ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ОРЕНБУРГСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОБУЧАЮЩИХСЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ

Б1.В.08 Системы искусственного интеллекта

Направление подготовки 27.03.04 Управление в технических системах

Профиль подготовки Интеллектуальные системы обработки информации и управления

Квалификация выпускника бакалавр

Форма обучения очная

СОДЕРЖАНИЕ

1. КОНСПЕКТ ЛЕКЦИЙ

1.1. Лекция № 1 (2 часа).

Тема: «Понятие искусственного интеллекта»

1.1.1. Вопросы лекции:

1. Понятие ИИ.
2. Основные направления развития ИИ.
3. Основные понятия ИИС.
4. Классификация ИИС.

1.1.2. Краткое содержание вопросов:

1. Понятие ИИ.

Интеллект — что означает ум, рассудок, разум, мыслительные способности человека.

Искусственный интеллект (ИИ) — обычно толкуется как свойство автоматических систем брать на себя отдельные функции интеллекта человека, например, выбирать и принимать оптимальные решения на основе ранее полученного опыта и рационального анализа внешних воздействий.

ИИ как наука существует около полувека. Первой ИИС считается программа «Логик-Теоретик», предназначенная для доказательства теорем. Ее работа впервые была продемонстрирована 9 августа 1956 г. В создании программы участвовали такие известные ученые, как А. Ньюэлл, А. Тьюринг, К. Шеннон, Дж. Лоу, Г. Саймон и др.

На сегодняшний день не существует единого определения, которое однозначно описывает ИИ как научную область. Среди многих точек зрения на нее доминируют следующие три.

Исследования в области ИИ относятся к фундаментальным, в процессе которых разрабатываются новые модели и методы решения задач, традиционно считавшихся интеллектуальными и не поддававшихся ранее формализации и автоматизации.

Согласно второй точки зрения это связано с новыми идеями решения задач на ЭВМ, с разработкой новых технологий программирования и с переходом к компьютерам не фон-неймановской архитектуры.

Третья точка зрения, наиболее прагматическая, основана на том, что в результате исследований, проводимых в области ИИ, появляется множество прикладных систем, способных решать задачи, для которых ранее создаваемые системы были непригодны.

2. Основные направления развития ИИ.

ИИС проникают во все сферы нашей жизни, поэтому трудно провести строгую классификацию направлений, по которым ведутся исследования в области ИИ. Рассмотрим кратко некоторые из них.

Разработка ИИС, основанных на знаниях - это одно из главных направлений ИИ. Основной целью построения таких систем является 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 БЗ, образующих ядро СОЗ. Частным случаем СОЗ является ЭС.

Разработка естественно-языковых интерфейсов и машинный перевод. Системы машинного перевода строятся как ИИС, т.к. в их основе лежат БЗ в определенной предметной области и сложные модели, обеспечивающие дополнительную трансляцию «исходный язык оригинала - язык смысла — язык перевода». Данное направление

охватывает также исследования методов и разработку систем, обеспечивающих реализацию процесса общения человека с ПК на естественном языке.

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.

Обработка визуальной информации. Задача обработки изображений связана с трансформированием графических образов, результатом которой 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 (системы машинной графики).

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).

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 знаний признаков.

Игры и машинное творчество. Машинное творчество охватывает сочинение компьютерной музыки, ИИС для изобретения новых объектов. Создание интеллектуальных компьютерных игр является одним из самых развитых коммерческих направлений в сфере разработки ПО.

Программное обеспечение систем ИИ. Инструментальные средства для разработки ИИС включает специальные языки программирования, ориентированные на обработку символьной информации (LