генетический алгоритм. Для этого задают значения размера популяции, вероятности мутации, описывают процесс работы операторов отбора, скрещивания, мутации и редукции, что и было показано в рассмотренных примерах. Однако может оказаться, что алгоритм, успешно решающий одну задачу, совершенно не подходит для решения другой. Создание генетических алгоритмов, эффективно решающих как можно большее число задач, является предметом проводимых в настоящее время исследований.

10 Системный подход к проектированию сложных систем

Свойства сложной системы и закономерности ее функционирования. Профессиональный подход к задачам проектирования сложных систем в электротехнике, электромеханике, электрооборудовании и других областях техники является одной из важнейших предпосылок их успешного функционирования. Усложнение конструкций технических устройств и систем определяется повышением требований к ним по экономичности, безопасности, экологичности, к качеству функциональных возможностей. Модернизация и создание новых промышленных технологий, использование достижений компьютерной техники, новейших информационных и коммуникационных сред приводит к изменению подходов в формировании принципов математического моделирования сложных технических систем при их исследовании и проектировании. При этом проблемы выработки творческих решений на всех стадиях исследования и проектирования сложных систем, оценки их состояния и прогнозирования последствий в условиях неопределенности становятся актуальными и могут быть решены с применением интеллектуальных методов и компьютерных

систем, обладающих знаниями и наделенных способностью логического вывода.

Особенностью методик анализа сложных систем является то, что они опираются на понятие системы и используют общие закономерности строения, функционирования и развития систем. С философской точки зрения природа систем двойственна, то есть понятие система одновременно отражает объективное существование и субъективное восприятие некоторой реальности. В настоящее время имеется несколько известных точек зрения, раскрывающих сущность понятия система. Основоположник теории систем Л. Берталанфи определил систему как совокупность элементов, находящихся в определенных отношениях друг с другом и со средой. Значительным вкладом в теорию систем является определение системы, данное П. К. Анохиным. Системой можно назвать только такой комплекс избирательно вовлеченных компонентов, у которых взаимодействие и взаимоотношение приобретают характер взаимосодействия компонентов, направленного на получение фокусированного полезного результата, то есть системообразующим фактором является полезный результат. По утверждению Дж. Клира общее определение системы можно сделать более полезным для практики, если ввести классы элементов и отношений между ними. При классификации систем выделяют два класса системных задач: задачи исследования и задачи проектирования. Задача исследования систем состоит в накоплении знаний о свойствах и отношениях существующих объектов в соответствии с конкретными целями. Задача проектирования систем заключается в создании новых объектов с заданными свойствами.

Сложным системам присущи следующие основные закономерности:

- целостность - свойства системы не являются суммой свойств ее элементов, хотя и зависят от них;
- коммуникативность - любая система является подсистемой в системе более высокого уровня;
- иерархичность;
- эквифинальность - способность системы достигать не зависящего от времени состояния, полностью детерминированного начальными условиями;
- историчность;
- закон необходимого разнообразия;
- закономерности целеобразования - зависимость целей от уровня познания объекта, а также от внешних и внутренних факторов.

Функционирование любой сложной системы подчиняется восьми законам композиции.

1. Перевод системы из одного качественного состояния в другое путем минимального воздействия в критическую точку фазового перехода системы.

2. Закон эволюции, который утверждает, что любая система в процессе развития проходит в сокращенной форме собственный эволюционный путь, включая все его этапы.

3. Закон пирамиды, который гласит, что коэффициент полезного действия любой реальной системы не может достигать 100 %, в связи с чем энергия, почерпнутая системой извне, постепенно уменьшается по мере приближения к конечной цели.

4. Закон «островного эффекта», позволяющий определить возможную степень автономности системы в зависимости от ее параметров и от свойств окружения.

5. Закон единства и борьбы противоположностей, определяющий возможность и условия объединения противоборствующих сторон.

6. Закон причинно-следственных связей.

7. Закон проявления нестабильностей системы, которые выражены нарушением согласованного (когерентного) взаимодействия с фоном этой системы.

8. Закон существенной зависимости потенциала системы от изменения характера взаимодействия между ее элементами.

Свойства сложных систем можно разделить на три основные группы.

- Свойства, определяющие взаимодействие системы с внешней средой. Важнейшие среди них - это устойчивость и характеристики состояний системы;
- Свойства, характеризующие внутреннее строение системы. Структура систем любой природы может изменяться как в результате взаимодействия с внешней средой, так и в результате протекания внутренних процессов. Параметром, характеризующим изменение структуры во времени, является энтропия. В открытых системах энтропия может не только увеличиваться, но и уменьшаться за счет ее увеличения во внешней среде;
- Интегральные свойства, описывающие поведение системы. К ним относятся полезность, эффективность, надежность, управляемость, безопасность, живучесть и др. В общем случае интегральные свойства сложной системы не являются суммой свойств ее частей, то есть нарушение принципа суперпозиции.

Характеристика системного подхода к исследованию сложных систем.

Основным методом исследования сложных систем является системный анализ, состоящий из нескольких этапов:

- постановка задачи;
- формирование описания системы;
- выбор наилучших решений.

На этапе постановки задачи определяются цели исследования, производится выделение системы из среды, рассматриваются способы взаимодействия системы со средой, формулируются основные допущения.

Этап формирования описания системы включает следующие действия:

- расчленение системы на элементы;

- выделение подсистем;
- определение общей структуры системы;
- определение связей системы со средой и выявление внешних факторов, подлежащих учету;
- выбор подхода к представлению системы и формирование вариантов ее представления.

Этап выбора наилучших решений сводится к определению целевой функции и нахождению такой комбинации значений характеристик сложной системы, которая приводит к ее экстремальному значению.

Системный подход к проектированию заключается в рассмотрении всего комплекса проблем, возникающих в течение жизненного цикла исследуемой системы:

- из неразрешимости общей задачи проектирования вытекает необходимость ее декомпозиции на совокупность локальных задач, упорядоченных многоуровневой параллельно-последовательной логической схемой проектирования;
- из неопределенности исходных данных и ограничений в общей задаче проектирования вытекает необходимость их прогнозирования и обмена проектными решениями между функциональными ячейками системы проектирования в соответствии с определенной логической схемой;
- из логической противоречивости общей задачи проектирования вытекает необходимость организации итерационных циклов, которые определяют сходимость системных решающих процедур;
- из невозможности сконструировать априори «сквозное» правило предпочтения следует необходимость «индивидуального» построения многоуровневого критерия оценки проектных решений, который может быть получен эвристически только в конце итерационного цикла.

Построение формального описания сложной системы является необходимым этапом исследования. Формальные модели нужны для изучения внутреннего строения систем, для прогнозирования, а также для определения оптимальных режимов функционирования. Высокий уровень абстрагирования имеют лингвистический и теоретико-множественный способы описания систем. Лингвистический подход к описанию систем исходит из «характерных черт» системы. Этот подход весьма привлекателен, так как знания в виде высказываний являются наиболее доступным видом информации. Теоретико-множественный подход к описанию систем является наиболее распространенным. При этом наиболее развитым направлением теории систем является феноменологическое, которое основано на представлении системы S как некоторого преобразования входных воздействий X в выходные величины Y. В условиях неопределенности традиционным приемом является усиление степени размытости языка описания, поэтому часто в таких случаях переходят от рассмотрения входных и выходных величин к рассмотрению подмножеств их элементов. Это приводит к вероятностным и нечетким описаниям систем.

Существуют разные подходы к формированию целей проектирования. Сложившийся на практике стереотип - ясно поставленная цель есть главнейшее условие успеха - противоречит опыту специалистов по системному анализу, утверждающих, что цель создания системы и ее составляющие уточняются в ходе работы. В связи с этим одним из условий, обеспечивающих успех любого проекта, является построение следующих прогнозов:

- прогноз состояния или поведения внешней среды (надсистемы), взаимодействующей с исследуемой системой (объектом);
- прогноз изменения целей функционирования и структуры исследуемой системы;
- прогноз развития конкурирующих или противоборствующих подсистем, их характеристик и стратегий поведения.

Взаимодействие системы со средой, а также элементов системы друг с другом может быть представлено моделями структуры и моделями функционирования. Модель структуры в зависимости от цели исследования может иметь следующие разновидности:

- внешняя модель - система представляется в каноническом виде, а ее связи с внешней средой выражаются посредством входов и выходов;
- иерархическая модель - система расчленяется по уровням согласно принципу подчинения низших уровней высшим;
- внутренняя модель - отражает состав и взаимосвязь между элементами системы.

Функционирование системы может быть представлено:

- моделью жизненного цикла системы, характеризующей процесс существования системы от ее замысла до гибели;
- операциональной моделью системы, представляющей совокупность процессов ее функционирования по основному назначению.

Характеристика подходов к моделированию сложных систем. Статистический подход основан на построении макромоделей больших систем, которые могут использовать различные типы описаний: теоретико-множественный, лексикографический, топологический и др. Макромодели дают представление о зависимости состояния системы от поведения человека и среды, но не учитывают состязательный характер развития и не содержат представлений об эффективности.

Структурно-функциональный подход связан с построением модели структуры, элементами которой являются функции. Для построения механизма, порождающего функции, используют теоретико-множественный аппарат и математико-лингвистические средства.

Ситуационное моделирование разработано для задач динамического управления сложными системами в условиях неопределенности. Система описывается конечным набором возможных ситуаций и соответствующих им управленческих решений. Главным условием применимости этого подхода является возможность классификации ситуаций.

Имитационное моделирование основано на использовании субъективных предположений исследователей о динамике рассматриваемых процессов. При моделировании динамики сложных систем дифференциальные уравнения можно составить только для идеализированной структуры при усредненных значениях параметров. Усложнение обычно сопровождается переходом к моделям, распределенным в пространстве. При этом возникают серьезные затруднения с заданием граничных условий, определением неизвестных коэффициентов уравнений, а также сложности вычислительного характера.

Синергетический подход к моделированию сложных систем учитывает нелинейность сложных систем, одним из проявлений которой является нарушение принципа суперпозиции. В данном подходе используется динамическая имитационная модель системы, описывающая процесс ее развития. Основополагающие понятия в теории развития - информация и энтропия. Энтропия есть мера недостатка информации о действительной структуре системы. Для эволюции существенно не количество информации, а ее ценность, которая непосредственно связана со степенью использования информации в системе. Ценность информации оказывается тем больше, чем меньше существует способов выполнить нужную функцию без данной информации. Другими словами, ценность характеризует незаменимость информации. В процессе использования информации происходит отбор тех элементов-признаков, которые дают преимущества в ходе развития. Согласно концепции И. Пригожина абсолютно устойчивая система не способна к развитию, так как она подавляет любые отклонения от устойчивого состояния.

Задачи проектирования систем всегда предполагают решение задач синтеза, которые в зависимости от исходных данных разделяют на три класса.

1. Синтез структуры системы при заданных алгоритмах функционирования;
2. Синтез оптимального поведения и алгоритмов функционирования системы при известной структуре;
3. Синтез структуры и алгоритмов функционирования системы, распределение функций по элементам и определение их оптимального состава.

Требования, предъявляемые к синтезируемым системам, зависят от конкретных условий и определяющим образом влияют на качество получаемых решений. Проблема выбора оптимальных решений из множества синтезированных объектов осложняется по следующим причинам:

- объекты выбора (альтернативы) реально не существуют;
- отсутствует возможность количественного или объективного измерения свойств проектируемой системы;
- представление о «наилучшем решении» может отсутствовать или быть неоднозначным и противоречивым;
- необходим учет многих критериев в процессах принятия решений;

- требования к проектируемой системе изменяются с течением времени.

Задача структурного синтеза считается наиболее трудной для формализации проектной процедурой. Для синтеза необходима информация о базовых элементах системы, макроэлементах (типовая совокупность взаимосвязанных элементов) и обобщенных структурах проектируемого объекта. Автоматизация структурного синтеза, как правило, сопряжена с разработкой интеллектуальных систем. Для представления знаний об объектах синтеза обычно используются объектно-ориентированные модели. Распространенным средством представления обобщенных структур являются И-ИЛИ-деревья. Известны следующие подходы к алгоритмизации структурного синтеза: перебор законченных структур; наращивание структуры; выделение варианта из обобщенной структуры; трансформация готовых описаний.

Отсутствие строгого общего решения задачи синтеза сложной неоднородной системы обусловлено следующими причинами:

- начальная информация о составе элементов и отношений между ними ограничена, различным уровням описания системы соответствуют свои составы элементов;
- возможны альтернативные способы членения сложной системы на элементы;
- не все свойства системы можно получить суммированием свойств ее частей;
- в процессе формализации связи между элементами обычно выражаются через бинарные отношения или отношения, сводимые к бинарным, но известно, что система, будучи целостной организацией, не сводима к бинарным отношениям;
- проектирование новых систем часто является поиском принципиально новых решений, следовательно, эффективный метод решения задачи должен включать процессы, ведущие к образованию новых структур с новым составом элементов и отношениями между ними.

I

I

Лабораторная работа № 1. Использование семантических сетей для представления знаний

Цель работы: Научиться использовать семантические сети для представления знаний в интеллектуальных системах.

1. Теоретическая часть

Семантическая сеть – это один из способов представления знаний. Изначально семантическая сеть была задумана как модель представления долговременной памяти в психологии, но впоследствии стала одним из способов представления знаний в экспертной системе (ЭС).

Семантика – означает общие отношения между символами и объектами из этих символов (рис.46).

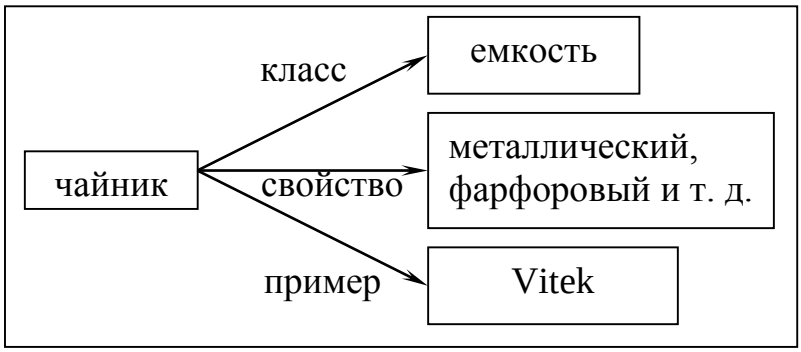


Рисунок 46- Простейший образец семантической сети

Вершины – это объекты, дуги – это отношения. Семантическая модель не раскрывает сама по себе каким образом осуществляется представление знаний, поэтому семантическая сеть рассматривается как метод представления знаний и структурирования знаний. При расширении семантической сети в ней возникают другие отношения:

IS – A (принадлежит) и PART OF (является частью) отношение:

целое → часть.

Например: Ласточка IS – A птица, «нос» PART OF «тело» (рис.47).

Могут быть и другие отношения: владеет. Тогда семантическая сеть расширяется иерархически (вершина имеет две ветви). Кроме того, можно расширить сеть и другим отношением:

период → «весна – лето».

Получается иерархическая структура понятия АКО («A Kind Of», «разновидность»), которую можно разбить на подсхемы. Большой проблемой для семантических сетей является то, что результат вывода не гарантирует достоверности, так как вывод есть просто наследование свойств ветви is-a.

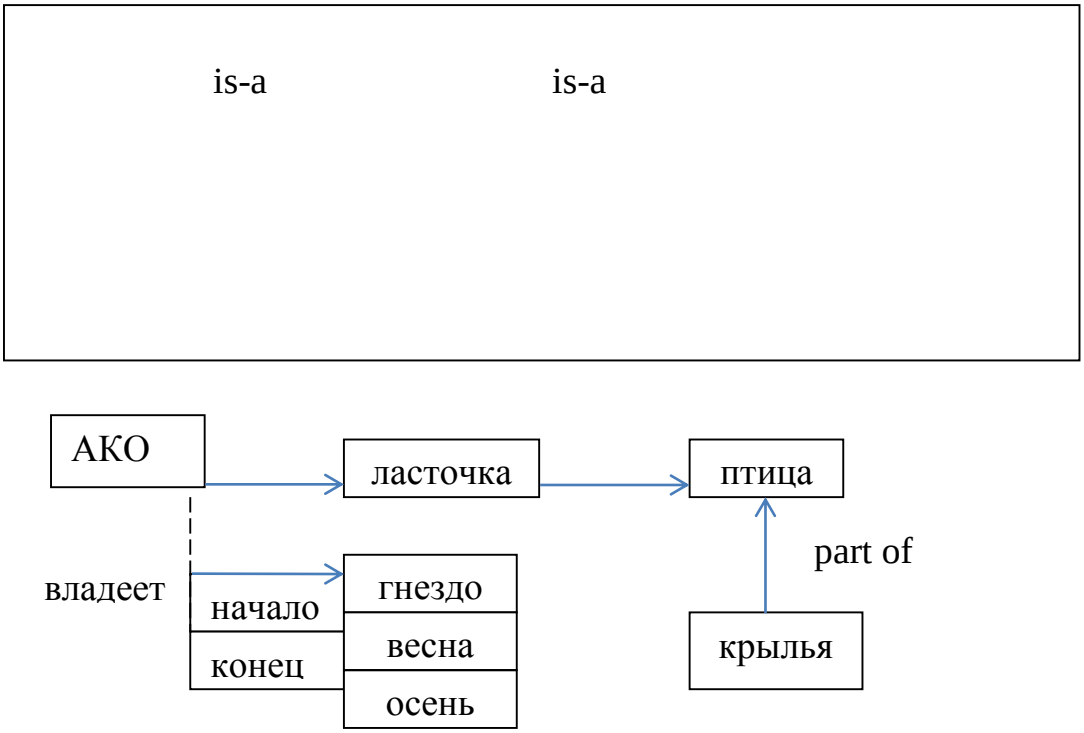


Рисунок 47 - Расширение семантической сети

Для отображения иерархических отношений между объектами и введения единой семантики в семантические сети, было предложено использовать процедурные сети. Каждый узел такой сети представляет концепцию, а дуги используются для определения отношений между концепциями. В самом общем случае семантическая сеть представляет собой информационную модель предметной области и имеет вид графа, вершины которого соответствуют объектам предметной области, а дуги — отношениям между ними. Дуги могут быть определены разными методами, зависящими от вида представляемых знаний. Обычно дуги, используемые для представления иерархии, включают дуги типа «множество», «подмножество», «элемент». Семантические сети, применяемые для описания естественных языков, используют дуги типа «агент», «объект», «реципиент».

Характерной особенностью семантических сетей является обязательное наличие трех типов отношений:

- ✓ класс — элемент класса;
- ✓ свойство — значение;
- ✓ пример элемента класса.

2. Порядок выполнения работы

1. Изучить теоретическую часть по приведенным выше данным и дополнительной литературе.
2. Получить у преподавателя вариант задания для выполнения.
3. Построить семантическую модель заданного объекта.

3. Варианты заданий

Используя соответствующие дуги построить семантическую сеть, касающуюся:

1) географии какого-либо региона. Дуги: государство, страна, континент, широта.
2) диагностики глазных заболеваний. Дуги: категории болезней, патофизиологическое состояние, наблюдения, симптомы.
3) распознавания химических структур. Дуги: формула вещества, свойства вещества, область применения, меры предосторожности.
4) процедуры поиска полезных ископаемых. Дуги: наименование ископаемого, расположение месторождения, глубина залегания, методы добычи.
5) судебной процедуры. Дуги: юридическое лицо, событие, меры воздействия, способы расследования.
6) распределения продуктов по магазинам. Дуги: источник снабжения, наименование продукта, способ транспортировки, конечный пункт транспортировки.
7) определения принадлежности животного к определенному виду, типу, семейству. Дуги: место обитания, строение, особенности поведения, вид питания.
8) классификации пищевых продуктов. Дуги: наименование продукта, составляющие части, способ приготовления, срок хранения.
9) распознавания типа компьютера. Дуги: страна изготовитель, стандартная конфигурация, область применения, используемое программное обеспечение.
10) иерархической структуры БД. Дуги: система, состояние, назначение, взаимодействие составляющих
11) описание студентов. Дуги: вуз, факультет, группа, предмет и т.д.
12) описание преподавателя. Дуги: вуз, кафедра, предмет и др.
13) процесс обучения. Дуги: экзамен, ВКР, зачисление и т.д.
14) строительство дома. Дуги: место, фундамент, тип стен и т.д.
15) уборка урожая. Дуги: сроки, тип культуры, оборудование.

4. Контрольные вопросы

1. Что такое семантическая сеть и для чего ее применяют?
2. В чем состоит идея создания семантической сети?
3. Каким образом представляются данные в семантической сети?
4. Существуют ли ограничения на число связей элементов, свойств и сложность при построении семантической сети?
5. Какие отношения предложены в качестве операторов отношения для группировки вершин?

Лабораторная работа № 2. Использование фреймов для представления знаний

Цель работы: Научиться использовать фреймы для представления знаний в интеллектуальных системах.

1. Теоретическая часть

Фреймы - один из распространенных формализмов представления знаний в ЭС. Фрейм можно представить себе как структуру, состоящую из набора ячеек - слотов. Каждый слот состоит из имени и ассоциируемых с ним значений. Значения могут представлять собой данные, процедуры, ссылки на другие фреймы или быть пустыми. Такое построение оказывается очень удобным для моделирования аналогий, описания областей с родовидовыми связями понятий и т.п.

Любой фрейм состоит из некоторых составляющих, имена и содержание которых описано ниже:

1. Имя фрейма. Это идентификатор, присваиваемый фрейму, фрейм должен иметь имя уникальное в данной фреймовой системе.
2. Имя слота. Это идентификатор, присваиваемый слоту; слот должен иметь уникальное имя во фрейме, к которому он принадлежит. Обычно имя слота не несет никакой смысловой нагрузки и является лишь идентификатором данного слота.
3. Указатели наследования. Эти указатели касаются только фреймовых систем иерархического типа, основанные на отношениях “абстрактное-конкретное”, они показывают, какую информацию об атрибутах слотов во фрейме верхнего уровня наследуют слоты с такими же именами во фрейме нижнего уровня. Типичные указатели наследования Unique (U: - уникальный), Same (S: такой же), Range (R: установление границ), Override (O: игнорировать) и т.п. U показывает, что фрейм может иметь слоты с разными значениями: S - все слоты должны иметь одинаковые значения, R - значение слотов фрейма нижнего уровня должны находиться в пределах, указанных значениями слотов фрейма верхнего уровня, O - при отсутствии указания значение слота фрейма верхнего уровня становится значением слота фрейма нижнего уровня, но в случае определения нового значения слотов фреймов нижних уровней указываются в качестве значений слотов.
4. Указание типа данных. указывается, что слот имеет численное значение, либо служит указателем другого фрейма. К типам данных относятся:
   FRAME (указатель), INTEGER (целый), REAL (действительный), BOOL (булев), LISP (присоединенная процедура), TEXT (текст), LIST (список), TABLE (таблица), EXPRESSION (выражение) и др.

5. Значение слота. Пункт ввода значения слота. Значение слота должно совпадать с указанным типом данных этого слота, кроме того должно выполняться условие наследования.
6. Демон. Здесь дается определение демонов типа IF-NEEDED, IF-ADDED, IF-REMOVED и т.д. Демоном называется процедура, автоматически запускаемая при выполнении некоторого условия. демоны запускаются при обращении к соответствующему слоту. Кроме того, демон является разновидностью присоединенной процедуры.
7. Присоединенная процедура. В качестве значения слота можно использовать программу процедурного типа. Когда мы говорим, что в моделях представления знаний фреймами объединяются процедурные и декларативные знания, то считаем демоны и присоединенные процедуры процедурными знаниями.

Особенностью иерархической структуры является то, что информация об атрибутах фрейма на верхнем уровне совместно используется всеми фреймами нижних уровней, связанных с ним.

Например: Фреймовое представление конференции.

Иерархические фреймовые структуры базируются на отношениях IS – A между фреймами, описывающими некоторую конференцию. Все фреймы должны содержать информацию о ДАТЕ, МЕСТЕ, НАЗВАНИИ ТЕМЫ, ДОКЛАДЧИКЕ. Таким образом, на самом верхнем уровне определен фрейм КОНФЕРЕНЦИЯ (рис.48).

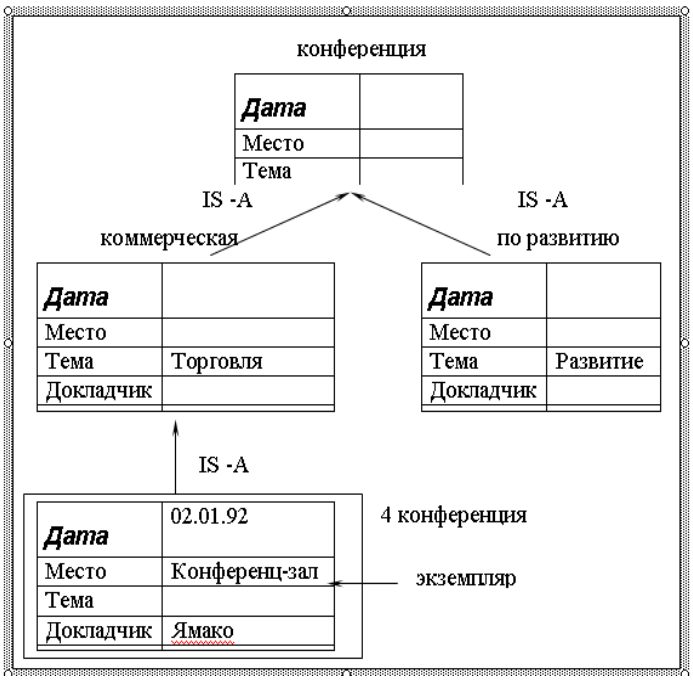


Рисунок 48 - Пример фреймовой модели

Конференции разделяются на коммерческие и по развитию. Они составляют дочерние фреймы. В них могут быть добавлены слоты: объем торговли и бюджет.

2. Порядок выполнения работы:

1. Изучить теоретическую часть по приведенным выше данным и дополнительной литературе.
2. Получить у преподавателя вариант задания для выполнения.
3. Построить фреймовую модель заданного объекта.

3. Варианты заданий

Используя фреймовую модель представления знаний реализовать структуру отношений, описывающие следующие ситуации:

1. экзамен по дисциплине за семестр у преподавателя при составляющих: семестр, экзамен, преподаватель, оценка, студент, получать.
2. ведомость при составляющих: дисциплина, студент, экзамен, семестр, преподаватель, оценка.
3. конференция по коммерческим вопросам при составляющих: дата, место проведения, тема, цель выступающие.
4. получение оценки при составляющих: преподаватель, студент, оценка, получать.
5. использования изделия при составляющих: организация, разработка технологического решения, исследование «физического эффекта», методы создания изделия.
6. информационная структура БД в машиностроении при составляющих: физические эффекты, технические решения, изделия, объект поставки изделия, приборы и стенды, нормативы.
7. классификация продукта при составляющих: название, область применения, способ хранения, способ транспортировки.
8. аудитория (описание) при составляющих: вместимость, назначение, составляющие, местонахождение.
9. животный мир при составляющих: вид, тип, среда обитания, особенности поведения.
10. обучение студентов при составляющих: вуз, факультет, группа, предмет и т.д.
11. строительство дома при составляющих: место, фундамент, тип стен и т.д.
12. уборка урожая при составляющих: сроки, тип культуры, оборудование.
13. классификация пород собак при составляющих: тип, порода, масть, характеристики и т.д.

4. Контрольные вопросы

1. Что представляет из себя фрейм, его составные части?
2. Что такое слот и из каких частей он состоит?
3. Для чего служат имя фрейма и имя слота?
4. Для чего служат указатели наследования?
5. для чего служат указание типа данных, демон?
6. Для чего служат присоединенная процедура и значение слота?

Лабораторная работа №3. Описание предметной области. Разработка базы фактов и правил интеллектуальной системы

Цель работы: Научиться строить модель предметной области, описывать решаемую задачу правилами продукционной системы и формализовать используемые знания.

1. Теоретическая часть

В данной работе рассмотрим построение базы знаний на основе сведений, полученных от эксперта. Процесс ее построения состоит из двух этапов:

- описание предметной области;
- выбор метода и модели представления знаний.

Инженер знаний должен корректно сформулировать задачу. В то же время он должен уметь распознать, что задача не структурирована, и в этом случае воздержаться от попыток ее формализовать или применить систематические методы решения. Главная цель начального этапа построения базы знаний - определить, как будет выглядеть описание предметной области на различных уровнях абстракции. Экспертная система включает базу знаний, которая создается путем формализации некоторой предметной области, а та в свою очередь является результатом абстрагирования определенных сущностей реального мира.

После того как предметная область выделена, инженер знаний должен ее формально описать. Для этого ему необходимо выбрать какой-либо способ представления знаний о ней (модель представления знаний). В. настоящее время отсутствует общий способ представления знаний, который бы годился для формализации предметных областей любой природы. Инженер знаний должен воспользоваться той моделью, с помощью которой можно лучше всего отобразить специфику предметной области. Когда будет создана общая теория представления знаний (если это вообще когда-нибудь произойдет), ее можно будет применять для формализации новых предметных областей без учета их особенностей.

Определение характера решаемых задач

Обратимся к примеру из медицинской практики. Предположим, что мы хотим построить экспертную систему, предназначенную для обработки результатов химического анализа крови, выполненного в лаборатории. Инженер знаний, прежде всего, обязан провести опрос эксперта и только

потом приступать к построению системы. Эксперт, безусловно, должен быть специалистом в той области, в которой будет работать система. Первым делом необходимо определить целевое назначение системы. Какие, собственно 'задачи предстоит решать системе, основанной на знаниях? Цели разработки системы следует сформулировать точно, полно и непротиворечиво. Например, для диагностической системы это может быть получение ответов на такие вопросы:

1. Здоров ли пациент (исправна ли система)? Если нет, то какое именно у него заболевание? Если имеете» несколько заболеваний, то какое из них наиболее опасно?

2. Какие изменения в диете и рационе питания следует рекомендовать и, какие из них считаются особенно важными?

3. Какие лабораторные исследования необходимо провести дополнительно и, какие из них являются первоочередными?

4. Как нужно изменить образ жизни пациента или климатические условия, в которых он находится?

5. Нужно ли направить пациента для обследования к врачам-специалистам и если да, то к каким именно? Подумайте, на какие еще вопросы должна уметь отвечать наша диагностическая система?

После того как цель разработки системы определена, инженер знаний приступает к формулированию подцелей. Это поможет ему установить иерархическую структуру системы и разбить ее на модули. Введение тех или иных подцелей обусловливается наличием связей между отдельными фрагментами знаний. Проблема сводится к разбиению задачи на две или несколько подзадач меньшей сложности и последующему поиску их решений. При необходимости, полученные в результате разбиения подзадачи могут дробиться и дальше.

Выявление объектов предметной области

Следующим шагом построения базы знаний является выделение объектов предметной области, или в терминах теории систем установление границ системы. Как и формальная система, *совокупность* выделенных понятий должна быть точной, полной и непротиворечивой. Итак, какие конкретно лабораторные анализы необходимо провести? Следует ли обратиться к истории болезни пациента и если да, то какие данные в ней наиболее важны? Какие еще сведения о пациенте могут представлять интерес (например, отмечались ли раковые заболевания у родственников)? Нужно ли учитывать лекарства, которые больной принимал ранее, а также предыдущие назначения врачей? Играет ли какую-нибудь роль род занятий и образ жизни больного, климатические условия и режим питания? Какие симптомы у него наблюдаются (головные боли, жар и т.д.)?

Установление взаимосвязей между объектами

После выявления объектов предметной области необходимо установить, какие между ними имеются связи. Например, низкое содержание тиреотропного гормона в крови может свидетельствовать о повышенной

активности поджелудочной железы, но может означать и нечто другое. Следует стремиться к выявлению как можно большего количества связей, в идеале - всех, которые существуют в предметной области.

Формализация знаний

Полученное качественное описание предметной области должно быть представлено средствами какого-либо формального языка, чтобы привести это описание к виду, позволяющему поместить его в базу знаний системы. Для решения этой задачи выбирается подходящая модель представления знаний, с помощью которой сведения о предметной области можно выразить формально.

Наиболее эффективным является применение продукционной модели, или модели, основанной на правилах, которая позволяет представить знания в виде предложений типа: ЕСЛИ (условие), ТО (действие).

Под условием понимается некоторое предложение-образец, по которому осуществляется поиск в базе знаний, а под действием — действия, выполняемые при успешном исходе поиска (они могут быть промежуточными, выступающими далее как условия, и терминальными или целевыми, завершающими работу системы).

При использовании продукционной модели база знаний состоит из набора правил. Программа, управляющая перебором правил, называется машиной вывода. Чаще всего вывод бывает прямой (от данных к поиску цели) или обратный (от цели для ее подтверждения – к данным). Данные — это исходные факты, на основании которых запускается машина вывода.

Если в памяти системы хранится некоторый набор продукций, то они образуют систему продукций. В системе продукций должны быть заданы специальные процедуры управления продукциями, с помощью которых происходит актуализация продукций и выполнение той или иной продукции из числа актуализированных.

Рассмотрим пример.

Подходящей задачей, при решении которой можно использовать продукционную модель представления знаний, может быть задача, вытекающая из следующей ситуации: к директору крупной технической фирмы пришёл человек, желающий устроиться на работу. Директор располагает сведениями о его квалификации, о потребностях фирмы в специалистах и общем положении дел в фирме. Ему нужно решить, какую должность в фирме может занять посетитель.

Рассмотрим модель «Посетитель», выявим необходимые атрибуты для принятия решения о приеме на работу (рис.49).

Объект: посетитель.

Атрибуты:

1) наличие ученого звания,
2) стаж работы по специальности,

3) посетитель сделал важное открытие,
4) средний бал посетителя за время учебы.

Рисунок 49 - Модель предметной области

2. Порядок выполнения работы

1. Проанализировать полученное задание
2. Определить характер решаемой задачи.
3. Выделить объекты предметной области.
4. Выбрать атрибуты, свойства характеризующие объекты.
5. Установить связи между объектами в виде правил продукционной системы.

3. Варианты заданий

Описать предметную область для следующих задач:

1) диагностика неисправностей электронной аппаратуры,
2) диагностика неисправностей автомобиля,
3) диагностика заболеваний (по выбору),
4) прогнозирование (по выбору)
    a. спортивных мероприятий
    b. телепередач
    c. природных катаклизмов
    и т.п.
5) классификация объектов (по выбору),
6) задачи информационно-советующего характера (по выбору)
    a. помощник заведующего склада
    b. помощник аптекаря
    c. помощник оператора справочной службы
    d. выбор должности

е. проведение отпуска

и т.п.

4. Контрольные вопросы

1. Какие модели представления знаний существуют?
2. Какие задач может решать экспертная система?
3. Чем характеризуются объекты предметной области?
4. Как могут быть представлены факты в ЭС?

Лабораторная работа № 4. Использование продукционной модели для представления знаний. Прямая цепочка рассуждений

Цель работы: Научиться использовать продукционную модель для представления знаний на основе прямой цепочки рассуждений.

1. Теоретическая часть

Представление знаний с помощью продукционной модели – самая распространенная форма реализации базы знаний (БЗ). С помощью продукции можно описать практически любую систему знаний.

Правила продукций представлены в виде импликации:

$$p_i : s_i \rightarrow d_i ,$$

где p_i - правило продукции,

s_i - условие применения правила,

d_i - результат применения правила (рис.50).

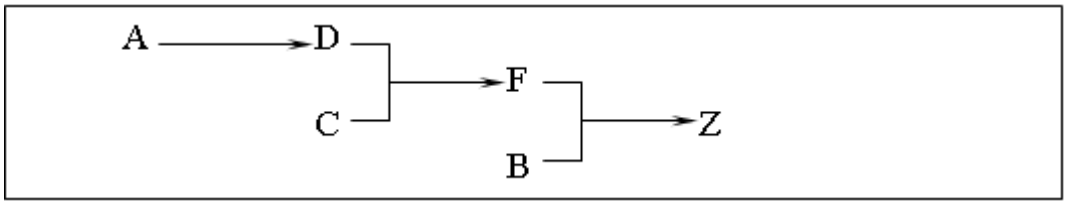


Рисунок 50 - Пример использования правил продукции:

1. Если есть цены на выпускаемые изделия (A) - завод отпускает продукцию (D).
2. Если завод выпускает продукцию и выполняет план по ее реализации (C) - рабочие получают премию (F).
3. Если рабочие получают премию и растет производительность производства (B) - завод производит продукцию сверх плана (Z).

Рассмотрим цепочки выводов.

Прямой способ рассуждения

По известным фактам отыскивается заключение, которое следует из этих фактов и накапливается рабочая память.

Это приводит к выполнению 2 правила.

C & D→ F, и факт «F» помещается в рабочую память. Тогда опять проверяются правила из базы. Первое правило выполняется F & B → Z, вследствие этого Z заносится в рабочую память. А так как Z является целью, то поиск заканчивается. Этот метод называется прямой цепочкой рассуждений, поскольку поиск новой информации происходит в направлении стрелок, разделяющих левые и правые части правил (рис.51).

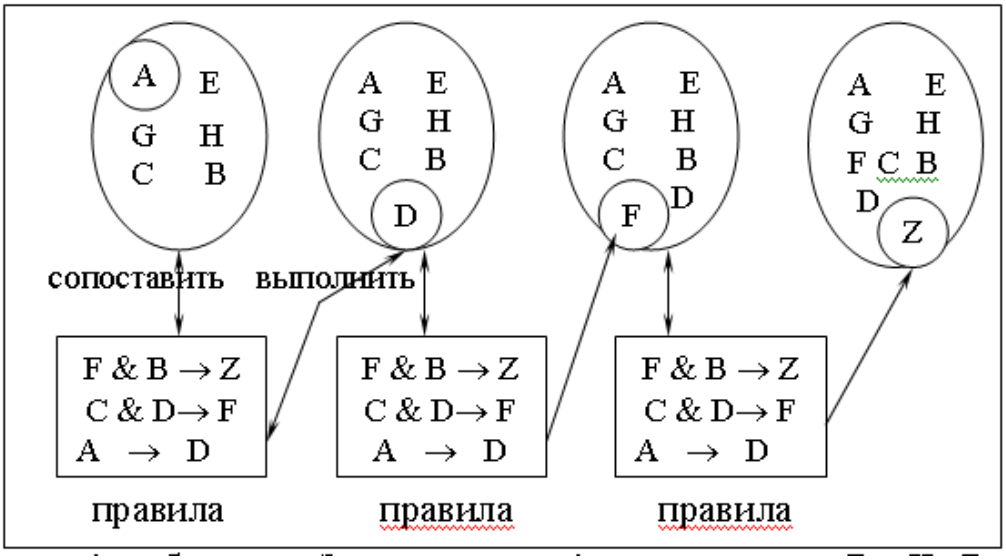


Рисунок 51 - Пример реализации прямой цепочки рассуждений

Обобщённый алгоритм работы системы, реализующий прямую цепочку рассуждений, можно свести к следующему:

1. Определить исходное состояние.
2. Занести переменную условия в очередь переменных логического вывода, а её значение - в список переменных.
3. Просмотреть список переменных и найти ту переменную, имя которой стоит в начале очереди переменных логического вывода. Если переменная найдена, записать в указатель переменных условия номер правила и число 1. Если переменная не найдена, перейти к шагу 6.
4. Присвоить значения не проинициализированным переменным условной части найденного правила (если такие есть). Имена переменных содержатся в списке переменных условия. Проверить все условия правила и в случае их истинности обратиться к части ТО правила.
5. Присвоить значение переменной, входящей в часть ТО правила, и поместить её в конец очереди переменных логического вывода.
6. Удалить переменную, стоящую в начале очереди переменных логического вывода, если она больше не встречается в условной части какого-либо правила.

Закончить процесс рассуждений, как только опустеет очередь переменных логического вывода. Если же в очереди ещё есть переменные, вернуться к шагу 3.

2. Порядок выполнения работы:

1. Изучить теоретическую часть по приведенным выше данным и дополнительной литературе.
2. Получить у преподавателя вариант задания для выполнения.
3. Построить прямую цепочку рассуждений.

3. Варианты заданий

Реализовать прямую цепочку рассуждений для следующих задач:

1) прогнозирование неисправностей электронной аппаратуры,
2) прогнозирование неисправностей автомобиля,
3) прогнозирование заболеваний (по выбору),
4) прогнозирование (по выбору)
    a. спортивных мероприятий
    b. телепередач
    c. природных катаклизмов
    и т.п.
5) классификация объектов (по выбору),
6) задачи информационно-советующего характера (по выбору)
    a. помощник заведующего складом
    b. помощник аптекаря
    c. помощник оператора справочной службы
    d. выбор должности
    e. проведение отпуска
    и т.п.

4. Контрольные вопросы

1. Что такое правила продукции и в чем их сущность?
2. В чем отличие прямой цепочки рассуждений от обратной цепочки рассуждений?
3. Из каких частей состоит продукционная система?
4. Значение и применение частей продукционной системы для представления знаний?

Лабораторная работа № 5. Использование продукционной модели для представления знаний. Обратная цепочка рассуждений

Цель работы: Научиться строить дерево целей и разрабатывать алгоритм решений на основе обратной цепочки рассуждений.

1. Теоретическая часть

Прямой метод рассуждений имеет следующий недостаток. При большом количестве правил, чтобы найти информацию, связанную с Z,

нужно выполнить много правил, не связанных с Z. При этом метод оказывается напрасной тратой времени и денег.

В таких ситуациях более рентабельной является обратная цепочка рассуждений (рис.52).

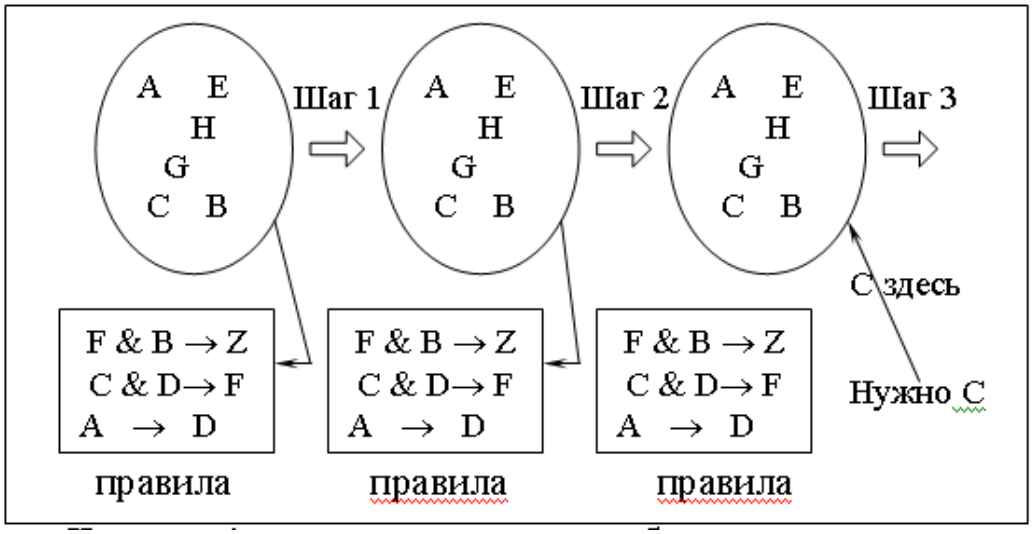


Рисунок 52 - Пример реализации обратной цепочки рассуждений

При этом методе система начинает с того, что нужно доказать, например, что ситуация Z существует, и нужно выполнить только те правила, которые относятся к установлению этого факта.

На шаге 1 системе говорится, что ситуация Z существует, система ищет Z в базе, а если Z нет, будет искать правило, приводящее к установлению Z. Она находит правило

$F \& B \rightarrow Z$

и решает, что надо установить F и B.

На шаге 2 система пытается найти факт F в базе знаний или среди правил. Находит правило $C \& D \rightarrow F$ и решает, что необходимо установить существование фактов C и D.

На шагах 3-5 система находит C, затем находит A прежде, чем получит заключение о D.

На шагах 6-8 система выполняет третье правило, чтобы установить D, затем исполняет второе правило, чтобы установить F и наконец – первое правило, чтобы установить основную цель – факт существования Z.

Теперь нужно наглядно её представить. Для описания подобных задач обычно используются диаграммы, которые называются ***деревьями решений.*** Деревья решений дают необходимую наглядность и позволяют проследить ход рассуждений.

Диаграммы называются деревьями решений потому, что, подобно настоящему дереву, имеют ветви. Ветви деревьев решений заканчиваются логическими выводами. Для рассматриваемого примера вывод заключается в том, предложит ли директор должность поступающему на работу, и если да, то какую. Многие задачи сложны, и их непросто представить (или для их

решения не собираются использовать ЭС). Дерево решений помогает преодолеть эти трудности.

На рисунке 53 показано дерево решений для примера с приёмом на работу. Видно, что диаграмма состоит из кружков и прямоугольников, которые называются вершинами. Каждой вершине присваивается номер. На вершины можно ссылаться по этим номерам. Линии, соединяющие вершины, называются дугами или ветвями. Кружки, содержащие вопросы, называются вершинами решений. Прямоугольники содержат цели диаграммы и означают логические выводы. Линии показывают направление диаграммы. Многие вершины имеют сразу по несколько ветвей, связывающих их с другими вершинами. Выбор выходящей из вершины ветви определяется проверкой условия, содержащегося в вершине.

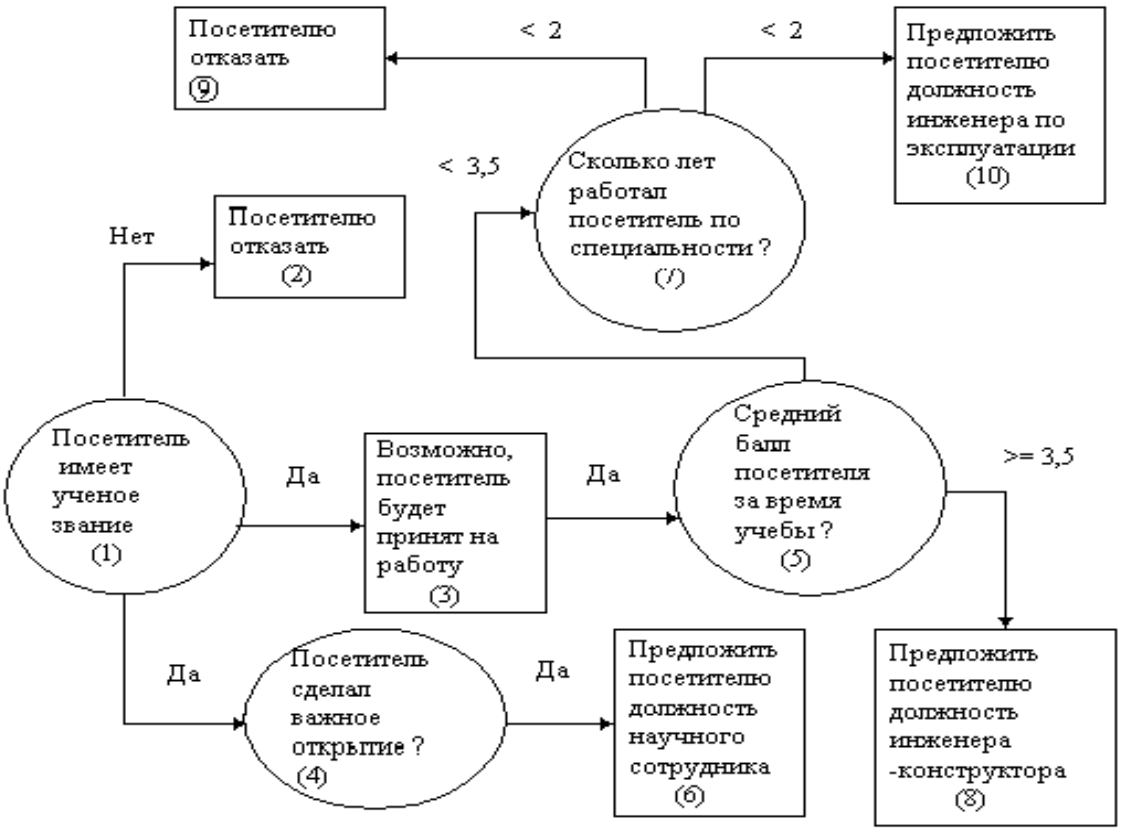


Рисунок 53 - Дерево решений для выбора должности

Например, вершина 5 (см. рис.53) содержит вопрос, на который есть два возможных ответа, и поэтому у неё два пути в зависимости от среднего балла посетителя за время учёбы, то есть возможен выбор одной из двух ветвей. Если средний балл равен 3.1, то будет выбран первый путь, так как 3.1 меньше 3.5. В программе под средний балл сначала отводится переменная, а затем ей присваивается значение. Можно сказать, что вершины содержат переменные, а пути - это условия, в соответствии с которыми переменным присваиваются значения. После того, как для проблемной области сформулированы правила,

эти условия становятся условными частями (ЕСЛИ) правила. Прямоугольники содержат частные или общие выводы. Например, прямоугольник на рис.8 может содержать ответ на вопрос, будет ли посетителю предложена работа. Общая цель системы, в которой реализованы обратные рассуждения, - получить окончательный ответ. Локальной целью может быть содержащийся в прямоугольнике на рис.8 ответ на вопрос, будет ли посетителю предложена должность. Однако эта вершина имеет и исходящие ветви, и, следовательно, через неё может проходить путь к следующему логическому выводу. В последнем случае, поскольку исходящая ветвь не содержит условия, и она только одна, говорят, что вершина содержит локальный вывод для другой цели. Локальный вывод - это также составляющая условной части правила.

Обобщённый алгоритм работы системы с обратной цепочкой выводов.

Система, реализующая обратную цепочку рассуждений, должна выполнять следующие шаги :

1. Определить переменную логического вывода.
2. В списке логических выводов искать первое вхождение этой переменной. Если переменная найдена, в стек логических выводов поместить номер соответствующего правила и установить номер условия равным 1. Если переменная не найдена, сообщить пользователю, что ответ найти невозможно.
3. Присвоить значения всем переменным условия из данного правила.
4. Если в списке переменных указано, что какой-либо переменной условия не присвоено значение и её нет среди переменных логического вывода (её нет в списке логических выводов), запросить её значение у пользователя.
5. Если какая-либо переменная условия входит в переменные логического вывода, поместить в стек номер правила, в логический вывод которого она входит, и вернуться к шагу 3.
6. Если из правила нельзя определить значение переменной, удалить соответствующий ему элемент из стека и в списке логических выводов продолжить поиск правила с этой переменной логического вывода.
7. Если такое правило найдено, перейти к шагу 3.
8. Если переменная не найдена ни в одном из оставшихся правил в логическом выводе, правило для предыдущего вывода не верно. Если предыдущего вывода не существует, сообщить пользователю, что ответ получить невозможно. Если предыдущий вывод существует, вернуться к шагу 6.
9. Определить значение переменной из правила, расположенного в начале стека; правило из стека удалить. Если есть ещё переменные логического вывода, увеличить значение номера условия и для проверки оставшихся

переменных вернуться к шагу 3. Если больше нет переменных логического вывода, сообщить пользователю окончательный вывод.

2. Порядок выполнения работы*:*

1. Выбрать по заданной теме 15-20 правил принятия решения.
2. Упорядочить их по степени важности.
3. Построить дерево принятия решения.
4. Построить список переменных логических выводов.

3. Варианты заданий

Реализовать прямую цепочку рассуждений для следующих задач:

1) диагностика неисправностей электронной аппаратуры,
2) диагностика неисправностей автомобиля,
3) диагностика заболеваний (по выбору),
4) Анализ объекта (по выбору)
    a. спортивных мероприятий
    b. телепередач
    c. природных катаклизмов

   и т.п.
5) задачи информационно-советующего характера (по выбору)
    a. помощник заведующего складом
    b. помощник аптекаря
    c. помощник оператора справочной службы
    d. выбор должности
    e. проведение отпуска

   и т.п.

4. Контрольные вопросы

1. Чем отличаются «прямая» и «обратная» цепочки рассуждений?
2. Какие виды правил существуют?
3. Как контролируется вывод правил из БЗ?
4. Как учитывается достоверность заключительной части правила?

Лабораторная работа № 6. Разработка экспертной системы

1 Цель работы

Целью работы является изучение принципов программирования простейших экспертных систем.

2 Краткие теоретические сведения

2.1 Структура экспертной системы

Под **экспертной системой** (ЭС) понимают набор программ, выполняющий функции эксперта при решении задач из некоторой предметной области. ЭС выдают советы, проводят анализ, дают консультации, ставят диагноз.

Главным достоинством ЭС, определяющим сравнительно высокий интерес к ним как к методам искусственного интеллекта, является возможность накопления знаний и сохранение их длительное время. В отличии от человека к любой информации ЭС подходят объективно, что улучшает качество проводимой экспертизы. При решении задач, требующих обработки большого объема знаний, возможность возникновения ошибки при переборе очень мала.

Структура традиционной **статической** ЭС включает следующие основные компоненты /1/ (рисунок 54):

- решатель (интерпретатор),
- рабочую память,
- базу знаний,
- компонент приобретения знаний,
- объяснительный компонент,
- диалоговый компонент.

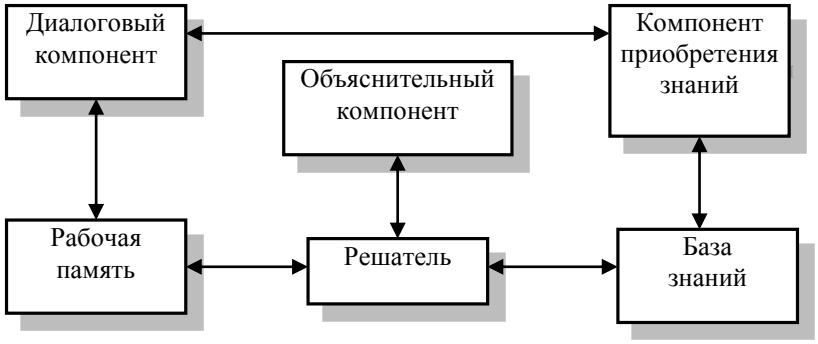


Рисунок 54 – Структура экспертной системы

В **базе знаний** содержатся факты, на основе которых производится выработка решения. **Решатель** – алгоритм, программа, набор правил, по которым осуществляется решение задачи. Процесс рассуждений реализуется на основе базы знаний и рабочей памяти. Решатель выполняет две функции: во-первых, просмотр существующих фактов из рабочей памяти и правил из базы знаний и добавление (по мере возможности) в рабочую память новых фактов и, во-вторых, определение порядка просмотра и применения правил. Одним из распространенных алгоритмов решателя является **байесовский алгоритм.** На **диалоговый компонент** возложена задача ведения диалога о решаемой задаче на языке пользователя (эксперта). **Компонент приобретения знаний** как программный модуль может в ЭС отсутствовать. Его задача – приобретать в ходе диалога новые знания. Наличие **объяснительного компонента** дает ЭС способность при решении задачи следовать линии рассуждений, понятной пользователю (эксперту), и объяснять ход рассуждений.

Система функционирует в следующем циклическом режиме:

1 Диалоговый компонент запрашивает данные или результатов анализов, наблюдений (этот этап может быть реализован в виде системы вопросов к пользователю) и помещает их в рабочую память.
2 Решатель интерпретирует результаты с помощью правил, извлеченных из базы знаний.
3 В случае нехватки информации для окончательного решения процесс продолжается до тех пор, пока не поступит достаточное количество информации.

В любой момент времени в системе существуют три типа знаний:

- статические знания о предметной области, после того как эти знания выявлены, они уже не изменяются;
- динамические знания о предметной области, они обновляются по мере выявления новой информации;
- рабочие знания - знания, применяемые для решения конкретной задачи или проведения консультации.

Все перечисленные выше знания хранятся в базе знаний. Для ее построения требуется провести опрос специалистов, являющихся экспертами в конкретной предметной области, а затем систематизировать, организовать и снабдить эти знания указателями, чтобы впоследствии их можно было легко извлечь из базы знаний.

Динамическая ЭС отличается от статической наличием двух дополнительных компонентов [1]:

- подсистемы моделирования внешнего мира;
- подсистемы связи с внешним окружением, осуществляющей связь с внешним миром посредством датчиков и контроллеров.

2.2 Продукционные системы

База знаний - наиболее важная компонента экспертной системы, на которой основаны ее «интеллектуальные способности». Существует несколько способов представления знаний в ЭС, однако общим для всех них является то, что знания представлены в символьной форме (элементарными компонентами представления знаний являются тексты, списки и другие символьные структуры). Тем самым в ЭС реализуется принцип символьной природы рассуждений, который заключается в том, что процесс рассуждения представляется как последовательность символьных преобразований.

Наиболее распространенный способ представления знаний - в виде конкретных **фактов** и **правил**, по которым из имеющихся фактов могут быть выведены новые. Факты представлены, например, в виде троек:

(АТРИБУТ ОБЪЕКТ ЗНАЧЕНИЕ).

Такой факт означает, что заданный объект имеет заданный атрибут (свойства) с заданным значением. В более простых случаях факт выражается неконкретным значением атрибута, а каким-либо простым *утверждением*, которое может быть *истинным* или *ложным*.

Наиболее простым с точки зрения построения и широко используемым типом моделей принятия решений являются **продукционные системы** (ПС). Они представляют собой структурированные наборы **продукционных правил** (ПП или PR) вида

$$PR = <S, N, F, A \Rightarrow C, W>,$$

где S - сфера применения данного правила; N - номер или имя правила; F - предусловие применения (условие активизации), содержащее информацию об истинности и приоритетности данного правила; $A \Rightarrow C$ - ядро ПП; W - постусловие.

Сфера применения S обозначает принадлежность ПП какому-либо определенному этапу функционирования ПС или состоянию процесса принятия решения.

В состав правил могут входить условия активизации F, которые представляют собой либо переменную, либо логическое выражение (предикат). Когда F принимает значение «истина», ядро продукции может быть активизировано. Если F «ложно», то ядро не активизируется.

Постусловие W описывает, какие изменения следует внести в ПС, и актуализируется только после того, как ядро продукции реализовалось.

Интерпретация ядра может быть различной в зависимости от вида A и C, находящихся по разные стороны знака секвенции «⇒».Наиболее часто в ПС используют ПП вида

«если A, то C»,

где A и C - логические выражения, которые могут включать в себя другие выражения; A называется **антецедентом**, C - **консеквентом.**

Прежде всего, все ядра делятся на два типа: **детерминированные** и **недетерминированные**. В детерминированных ядрах при актуализации ядра и при выполнимости A правая часть ядра выполняется обязательно («если A, то C»); в недетерминированных ядрах C может выполняться с определенной вероятностью. Недетерминированное ядро может выглядеть так:

«если A, то возможно C».

Возможность может определяться некоторыми оценками реализации ядра. Например, если задана вероятность выполнения C при актуализации A, то ПП может быть таким:

«если A, то с вероятностью P выполнить C».

Оценка реализации ядра может быть лингвистической, связанной с лингвистической переменной:

«если A, то с большей долей уверенности возможно C».

К недетерминированным ПП относятся т.н. прогнозирующие ПП, в которых описываются, например, последствия, ожидаемые при актуализации A:

«если A, то с вероятностью P можно ожидать C».

Таким образом, секвенция «⇒» в детерминированных ядрах реализуется с необходимостью, а в недетерминированных - с возможностью.

Детерминированные ПП могут быть **однозначными** и **альтернативными**. Во втором случае в правой части ядра указываются

альтернативные возможности выбора, которые оцениваются специальными весами выбора. В качестве таких весов могут использоваться вероятностные, лингвистические, экспертные и прочие оценки.

ПП могут быть доопределены логическими выражениями, определяющими инициируемые процедуры, которые имеют место в случае отсутствия ее активности:

«если А, то C_1, иначе C_2».

Продукционные правила, учитывают накладываемые ограничения, а также показатели эффективности, по которым определяются управляющие воздействия и которые часто являются неизмеримыми лингвистическими переменными.

Достоинствами ПС являются:

- удобство описания процесса принятия решения экспертом (формализация его интуиции и опыта);
- простота редактирования модели;
- прозрачность структуры.

ПС в качестве моделей применимы в следующих случаях:

- не могут быть построены строгие алгоритмы или процедуры принятия решений, но существуют эвристические методы решения;
- существует, по крайней мере, один эксперт, который способен явно сформулировать свои знания и объяснить свои методы применения этих знаний при принятии решения;
- пространство возможных решений относительно невелико (число решений счетно);
- задачи решаются методом формальных рассуждений;
- данные и знания надежны и не изменяются со временем.

3 Экспериментальная часть

3.1 Задания, порядок выполнения работы и содержание отчета

Для работы используется расширение **Fuzzy Logic Toolbox** пакета **MATLAB.**

Общее задание на лабораторную работу:

Разработать экспертную систему оценки знаний экзаменуемого в соответствии с заданием. Для работы использовать расширение **Fuzzy Logic Toolbox** пакета **MATLAB** (описание приводится).

Содержание отчета по этапам работы:

- задание,
- продукционные правила разработанной экспертной системы,
- тип и интервалы определенности функций принадлежности входных и выходной переменной;
- протоколы проверки работоспособности на примерах.

Задание

Разработать экспертную систему оценки знаний по результатам ответа на экзаменационный билет. Билет содержит два теоретических вопроса и одно

практическое задание. При неудовлетворительной оценке за практическое задание общая оценка за экзамен признается неудовлетворительной. Знания по каждому вопросу билета и на весь билет оцениваются по 4-бальной шкале – неудовлетворительно, удовлетворительно, хорошо, отлично.

3.2 Продукционная экспертная система в среде MATLAB

Перед началом разработки экспертной системы с использованием инструментальных средств **MATLAB** необходимо разработать продукционную систему (продукционные правила), которые будут использоваться при выводе. Правила имеют следующую структуру:

ЕСЛИ (1 воп. – хор) и (2 воп. – хор) и (3 воп. –хор), ТО (оцен. – хор.);

ЕСЛИ (1 воп.–хор) и (2 воп. –хор) и (3 воп.–неуд.), ТО (оцен. – неуд.);

ЕСЛИ (1 воп.–отл.) и (2 воп.–отл.) и (3 воп.–уд.), ТО (оцен. – хор.);

и т. д.

Вывод каждого правила определяется субъективными представлениями эксперта.

Разрабатываемая экспертная система оценки ответа на экзаменационный билет в среде расширения **Fuzzy Logic Toolbox** пакета **MATLAB** использует в своем выводе элементы теории нечетких множеств.

Редактор **Fuzzy Logic Toolbox** запускается командой **fuzzy** из командной строки пакета **MATLAB.** Новая система с выводами по алгоритму Мамдани задается командой *File/New FIS/Mamdani*. Данный алгоритм описывает несколько последовательно выполняющихся этапов. При этом каждый последующий этап получает на вход значения, полученные на предыдущем шаге.

По заданию оценивается билет из трех вопросов, поэтому на входе системы должны быть заданы три входа. Входы добавляются последовательно (*Edit/Add Variable…/Input*) после конфигурирования каждого предыдущего входа. Конфигурирование заключается в определении функции степени принадлежности, назначении интервалов определенности и названия.

В рассматриваемой задаче диапазон изменения входной переменной (поле *Range*, окно *Member Function Editior*) назначается от 0 до 5. Всего задаются 4 функции принадлежности. Каждой функции определяется название – n (неуд.), ud (удовл.), hor (хор.), ot (отл.). Тип функции принадлежности – трапециевидная (trapmf), с границами, например, как на рисунке 10. (можно задаться другим видом функции степени принадлежности).

Аналогично определяется выходная переменная. Вход в режим редактирования входной переменной производится двойным нажатием левой клавиши мыши. Добавление функции принадлежности – *Edit/Add MFs…*

Для наполнения базы знаний входят в режим ввода правил (*Edit/Rules*). Разработанные продукционные правила последовательно вводятся (*Add rule*) в систему вывода (рис.55).

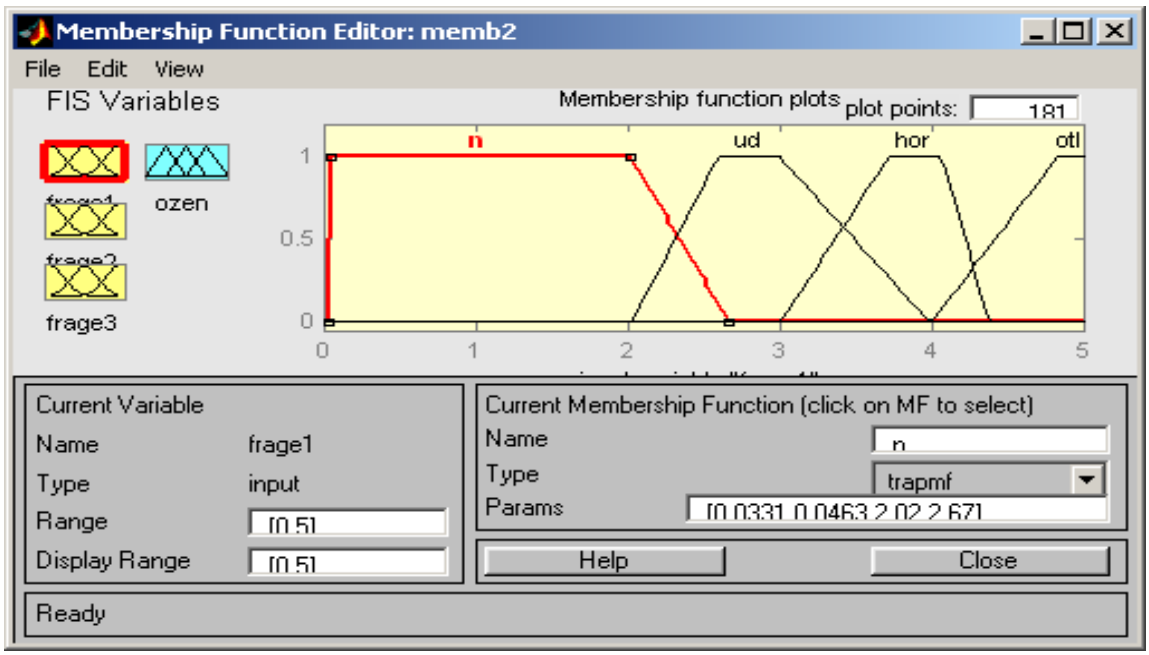


Рисунок 55 – Формы функций степени принадлежности

Для работы с созданной экспертной системой нужно из окна редактора правил (*Rule Editor*) перейти в окно вывода (*Rule Viewer*) *View/Rules*. Входными переменными является оценка за ответ каждого вопроса по 5-бальной шкале. Значения входных переменных вводятся в поле *Input*. Общая оценка ответа согласно вывода выводится в верхней части экране (синим цветом).

Варианты заданий

Вариант 1. Идентификация типа транспортного средства (велосипед, мотоцикл, мотороллер, телега, карета, автобус, грузовик, легковые: пикап, седан, хэтчбек, кабриолет…).

Вариант 2. Проведение летнего отдыха (дома, в саду, в пешем походе, в местном санатории, на Черном море, на Средиземном море, в круизе на теплоходе, на горном курорте, в африканских странах и т.д.).

Вариант 3. Выбор принтера (или к.-л. другой техники по выбору) для покупки (матричного, струйного, лазерного).

Вариант 4. Где поужинать вечером? (дома, у друзей, в столовой, в кафе, в ресторане, в клубе).

Вариант 5. Выбор телевизора для дома (диагональ, тип, цена, марка и т.д.).

Вариант 6. Покупка квартиры в г. Уфе (цена, площадь, престижность района, экологическая ситуация в районе, транспорт, тип дома и т.д.).

Вариант 7. Идентификация заглавных букв греческого алфавита.

Вариант 8. Идентификация садовых растений (огурцы, томаты, лук, яблоня, вишня, смородина, крыжовник и т.д.).

4 Контрольные вопросы

1 Что такое экспертная система?
2 Как функционирует экспертная система? Какие функции выполняет каждый элемент системы?
3 К какому типу относятся ядра продукций в разработанной Вами ЭС?
4 Приведите пример одного продукционного правила, соответствующего разработанной базе знаний.
5 Как осуществляется приобретение знаний в разработанной ЭС?
6 В чем отличие однозначных и альтернативных продукционных правил?

Лабораторная работа №7. Реализация экспертной системы

Цель работы: Целью работы является изучение принципов реализации простейших экспертных систем с использованием MS Excel.

1. Задание

В таблице приведены характеристики и их весовые факторы, расставленные экспертом (табл.9). Пользователи экспертной системы в зависимости от имеющихся у них ресурсов хотят получить совет по выпуску продукции.

Требуется разработать экспертную систему по автоматизации принятия решений. Приведенные правила помогут сформировать дерево решений.

Таблица 9– Характеристики и весовые факторы

№ п/п	Характеристика	Порядковый № характеристики атрибута	Атрибут	Весовой фактор атрибута (ВФ)
1.	Развитие транспортной сети предприятия	1.1 1.2 1.3	Хорошее Среднее Плохое	40 30 10
2.	Наличие сырья для выпуска изделия в районе изготовления	2.1 2.2 2.3	Много Немного Отсутствует	30 20 5
3.	Наличие сырья в ближайших районах	3.1 3.2 3.3	Много Немного Отсутствует	20 10 5
4.	Наличие сети сбыта изделия	4.1 4.2 4.3	Развитая Не развитая Слабо развитая	30 20 10

Правила вывода:

1. Если суммарный весовой фактор меньше 70, то принято решение "Нет смысла в выпуске изделия".

2. Если суммарный весовой фактор выше 70, но меньше 90, то решение "Можно наладить выпуск небольшой партии изделия ".

3. Если суммарный весовой фактор выше или равен 90 и фактор п.4 меньше или равен 20, то "Имеет смысл вложить средства в развитие сети сбыта".

4. Если суммарный весовой фактор выше 90, то "Имеет смысл наладить выпуск крупной партии изделия".

2. Дерево принятия решений

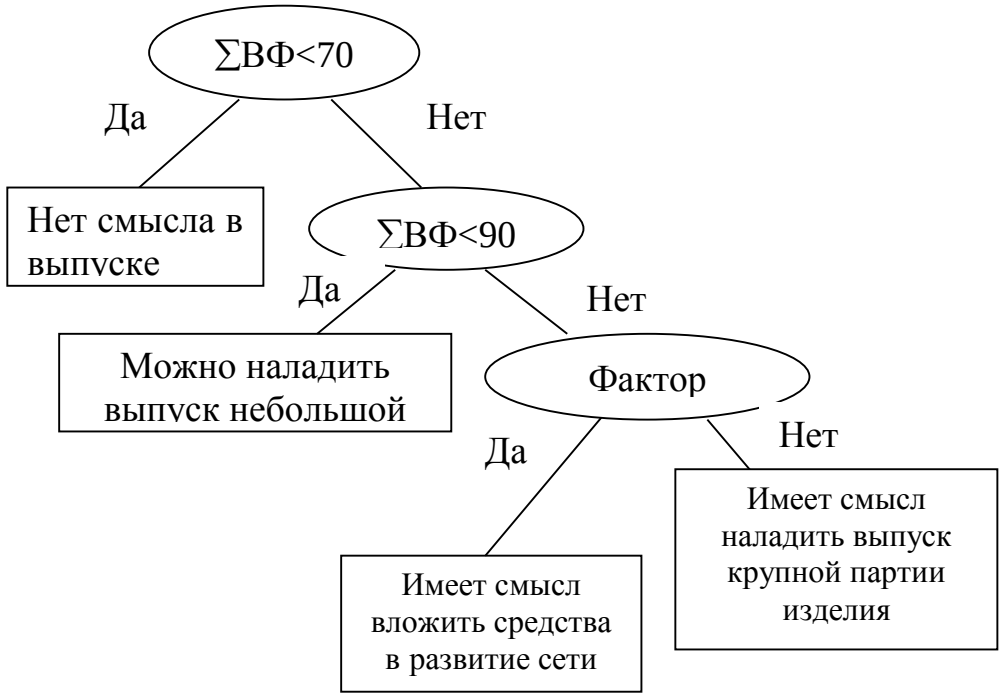


Рисунок 56 – Дерево принятия решений

2. База данных, заполненная одним из пользователей экспертной системы

Таблица 10 – База данных, заполненная одним из пользователей экспертной системы

№	Характеристика	Порядковый № характеристики атрибута	Атрибут	Весовой фактор атрибута
1	Развитие транспортной сети предприятия	1.1 1.2 1.3	Хорошее	40
2	Наличие сырья для выпуска изделия в районе изготовления	2.1 2.2 2.3	Немного	20

3	Наличие сырья в ближайших районах	3.1 3.2 3.3	Много	20
4	Наличие сети сбыта изделия	4.1 4.2 4.3	Развитая	30
5				∑ВФ=110

4. База знаний

Таблица 11 – База знаний

№ п/п	Характеристика	Порядковый № характеристики атрибута	Атрибут	Весовой фактор атрибута
1.	Развитие транспортной сети предприятия	1.1 1.2 1.3	Хорошее Среднее Плохое	40 30 10
2.	Наличие сырья для выпуска изделия в районе изготовления	2.1 2.2 2.3	Много Немного Отсутствует	30 20 5
3.	Наличие сырья в ближайших районах	3.1 3.2 3.3	Много Немного Отсутствует	20 10 5
4.	Наличие сети сбыта изделия	4.1 4.2 4.3	Развитая Не развитая Слабо развитая	30 20 10

5. Правила, формирующие принятие решений

1. Если суммарный весовой фактор меньше 70, то принято решение "Нет смысла в выпуске изделия".

2. Если суммарный весовой фактор выше 70, но меньше 90, то решение "Можно наладить выпуск небольшой партии изделия ".

3. Если суммарный весовой фактор выше или равен 90 и фактор пункта 4 меньше или равен 20, то "Имеет смысл вложить средства в развитие сети сбыта".

4. Если суммарный весовой фактор выше 90, то "Имеет смысл наладить выпуск крупной партии изделия".

6. Экспертная система в режиме формул – с показом формул, которые были использованы при разработке экспертной системы

31		Наличие сырья в ближайших районах		Много	1						
32				Немного	2	1					
33				Отсутствует	3						
34											
35		Наличие сети сбыта изделия		Развитая	1						
36				Не развитая	2	1					
37				Слабо развитая	3						
38											
39							Обработка ответов				
40											
41						=ЕСЛИ(G23=1;40;ЕСЛИ(G23=2;30;ЕСЛИ(G23=3;10)))						
42						=ЕСЛИ(G27=1;30;ЕСЛИ(G27=2;20;ЕСЛИ(G27=3;5)))						
43						=ЕСЛИ(G31=1;20;ЕСЛИ(G31=2;10;ЕСЛИ(G31=3;5)))						
44						=ЕСЛИ(G35=1;30;ЕСЛИ(G35=2;20;ЕСЛИ(G35=3;10)))						
45		Суммарный весовой фактор равен				=СУММ(G41:G44)						
46											
47		Принятие решения									
48	=ЕСЛИ(G45<70;"Нет смысла в выпуске изделий";ЕСЛИ(И(G45>70;G45<90);"Можно наладить выпуск небольшой партии изделий";ЕСЛИ(И(ИЛИ(G45=90;G45>90);ИЛИ(G44=20;G44<20));"Имеет смысл вложить средства в развитие сети сбыта";"Имеет смысл наладить выпуск крупной партии												

Рисунок 57 - Экспертная система в режиме формул

5. ***Экспертная система в режиме пользователя***, которая должна содержать только вопросы для пользователя и рекомендуемое решение. Вся остальная информация, связанная с анализом ответов, вычислениями и т.п. должна быть скрыта (рис.58,59).

	A	B	C	D	E	F	G	H	I	J	K
1					Экспертная система по автоматизации принятия решений						
2											
3								База знаний			
4											
5		Развитие транспортной сети предприятия			Хорошее	40					
6					Среднее	30					
7					Плохое	10					
8											
9		Наличие сырья для выпуска изделия в районе изготовления			Много	30					
10					Немного	20					
11					Отсутствует	5					
12											
13		Наличие сырья в ближайших районах			Много	20					
14					Немного	10					
15					Отсутствует	5					
16											
17		Наличие сети сбыта изделия			Развитая	30					
18					Не развитая	20					
19					Слабо развитая	10					
20											

Рисунок 58 - Экспертная система в режиме пользователя

19			Слабо развитая	10		
20						
21						База данных
22						
23		Развитие транспортной сети предприятия	Хорошее	1		
24			Среднее	2	1	
25			Плохое	3		
26						
27		Наличие сырья для выпуска изделия в районе изготовления	Много	1		
28			Немного	2	2	
29			Отсутствует	3		
30						
31		Наличие сырья в ближайших районах	Много	1		
32			Немного	2	1	
33			Отсутствует	3		
34						
35		Наличие сети сбыта изделия	Развитая	1		
36			Не развитая	2	1	
37			Слабо развитая	3		
38						
39						Обработка ответов
40						
41					40	
42					20	
43					20	
44					30	
45		Суммарный весовой фактор равен			110	
46						
47		Принятие решения				
48	Имеет смысл наладить выпуск крупной партии изделий					

Рисунок 59 - Экспертная система в режиме пользователя

Варианты заданий

Вариант №1

В таблице приведены характеристики и их весовые факторы, расставленные экспертом. Пользователи экспертной системы в зависимости от имеющихся у них оценок хотят получить совет по возможности получения стипендии.

Требуется разработать экспертную систему по автоматизации принятия решений. Приведенные правила помогут сформировать дерево решений (табл.12).

Таблица 12 – Правила и весовые факторы

№ п/п	Характеристика	Порядковый № характеристики атрибута	Атрибут	Весовой фактор атрибута
1.	Оценки	1.1 1.2 1.3	Все «5» «4» и «5» Имеются «3»	40 30 10
2.	Сдача всех экзаменов и зачетов своевременно	2.1 2.2 2.3	В срок; С задержкой на неделю; После окончания сессии.	30 20 5
3.	Посещаемость	3.1 3.2 3.3	Хорошая Средняя Плохая	20 10 5
4.	Участие в олимпиадах, конференциях и т.п.	4.1 4.2 4.3	Активное Среднее Пассивное	30 20 10

Правила вывода:

1. Если суммарный весовой фактор меньше 70, то принято решение "Нет стипендии".
2. Если суммарный весовой фактор выше 70, но меньше 90 и 1 фактор больше 10, то решение "Обычная стипендия ".
3. Если суммарный весовой фактор выше или равен 90, то "Повышенная стипендия".

Вариант №2

В таблице приведены характеристики и их весовые факторы, расставленные экспертом. Требуется разработать экспертную систему по распределению баллов для членов жюри конкурса «Мисс Академия».

Приведенные правила помогут сформировать дерево решений (табл.13).

Таблица13 – Правила и весовые факторы

№ п/п	Характеристика	Порядковый № характеристики атрибута	Атрибут	Весовой фактор атрибута
1.	Ум, эрудиция	1.1 1.2 1.3	Высокий уровень Средний уровень Низкий уровень	40 30 10

2.	Красота, привлекательность	2.1 2.2 2.3	Очень красивая Хорошенькая Среднестатистическая	30 20 5
3.	Таланты	3.1 3.2 3.3	Более 3-х 2 1	20 10 5
4.	Харизма, подача	4.1 4.2 4.3	Отличная Частично присутствует Отсутствует	30 20 10

Правила вывода:

1. Если суммарный весовой фактор меньше 70, то принято решение "Нет места".
2. Если суммарный весовой фактор выше 70, но меньше 80, то решение "3 место ".
3. Если суммарный весовой фактор выше 80 и меньше 90, то "2 место".
4. Если суммарный весовой фактор выше или равен 90 и 1 фактор больше 10, то "1 место".

Вариант №3

В таблице приведены характеристики и их весовые факторы, расставленные экспертом. Требуется разработать экспертную систему по распределению баллов для членов жюри конкурса «Мистер Академия».

Приведенные правила помогут сформировать дерево решений (табл14).

Таблица 14– Правила и весовые факторы

№ п/п	Характеристика	Порядковый № характеристики атрибута	Атрибут	Весовой фактор атрибута
1.	Ум, эрудиция	1.1 1.2 1.3	Высокий уровень Средний уровень Низкий уровень	40 30 10
2.	Сила	2.1 2.2 2.3	Очень сильный Сильный Слабый	30 20 5
3.	Ловкость	3.1 3.2 3.3	Очень ловкий Ловкий Неловкий	20 10 5
4.	Харизма	4.1 4.2 4.3	Отличная Частично присутствует Отсутствует	30 20 10

Правила вывода:

1. Если суммарный весовой фактор меньше 70, то принято решение "Нет места".

2. Если суммарный весовой фактор выше 70, но меньше 80, то решение "3 место ".

3. Если суммарный весовой фактор выше 80 и меньше 90, то "2 место".

4. Если суммарный весовой фактор выше или равен 90 и 4 фактор больше 10, то "1 место".

Вариант №4

В таблице приведены характеристики и их весовые факторы, расставленные экспертом. Пользователи экспертной системы в зависимости от имеющихся у них баллов хотят получить совет по возможности получения гранта.

Требуется разработать экспертную систему по автоматизации принятия решений для выбора претендентов на получение гранта по НИОКР. Приведенные правила помогут сформировать дерево решений (табл.15).

Таблица 15 – Правила и весовые факторы

№ п/п	Характеристика	Порядковый № характеристики атрибута	Атрибут	Весовой фактор атрибута
1.	Новизна работы	1.1 1.2 1.3	Более 4-х пунктов 3 пункта 1 – 2 пункта	40 30 10
2.	Актуальность	2.1 2.2 2.3	Высокая Средняя Низкая	30 20 10
3.	Сроки выполнения работы	3.1 3.2 3.3	1 год 3 года 5 лет	20 10 5
4.	Направление	4.1 4.2 4.3	Техническое Гуманитарное	30 20

Правила вывода:

1. Если суммарный весовой фактор меньше 70, то принято решение "В гранте отказано".

2. Если суммарный весовой фактор выше 70, но меньше или равно 90 и 1 фактор больше или равно 10, то решение "3 место и 200 000 руб. ".

3. Если суммарный весовой фактор выше 90, но меньше 110 и 1 фактор больше или равно 30, то "2 место и 400 000 руб.".

4. Если суммарный весовой фактор выше 110, то "1 место и 600 000 руб.".

Лабораторная работа №8. Синтез управляющих систем на основе нечеткой логики

1 Цель работы

Изучение основных понятий теории нечетких множеств и их приложений. Знакомство с пакетом математических приложений MATLAB.

2 Задача работы

- Аппроксимация функции правилами нечеткого вывода.

- Исследование системы автоматического регулирования с нечетким регулятором.

3 Краткие теоретические сведения

3.1 Понятие нечеткого множества

Значительный шаг в направлении развития теории нечетких множеств сделал профессор Калифорнийского университета Лотфи А. Заде (1965 г. - публикация работы “Fuzzy Sets”). Заде расширил понятие множества, допустил, что характеристическая функция (функция принадлежности элемента множеству) может принимать любые значения в интервале [0, 1]. Такие множества были названы им нечеткими (fuzzy).

Пусть E – универсальное множество, x – элемент E, P – некоторое свойство. Обычное (четкое) множество A универсального множества E, элементы которого удовлетворяют свойству P, определяются как множество упорядоченных пар

$$A = \{\mu_A(x)/x\},$$

где $\mu_A(x)$ - характеристическая функция, принимающая значение 1, если x удовлетворяет свойству P, и 0 – в противном случае.

В теории нечетких множеств для элементов x из E нет однозначного ответа «да/нет» относительно свойства P. В связи с этим нечеткое множество A универсального множества E определяется как множество упорядоченных пар с функцией принадлежности $\mu_A(x)$, принимающей значение в некотором упорядоченном множестве M (например, M=[0, 1]).

Функция принадлежности указывает степень (или уровень) принадлежности элемента x подмножеству A. Множество M называют множеством принадлежностей. Если M={0, 1},то нечеткое подмножество A может рассматриваться как обычное или четкое множество.

Примеры записи нечеткого множества.

1 Пусть E ={x_1, x_2, x_3, x_4, x_5}; A – нечеткое множество, для которого $\mu_A(x_1)$=0,3; $\mu_A(x_2)$=0; $\mu_A(x_3)$=1; $\mu_A(x_4)$=0,6; $\mu_A(x_5)$=0,9.

Тогда A можно представить в виде

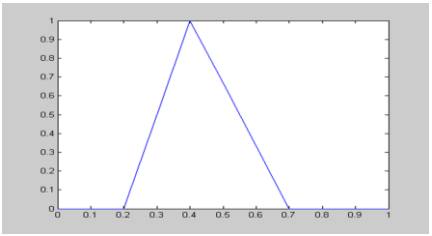
A = {0,3/x_1; 0/x_2; 1/x_3; 0,6/x_4; 0,9/x_5}.

2 Пусть E = {1, 2, 3, … 100} и соответствует понятию «возраст», тогда нечеткое множество «молодой» может быть определено с помощью выражения

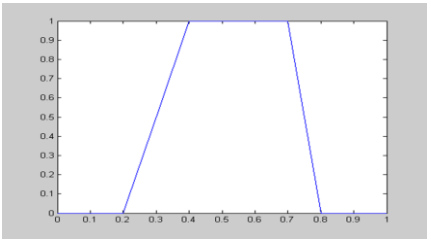
$$\mu_{"молодой"} = \begin{cases} 1, & x \in [1,25], \\ \dfrac{1}{1+\left(\dfrac{x-25}{5}\right)^2}, & x > 25. \end{cases}$$

В приведенных выше примерах использованы прямые методы определения функций принадлежности, когда эксперт задает значение $\mu_A(x)$ для каждого $x \subset E$. Такой способ используется для измеряемых понятий (скорость, температура и т.д.) или когда выделяют полярные значения. Разновидностью прямого способа задания функции принадлежности является групповой метод, когда группе экспертов предлагают сделать оценку того или иного явления, например, оценить: «этот человек лысый» или нет, - тогда количество утвердительных ответов, деленное на общее число экспертов, дает значение $\mu_{лысый}$ для данного лица.

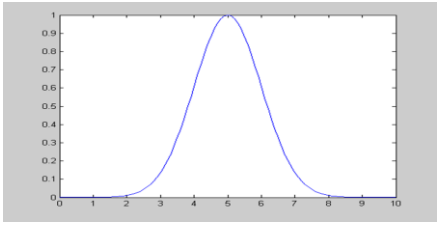
Кроме указанных способов задания функций принадлежности используют также типовые формы функций принадлежности (рисунок 60).



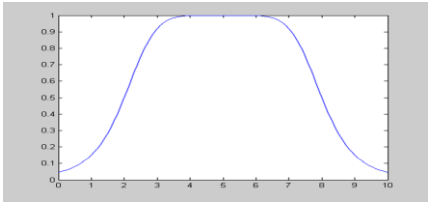
а) треугольная (trimf)



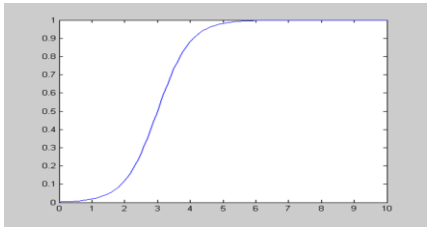
б) трапецеидальные (trapmf)



в) гауссова (gaussmf)



г) обобщенная колоколообразная (gbellmf)



д) сигмоидная (sigmf)

Рисунок 60 - Примеры типовых форм функций принадлежности (ниже в скобке приводится обозначение функции в среде MATLAB)

Аналитическая запись некоторых типовых функций принадлежности:

треугольная - $\mu(x)=\max\left(\min\left(\frac{x-a}{b-a},\frac{c-x}{c-b}\right),0\right)$, a,b,c - определяют параметры функции;

гауссова - $\mu(x)=e^{-\left(\frac{x-a}{b}\right)^2}$, a,b - параметры функции принадлежности;

сигмоидная - $\mu(x)=\frac{1}{1+\exp(-a(x-b))}$, a,b - параметры функции принадлежности.

3.2 Основные операции над нечеткими множествами

Включение

Пусть A и B нечеткие множества на универсальном множестве E. A содержится в B, если $\forall x\in E\ \mu_A(x)\le\mu_B(x)$. Обозначается $A\subset B$.

Равенство

A и B равны, если $\forall x\in E\ \mu_A(x)=\mu_B(x)$. Обозначение: $A=B$.

Дополнение

Пусть M=[0,1], A и B дополняют друг друга, если $\forall x\in E\ \mu_A(x)=1-\mu_B(x)$. Обозначается $B=\overline{A}$ или $A=\overline{B}$.

Пересечение

$A\cap B$ - наименьшее нечеткое подмножество, содержащееся одновременно в A и B: $\mu_{A\cap B}(x)=\min(\mu_A(x),\mu_B(x))=\mu_A(x)\,\tilde{\wedge}\,\mu_B(x)$.

Или, по другому, определение в классе треугольных норм (t – норма).

Типичными t - нормами являются:

- операция min как нечеткое логическое произведение с нечеткими переменными X1, X2: $\mathbf{X_1}\,\tilde{\wedge}\,\mathbf{X_2}=\mathbf{\min(X_1,X_2)}$;

- алгебраическое произведение $\mathbf{X_1}\,\tilde{\wedge}\,\mathbf{X_2}=\mathbf{X_1}\cdot\mathbf{X_2}$ [2, 3].

Объединение

$A\cup B$ - наибольшее нечеткое подмножество, включающее как A, так и B, с функцией принадлежности:

$\mu_{A\cup B}(x)=\max(\mu_A(x),\mu_B(x))=\mu_A(x)\,\tilde{\vee}\,\mu_B(x)$.

Или, по-другому, определение в классе треугольных конорм (s – норма).

Типичными s - нормами являются:

- операция нечеткого логического сложения $\mathbf{X_1}\,\tilde{\vee}\,\mathbf{X_2}=\mathbf{\max(X_1,X_2)}$;

- алгебраическая сумма $\mathbf{X_1}\tilde{\mathbf{V}}\mathbf{X_2}=\mathbf{X_1}+\mathbf{X_2}-\mathbf{X_1X_2}$,

где «~» означает нечеткую логическую операцию.

Разность

A-B= $A\cap\overline{B}$ с функцией принадлежности: $\mu_{A-B}(x)=\min(\mu_A(x),1-\mu_B(x))$.

Нечеткие отношения

Нечеткое n-мерное отношение определяется как нечеткое подмножество R на E, принимающее свои значения в M. В случае n=2 и M=[0,1] нечетким

отношением R между множествами X=E_1 и Y=E_2 будет называться функция R: (X, Y) → [0,1], которая ставит в соответствие каждой паре элементов (x,y) ∈ *X* x *Y* величину $\mu_R(x,y) \in [0,1]$.

Обозначение: $x \in X,\ y \in Y : xRy$.

Алгебраические операции над нечеткими отношениями аналогичны операциям с нечетким множествам.

Композиция (свертка) двух нечетких отношений

Пусть R_1 – нечеткое отношение R_1: $(X * Y) \rightarrow [0,1]$ между *X* и *Y*, и R_2 – нечеткое отношение R_2: $(Y * Z) \rightarrow [0,1]$ между *Y* и *Z*. Нечеткое отношение между *X* и *Z* обозначается $R_1 \bullet R_2$ и определяется как

$$\mu_{R_1 \bullet R_2}(x,y) = \bigvee_y \left[\mu_{R_1}(x,y) \wedge \mu_{R_2}(y,z)\right] = \max\left(\min\left(\mu_{R_1}(x,y), \mu_{R_2}(y,z)\right)\right),$$

где символ $\bigvee_y$ - обозначает операцию выбора наибольшего по *y* значения и называется (max-min)-сверткой отношений R_1 и R_2.

Пример. Пусть заданы отношения R_1 и R_2.

Таблица 1

R_1	y_1	y_2	y_3
x_1	0,1	0,7	0,4
x_2	1	0,5	0

Таблица 2

R_2	z_1	z_2	z_3	z_4
y_1	0,9	0	1	0,2
y_2	0,3	0,6	0	0,9
y_3	0,1	1	0	0,5

$$\mu_{R_1 \bullet R_2}(x_1, y_1) = \left[\mu_{R_1}(x_1, y_1) \wedge \mu_{R_2}(y_1, z_1)\right] \vee \left[\mu_{R_1}(x_1, y_2) \wedge \mu_{R_2}(y_2, z_1)\right] \vee$$
$$\vee \left[\mu_{R_1}(x_1, y_3) \wedge \mu_{R_2}(y_3, z_1)\right] =$$
$$= (0,1 \wedge 0,9) \vee (0,7 \wedge 0,3) \vee (0,4 \wedge 0,1) = 0,1 \vee 0,3 \vee 0,1 = 0,3.$$

Таблица 3

$R_1 \cdot R_2$	z_1	z_2	z_3	z_4
x_1	0,3	0,6	0,1	
x_2	0,9	0,5	1	

3.3 Нечеткие выводы

В экспертных и управляющих системах механизм нечетких выводов в своей основе имеет базу знаний, формируемую специалистами предметной области в виде совокупности нечетких предикатных правил вида:

$П_1$: если *x* есть A_1, то *y* есть B_1,

$П_2$: если *x* есть A_2, то *y* есть B_2,

…

$П_n$: если *x* есть A_n, то *y* есть B_n,

где x – входная переменная, y – переменная вывода, A и B – функции принадлежности, определенные на x и y соответственно.

Знания эксперта $A \rightarrow B$ отражает нечеткое причинное отношение предпосылки и заключения, поэтому его называют нечетким отношением:

$R = A \rightarrow B,$

где «$\rightarrow$» - нечеткая импликация.

Отношение R можно рассматривать как нечеткое подмножество прямого произведения $X \times Y$ полного множества предпосылок X и заключений Y. Таким образом, процесс получения (нечеткого) результата вывода B' с использованием данного наблюдения A' и значения $A \rightarrow B$ можно представить в виде

$B' = A' \bullet R = A' \bullet (A \rightarrow B).$

Алгоритм нечеткого вывода

1 ***Нечеткость*** (фаззификация, fuzzification). Функции принадлежности, определенные для входных переменных, применяются к их фактическим значениям для определения степени истинности каждой предпосылки каждого правила).

2 ***Логический вывод.*** Вычисленное значение истинности для предпосылок каждого правила применяется к заключениям каждого правила. Это приводит к одному нечеткому подмножеству, которое будет назначено переменной вывода для каждого правила. В качестве правил логического вывода используются только операции min (минимума) или prod (умножение).

3 ***Композиция***. Нечеткие подмножества, назначенные для каждой переменной вывода (во всех правилах), объединяются вместе, чтобы сформировать одно нечеткое подмножество для каждой переменной вывода. При подобном объединении обычно используются операции max (максимум) или sum (сумма).

4 ***Дефаззификация*** – приведение к четкости (defuzzification). Преобразование нечеткого набора выводов в число.

Алгоритмы нечеткого вывода Мамдани (Mamdani)

Пусть заданы два нечетких правила:

$П_1$: если x есть A_1 и y есть B_1, тогда z есть C_1,

$П_2$: если x есть A_2 и y есть B_2, тогда z есть C_2.

1 ***Нечеткость.*** Находят степени принадлежности для предпосылок каждого правила: $A_1(x_0)$, $A_2(x_0)$, $B_1(y_0)$, $B_2(y_0)$.

2 ***Нечеткий вывод.*** Определяют уровни «отсечения» для предпосылок каждого правила (операция min):

$\alpha_1 = A_1(x_0) \wedge B_1(y_0)$,

$\alpha_2 = A_2(x_0) \wedge B_2(y_0)$,

определяют усеченные функции принадлежности

$C'_1 = (\alpha_1 \wedge C_1(z))$,

$C'_2 = (\alpha_2 \wedge C_2(z))$.

3 ***Композиция***. Производится объединение найденных усеченных функций (операция max), получают нечеткое подмножество для переменной выхода с функцией принадлежности:

$$\mu_{\Sigma}(z) = C(z) = C'_1(z) \vee C'_2(z) = (\alpha_1 \wedge C_1(z)) \vee (\alpha_2 \wedge C_2(z)).$$

4 ***Дефаззификация***. Приведение к четкости (определение z_0), например, *центроидным* методом (как центр тяжести для кривой $\mu_{\Sigma}(z)$):

$$z_0 = \frac{\int_{\Omega} z\mu_{\Sigma}(z)dz}{\int_{\Omega} \mu_{\Sigma}(z)dz}.$$

Алгоритм иллюстрируется на рисунке 61.

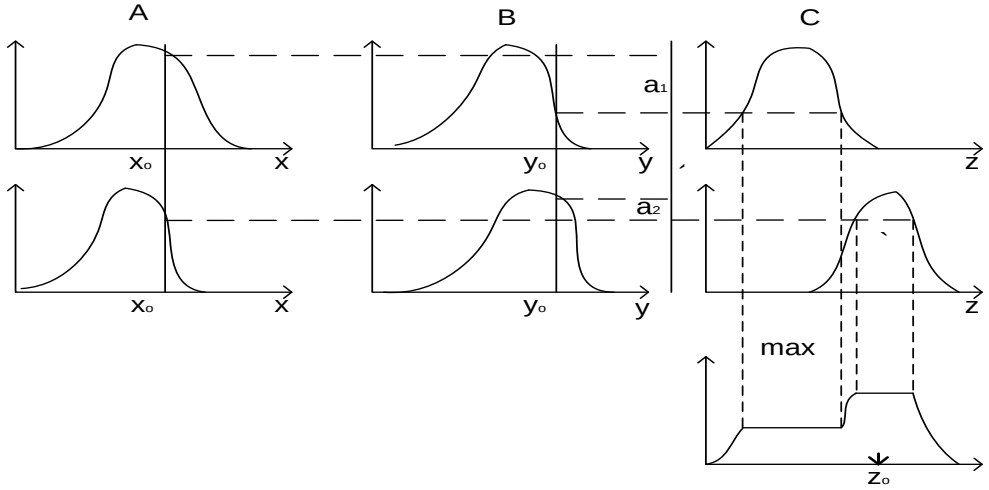


Рисунок 61 – Графическая интерпретация алгоритма Мамдани

Алгоритмы нечеткого вывода Сугено (Sugeno) и Такаги (Takagi)

Сугено (Sugeno) и Такаги (Takagi) использовали набор правил в следующей форме:

$П_1$: если x есть A_1 и y есть B_1, тогда $z=a_1x+b_1y$,

$П_2$: если x есть A_2 и y есть B_2, тогда $z=a_2x+b_2y$.

1 Первый этап – аналогично алгоритму Мамдани.

2 Определение $\alpha_1 = A_1(x_0) \wedge B_1(y_0)$, $\alpha_2 = A_2(x_0) \wedge B_2(y_0)$ и индивидуальные выходы правил:

$z^*_1 = a_1x_0 + b_1y_0$,

$z^*_2 = a_2x_0 + b_2y_0$.

3 Определяется значение переменной вывода:

$$z_0 = \frac{\alpha_1 z_1^* + \alpha_2 z_2^*}{\alpha_1 + \alpha_2}.$$

Представленная форма правил иллюстрирует алгоритм Сугено 1-го порядка. Если правила записаны в форме:

$П_1$: если x есть A_1 и y есть B_1, тогда $z=c_1$,

$П_2$: если x есть A_2 и y есть B_2, тогда $z=c_2$,

то задан алгоритм Сугено 0-го порядка.

Иллюстрация алгоритма Сугено 0-го порядка представлена на рисунке 62.

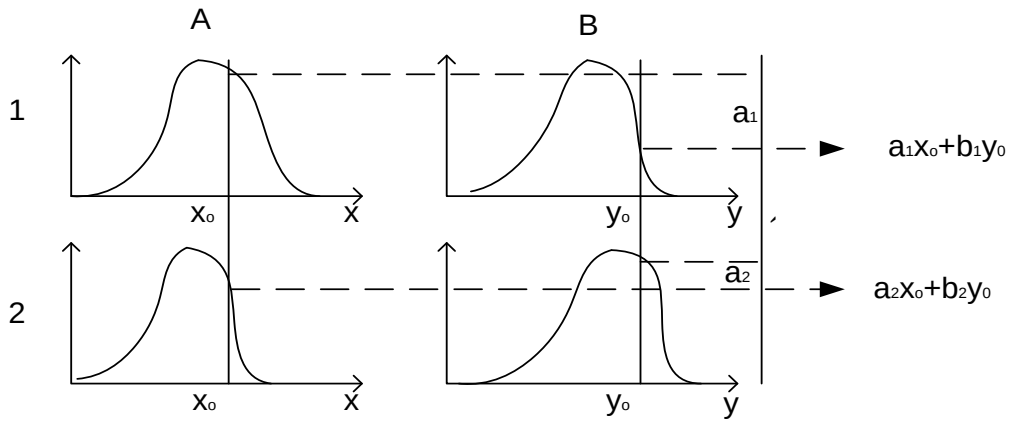


Рисунок 62 – Графическая интерпретация алгоритма Сугено

Перечислим наиболее известные методы дефаззификации.

Метод Максимума – выбирается тот элемент нечеткого множества, который имеет наивысшую степень принадлежности этому множеству.

Если этот элемент не является единственным, т.е. функция принадлежности $\mu_A(y_1)$ имеет несколько локальных максимумов, или если имеется максимальное «плато», то выбор среди элементов, имеющих наивысшую степень принадлежности множеству, осуществляется на основе некоторого критерия.

Метод левого (правого) максимума – выбирается наименьшее (наибольшее) из чисел y_1, y_2, .. ,y_n, имеющих наивысшую степень принадлежности нечеткому множеству.

Метод среднего из максимумов – в качестве искомого четкого значения y_o принимается среднее арифметическое координат локальных максимумов $y_o = \frac{1}{n}\sum_{1}^{n} y_i$.

Возможные методы дефаззификации и их обозначения в MATLAB приведены на рисунке 63: наименьший максимум (smallest of max, som), наибольший максимум (largest of max, lom), средний максимум (mean of max, mom), бисекторный (bisector of area), рассмотренный выше центроидный (centroid).

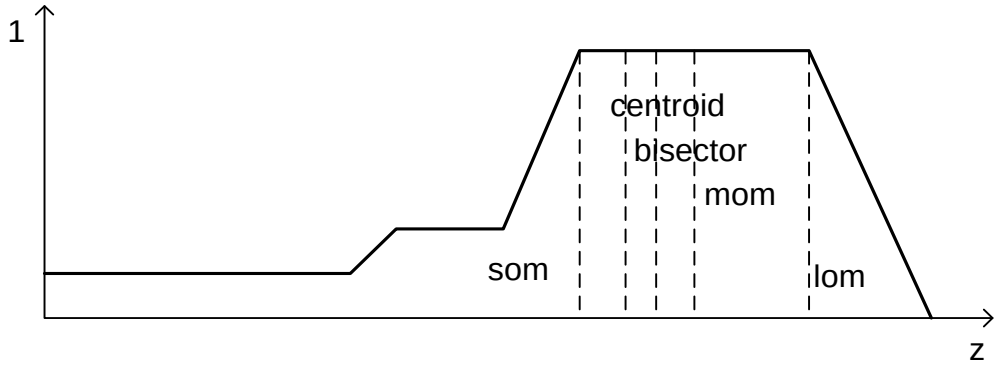


Рисунок 63 – Иллюстрация примеров фаззификации z

4 Проектирование системы нечеткого вывода в среде MATLAB

4.1 Построение нечеткой аппроксимирующей функции $y=x^2$

Исходные данные для аппроксимации представлены в таблице 16.

Таблица 16 – Значения x и y

X	-1	-0,6	0	0,4	1
Y	1	0,36	0	0,16	1

1 Командой **fuzzy** из режима командной строки запускается основная интерфейсная программа пакета Fuzzy Logic – редактор нечеткой системы вывода.

2 В меню **File** выбрать команду New Sugeno FIS.

3 Выбрать входной элемент системы *input1* и ввести в поле *Name* обозначение входной переменной (*x)*.

4 Войти в режим редактирования функции принадлежности – *Membership Function Editor* (двойное нажатие левой клавиши мыши). Выбором **Edit/Add MFs** (добавить функцию принадлежности) задать тип функции принадлежности (gaussmf) и количество (5).

5 Установить диапазон (*Range*) изменения *x* от -1 до 1 (определяется заданным диапазоном *x*).

6 Совместить ординаты максимумов функций принадлежности с значениями аргумента *x*.

Шаг выполняется двумя способами: выделить редактируемую функцию принадлежности, перетащить мышью кривую функции принадлежности; более точную установку проводят заданием числовых значений параметров функции принадлежности в поле *Params* (первое значение определяет размах кривой, второе – положение центра).

В поле *Name* вводится имя функции принадлежности (1- bn, 2 – n, 3 (центральная) – z, 4 – p, 5 – bp).

Выйти из редактора функций принадлежности – *Close*.

7 Ввести в поле название выходной переменной *output1* (*y)*, войти в режим редактирования функций принадлежности.

8 Задать вид функции принадлежности для выходной переменной. Предлагается выбрать в качестве функции принадлежности линейные (*linear*) или постоянные (*constant*) – в зависимости от алгоритма Сугено (1-го или 0-го порядка). В поставленной задаче необходимо выбрать постоянные функции принадлежности с общим числом 4 (по числу различающихся значений *y*). Нажать Ok.

9 Установить диапазон (*Range*) – [0, 1]. Изменить значения (*Params*) и задать для выходных переменных имена (*Name*) соответственно 0; 0,16; 0,36; 1. Закрыть редактор.

10 Перейти в редактор привил (*Rule Editor*) (дважды щелкнуть на средний белый квадрат разрабатываемой структуры системы нечеткого вывода). При вводе каждого правила необходимо обозначить соответствие между каждой функцией принадлежности аргумента *x* и числовым значением *y*. Кривая, обозначенная bn, соответствует *y=1*, для чего выбирается в левом поле (с заголовком *x is*) вариант bn, а в правом – 1 и нажимается кнопка *Add rule*.

Введенное правило появится в окне правил в виде

If (x is bn) then (y is 1) (1).

Аналогично вводятся все правила (всего 5).

Закрыть окно редактора правил и вернуться в окно FIS-редактора.

Построение системы закончено.

Сохранить на диске (**File/Save to disk as**) созданную систему. Перейти в редактор функций принадлежности (**View/Edit Membership function**). Из окна редактора командой можно перейти в окно просмотра правил (**View rules**), просмотра поверхности (**View surface**).

В окне просмотра правил иллюстрируется процесс принятия решения (вычисления *y*). Красная вертикальная черта, пересекающая графики в правой части окна, которую можно перемещать с помощью мыши, позволяет изменять значения переменной входа (либо вводят значение с клавиатуры в поле *Input*), при этом соответственно изменяется значение выхода.

Просмотр кривой *y(x)* осуществляется командой **View/View surface**. Большая погрешность аппроксимации заданной зависимости объясняется малым числом экспериментов.

С помощью рассмотренных редакторов на любом этапе проектирования нечеткой модели можно внести необходимые коррективы, например задание пользовательской функции принадлежности.

В рассмотренном примере использованы следующие операции, устанавливаемые по умолчанию в алгоритме Сугено:

- логический вывод организуется с помощью операции логического умножения (prod);

- композиция организуется с помощью операции логической суммы (вероятностного ИЛИ, probor);

- приведение к четкости (дефаззификация) организуется дискретным вариантом центроидного метода (взвешенным средним, wtaver).

4.2 Исследование автоматической системы управления с fuzzy-регулятором

Открыть файл *model2.mdl* каталога *Demo* из запущенного приложения Matlab. На экране должна быть изображена следующая структура АСР (рисунок 64).

Теперь при помощи инструментов графического интерфейса пользователя (GUI) пакета "Fuzzy Logic Toolbox" создадим нечёткую систему, реализующую типовой аналоговый ПИ-регулятор. Заметим, что с помощью пакета "Fuzzy Logic Toolbox" можно строить нечеткие системы двух типов - Мамдани и Сугэно. Остановимся на системе типа Мамдани. Командой fuzzy в окне MATLAB вызываем окно Редактора фази-инференционной системы (Fuzzy Inference System Editor), выбираем тип системы - Мамдани, задаём два входа - для пропорциональной и интегральной составляющих и называем входные переменные, например, x1 и x2, а выходную - y.

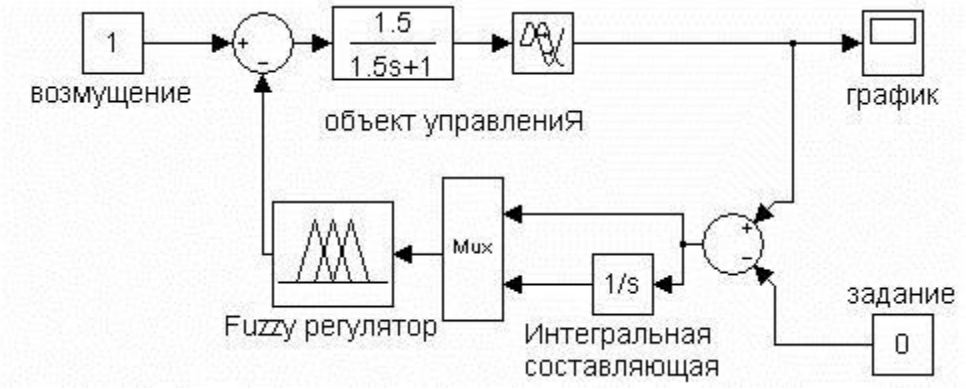


Рисунок 64 - Модель одноконтурной системы автоматического регулирования с ПИ-подобным fuzzy-регулятором

Из данного окна вызываем окно Редактора функций принадлежности (Membership Function Editor) двойным щелчком мыши по изображению переменной x1 или при помощи меню Edit. Здесь для лингвистического описания каждой переменной выберем семь треугольных термов (NB,NM,NS,ZE,PS,PM,PB). Термы выходной переменной лучше выбирать непересекающимися. Это повысит чёткость регулирования. В этом же окне зададим диапазоны изменения переменных:

Для входных переменных регулятора рекомендуются симметричные диапазоны изменения, при этом

$$x_1 \in \left[-\frac{1}{k_1};\frac{1}{k_1}\right] \text{ и } x_2 \in \left[-\frac{1}{k_0};\frac{1}{k_0}\right],$$

где $К_1$, $К_0$ – оптимальные настройки пропорциональной и интегрирующей частей ПИ – регулятора в смысле какого-либо критерия.

Для выходной переменной регулятора диапазон изменения рекомендуется брать в виде $y \in [0; C]$, где верхняя граница С при единичном

ступенчатом воздействии варьируется от 1.1 до 2, чтобы выходной сигнал регулятора мог компенсировать это возмущение. По мере увеличения значения C уменьшается динамическая ошибка, но возрастают время регулирования и число колебаний переходного процесса. Поэтому рекомендуется C принимать равным 2, когда наблюдается оптимальное соотношение между величиной динамической ошибки, времени регулирования и количеством колебаний.

Теперь необходимо сформировать базу правил fuzzy-регулятора. В основу положен способ, предложенный в литературе. Линейный непрерывный ПИ-регулятор с дифференциальным уравнением

$$y(t) = k_D \cdot \varepsilon(t) + \frac{1}{T_E}\int_0^t \varepsilon(t) \cdot dt$$

можно заменить близким по стратегии и логике управления fuzzy-регулятором, если в качестве его выходной переменной рассматривать приращение управляющего воздействия Δy. Тогда ПИ закон регулирования можно представить в следующей дифференциальной форме:

$$\frac{dy(t)}{dt} = k_D \cdot \frac{d\varepsilon(t)}{dt} + \frac{1}{T_E} \cdot \varepsilon(t),$$

или в разностной форме:

$$\Delta y(k) = y(k) - y(k-1) = k_D \cdot \Delta\varepsilon(k) + \frac{\Delta t}{T_E} \cdot \varepsilon(k).$$

Таким образом, для входных переменных ε(k) и Δε(k) и выходной Δy(k) может быть синтезирован fuzzy-регулятор, реализующий нелинейный закон

$$\Delta y(k) = F\left[\Delta\varepsilon(k), \varepsilon(k)\right]$$

и эквивалентный в определённом смысле ПИ-регулятору.

Для нашего случая x_1 соответствует сигналу рассогласования ε(k), x_2 соответствует приращению сигнала рассогласования Δε(k), а y соответствует Δy(k). Лингвистические правила для такого ПИ-подобного fuzzy-регулятора приведены в таблице 17.

Вызываем окно Редактора правил (Rule Editor) в меню Edit или нажатием Ctrl+3 и заполняем список правилами из таблицы 5. Правила формируются по типу: ЕСЛИ … И …, ТО…. Можно посмотреть пространство управления, вызвав окно Просмотра пространства управления(Surface Viewer) из меню View или комбинацией клавиш Ctrl+6. Полученный файл сохраним под именем **fuzzy1.fis.**

В окне параметров блока Fuzzy Logic Controller укажем имя файла **fuzzy1**. В окне модели в меню **File** выберем пункт **Model Properties**. В открывшемся окне выберем вкладку **CallBacks** и в поле **Model pre-load function** напишем:

```
fuzzy1=readfis('fuzzy1').
```

Данная команда будет каждый раз при открытии файла модели помещать файл **fuzzy1.fis** в WorkSpace (рабочее пространство системы MATLAB). Это необходимо для нормального функционирования модели.

Стоит заметить, что при внесении изменений в fis-файл нужно помещать его исправленную версию в WorkSpace либо при помощи пункта Export/To Workspace меню File, либо комбинацией клавиш Ctrl+T, либо каждый раз закрытием и открытием файла модели.

Таблица 17 – Лингвистические правила для ПИ fuzzy-регулятора

Δε \ ε	NB	NM	NS	ZE	PS	PM	PB
NB	NB	NB	NB	NB	NM	NS	ZE
NM	NB	NB	NB	NM	NS	ZE	PS
NS	NB	NB	NM	NS	ZE	PS	PM
ZE	NB	NM	NS	ZE	PS	PM	PB
PS	NM	NS	ZE	PS	PM	PB	PB
PM	NS	ZE	PS	PM	PB	PB	PB
PB	ZE	PS	PM	PB	PB	PB	PB

В диалоговом окне **Simulation Parameters** меню **Simulation** во вкладке **Advanced** для опции **Boolean logic signals** необходимо установить значение **off**. При этом блоки логики будут допускать переменные в форме с плавающей точкой.

Запуск модели – **Simulation/Start** (Ctrl+T).

В папке Demo находится файл с правилами нечеткого вывода. Загрузка файла осуществляется из окна редактора правил Rule Viewer: во вкладке **File/Import From Disk** открыть файл fuzzy1.fsi. Затем нужно поместить правила в WorkSpace системы (описано выше).

Работу разработанного fuzzy-регулятора можно сравнить с аналоговой системой регулирования (рисунок 65).

Дифференциальное уравнение ПИД-регулятора имеет вид:

$$y(t) = k_D \cdot \varepsilon(t) + \frac{1}{T_E}\int_0^t \varepsilon(t) \cdot dt + T_Д \cdot \frac{d\varepsilon(t)}{dt}.$$

Для исследования ПИ-регулятора в блоке настроек регулятора необходимо в поле параметра $T_Д$ (настройка дифференцирующей части ПИД-регулятора) установить значение 0.

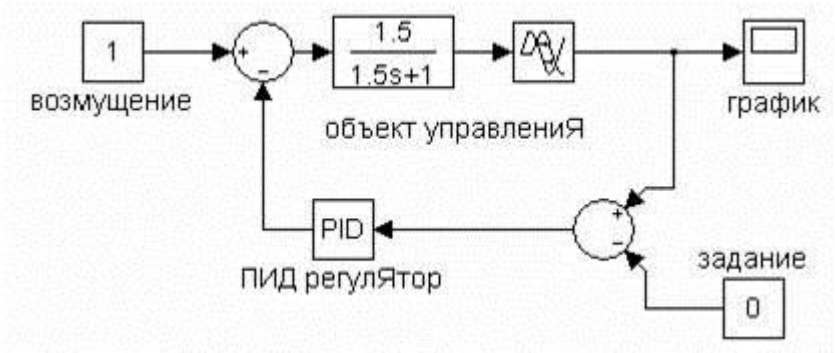


Рисунок 65 - Модель одноконтурной системы автоматического регулирования

5 Задание на самостоятельную работу

1 Создать систему нечеткого вывода для аппроксимации функции $y=k*x^2$ (вариант см. в таблице 18). Для аппроксимации взять 4 значения функции.

Таблица 18 – Варианты задания для аппроксимации функции

Вариант	1	2	3	4	5	6	7	8
k	1.5	1.8	1.3	1.9	2	2.2	1.4	2.5
mf input	trimf	gaussmf	trimf	gaussmf	trimf	trimf	trimf	gaussmf2

9	10	11	12	13	14	15	16	17	18
2.3	2.4	3.5	3.8	3.3	3.9	3	3.2	3.4	4
trimf	gaussmf	trimf	gaussmf	trimf	gaussmf	trimf	trimf	trimf	gaussmf

2 Исследовать АСР с аналоговым ПИ-регулятором и fuzzy-регулятором. В качестве модели объекта принять апериодическое звено, параметры объекта принять из таблицы 19.

Таблица 19 – Параметры объекта к заданию 5.2

Вариант	1	2	3	4	5	6	7	8
K	1.5	1.8	1.3	1.9	2	2.2	1.4	2.5
T	1.8	1.8	1.5	1.5	1.3	1.5	1.5	1.6

9	10	11	12	13	14	15	16	17	18
2.3	2.4	3.5	3.8	3.3	3.9	3	3.2	3.4	4
1.4	1.7	1.8	1.5	1.5	1.5	1.8	1.2	1.4	1.9

Настройки ПИ-регулятора определить по формульному методу:

Регулятор	Апериодический процесс	Процесс с перерегулированием 20 %	Процесс с минимальным временем регулирования
ПИ	$K_1 = \frac{0{,}6 \cdot T}{K \cdot \tau}$, $K_0 = \frac{1}{K \cdot \tau}$	$K_1 = \frac{0{,}7 \cdot T}{K \cdot \tau}$, $K_0 = \frac{1}{K \cdot \tau}$	$K_1 = \frac{T}{K \cdot \tau}$, $K_0 = \frac{1}{K \cdot \tau}$

6 Содержание отчета

1 Цель работы.

2 Исходные данные для задачи 5.1. Правила нечеткого вывода, форма функций принадлежности на всем диапазоне значений аргумента, результат вывода в графическом виде, относительная погрешность аппроксимации. Результат оформить в форме таблицы:

X				
Y				
$y^{аппрокс.}$				
σ, %				

3 Исходные данные для задачи 5.2. Блок схема моделируемой системы, графики переходных процессов для аналогового и нечеткого регуляторов. Параметры элементов системы нечеткого вывода проектируемой системы (тип функций принадлежности, способ реализации логических выражений, способ дефаззификации). Анализ качества переходных процессов.

4 Выводы по работе.

7 Контрольные вопросы

1 Что такое нечеткая логика? В каких приложениях используется?
2 Операции над нечеткими множествами.
3 Способы задания нечеткого множества.
4 Функция принадлежности. Типы функций принадлежности.
5 Алгоритмы нечеткого вывода. Алгоритм Мамдани.
6 Алгоритм Сугено.
7 Экспертные системы на основе использования нечеткой логики.
8 Гибридные сети.

Список литературы

1 Попов Э.В., Фоминых И.Б., Кисель Е.Б. Статические и динамические экспертные системы (классификация, состояние, тенденции): метод. материалы. -М.: Центральный росс. дом знаний, 1995. -126 с.

2 Керридж А.Е. Использование экспертных систем // Нефть, газ и нефтехимия за рубежом. -1987. -№ 9. -С. 107-110.

3 Фрэнк Дж. Бартос. Искусственный интеллект: принятие решений в сложных системах управления // Мир компьютерной автоматизации. -1997. -№ 4. -С. 2-27.

4 Нейлор К. Как построить свою экспертную систему / пер. с англ. -М.: Энергоатом издат, 1991. -286 с., ил.

5 Ларьер Ж.-Л. Системы искусственного интеллекта: Пер. с франц. -М.: Мир, 1991.- 568 с., ил.

6 Искусственный интеллект: в 3-х кн. Справочник / Под ред. В.Н. Захарова, В.Ф. Хорошевского -М.: Радио и связь, 1990. -Кн. 1. - 426 с.; кн. 2. - 304 с.; кн. 3. - 368 с.

7 Базы знаний интеллектуальных систем / Т.А. Гаврилова, В.Ф. Хорошевский. -СПб.: Питер, 2000.- 384 с., ил.

8 Автоматическое проектирование информационно-управляющих систем. Проектирование экспертных систем на основе системного моделирования / Г.Г. Куликов, А.Н. Набатов, А.В. Речкалов и др.; Уфимск. гос. авиац. техн. ун-т. -Уфа, 1999.-233 с.

9 Заде Л.А. Понятие лингвистической переменной и ее применение к принятию приближенных решений. -М.: Мир, 1976.

10 Дьяконов В., Круглов В. Математические пакеты расширения MATLAB: специальный справочник. – СПб.: Питер, 2001. – 480 с.

11 Асаи К. и др. Прикладные нечеткие системы / пер. с япон.; под. ред. Тэрано Т., Асаи К. Сугэно М. – М.: Мир, 1993. –368с.

12 Искусственный интеллект: справочник / под ред. Э.В. Попова. – М.: Радио и связь, 1990. – Т.1. – 461 с., Т.2. – 304 с., Т.3 – 363 с.

13 Кафаров В.В., Дорохов И.М., Марков В.П. Системный анализ процессов химической технологии. Применение метода нечетких множеств. – М.:Наука, 1986. – 356.

14 Кофман А. Введение в теорию нечетких множеств. –М.: Радио и связь, 1982. – 432 с.

15 Мелихов А.Н., Бернштейн Л.С., Коровин С.Я. Ситуационные советующие системы с нечеткой логикой. – М.: Наука, 1990.

Глоссарий

1. Искусственный интеллект а) научное направление, в рамках которого ставятся и решаются задачи аппаратного или программного моделирования тех видов человеческой деятельности, которые традиционно считаются интеллектуальными; б) свойство интеллектуальных систем выполнять функции (творческие), которые традиционно считаются прерогативой человека.
2. Интеллектуальная система (ИС) - техническая или программная система, способная решать задачи, традиционно считающиеся творческими, принадлежащие конкретной предметной области, знания о которой хранятся в памяти ИС. Структура ИС включает три основных блока - базу знаний, решатель, интеллектуальный интерфейс.
3. Система, основанная на знаниях - интеллектуальная система, функционирование которой определяется совокупность знаний о проблемной области , в которой она используется.
4. Экспертная система (ЭС) – интеллектуальная система, предназначенная для оказания консультационной помощи специалистам, работающим в некоторой предметной области. Особенностью ЭС является наличие в них системы объяснений, повышающей консультационную силу ЭС.
5. Система объяснений - одна из функций ИС, она. предоставляет пользователю информацию о том, как интеллектуальная система получила выданное пользователю решение. В отличие от обоснования система объяснений опирается лишь на тот маршрут, который сохранился в памяти системы от процесса поиска решения. Используя этот маршрут, интеллектуальная система формирует пользователю объяснение. на профессиональном естественном языке, позволяющее ему представить все принципиальные шаги решения.
6. Инженерия знаний – раздел искусственного интеллекта, в рамках которого решаются проблемы, связанные с извлечением знаний, приобретением знаний, представлением знаний и манипулированием знаниями. Инженерия знаний служит основой для создания экспертных систем и других интеллектуальных систем.
7. Нейробионика - направление в исследованиях по искусственному интеллекту для которого характерно использование для воспроизведения в интеллектуальных системах процессоров, присущих биологическим объектам, структур и функций, аналогичных структурам и функциям этих объектов. В рамках этого направления были созданы формальные модели нейронов, на основе которых строятся сети, позволяющие решать задачи распознавания образов, классификации.
8. Предметная (проблемная) область - совокупность реальных или абстрактных объектов (сущностей), связей и отношений между этими объектами, а также процедур преобразования этих объектов для решения

задач, возникающих в предметной области.

9. Знания - совокупность сведений, образующих целостное описание, соответствующее некоторому уровню осведомленности об описываемом вопросе, предмете, проблеме и т.д.
10. База знаний - совокупность программных средств, обеспечивающих поиск, хранение, преобразование и запись в памяти ЭВМ сложно структурированных информационных единиц (знаний).
11. Решатель - система, способная благодаря встроенной в нее общей стратегии нахождения решения путем поиска в пространстве альтернатив или путем логического вывода находить решения задач.
12. Логический вывод - последовательность рассуждений, приводящая от посылок к следствию с использованием аксиом и правил вывода.
13. Вывод на знаниях - вывод, использующий в качестве посылок выражения, хранящиеся в базе знаний.
14. Интеллектуальный интерфейс - интерфейс, в который включены средства, позволяющие человеку вести общение с ЭВМ, не используя для ввода в ЭВМ специальные программы.
15. Инженер по знаниям - специалист, основной задачей которого является проектирование баз знаний и наполнение их знаниями по проблемной области. В процессе этой деятельности инженер по знаниям выбирает форму представления знаний, удобную для данной проблемной области, организует приобретение знаний из различных источников (официальные документы, учебники, монографии и т.п.), а также в результате общения с экспертами-специалистами в данной проблемной области.
16. Представление знаний - совокупность методов и процедур, которые применяет инженер по знаниям при заполнении им базы знаний. Представление знаний предполагает использование источников знаний двух типов: пассивных и активных. К первым относятся официальные документы, инструкции, печатные издания, кино-фото-документы и многие другие источники, в которых содержатся сведения, важные для описания знаний о предметной области. Ко второму типу источников знаний относятся люди - специалисты в данной предметной области.

Приложения

Приложение 1

Структура отчета по лабораторной работе

1. Титульный лист (форма титульного листа приведена в приложении 2)
2. Задание на проектирование. Задание на проектирование оформляется в соответствии с выбранным заданием.
3. Содержание. Содержание включает наименование всех разделов, подразделов и пунктов, если они имеют наименование, а также список литературы и приложений, с указанием номера страниц, с которых они начинаются. Форма оформления содержания приведена в приложении 3.
4. Введение. Введение должно содержать вводную информацию к лабораторной работе. То есть цель лабораторной работы, краткое содержание теоретического материала по данной теме.
5. Основная часть работы должна содержать этапы выполнения лабораторной работы и результаты выполнения контрольных тестов (результаты ответа системы на вопросы, приведенные в заданиях к работам)
6. Заключение. Заключение должно содержать основные выводы, сделанные студентом, по выполнению лабораторной работы.
7. Список используемой литературы. В список литературы должны быть включены все источники, которые были использованы при выполнении лабораторной работы. Пример оформления списка литературы приведен в приложении 4
8. Приложения. Приложения содержат вспомогательный материал, такой как базы знаний в виде одной из моделей представления знаний.

Приложение 2

Форма титульного листа к лабораторной работе

Министерство образования Российской Федерации
Министерство сельского хозяйства Российской Федерации
Иркутская государственная сельскохозяйственная академия

Кафедра информатики и математического моделирования

Тема лабораторной работы

Отчет по лабораторной работе № 1 по дисциплине «Интеллектуальные информационные системы»

Выполнил:
Студент гр. 00-00
Иванов И.И.
Принял:
Должность преподавателя
Иванов И.И.

Иркутск 20XX

Приложение 3

Пример оформления содержания

Содержание

Приложение 4

Пример оформления списка литературы

1. Андрейчиков, А.В. Интеллектуальные информационные системы: Учебник / Андрейчикова О.Н. // М.: Финансы и статистика, 2004. – 424 с.
2. Афонин, В. Л. Интеллектуальные робототехнические системы. / Макушкин В. А. // Серия: Основы информационных технологий. Издательство: Интернет-университет информационных технологий, 2005. - 208 с.
3. Братко, Иван. Язык PROLOG (Пролог): алгоритмы искусственного интеллекта. / Братко, Иван // 3-е издание. М.: Вильямс, 2000. - 640 с.
4. Гаврилова, Т.А. Базы знаний интеллектуальных систем. / Хорошевский В.Ф. // СПб.: Питер, 2012 – 384с.
5. Гарднер, Говард. Структура разума: теория множественного интеллекта. / Гарднер, Говард // М.: Вильямс, 2003. - 512 с.
6. Гладков, Л.А. Генетические алгоритмы. / Курейчик, В.В., Курейчик, В.М. // М.: Физматлит, 2010. – 320.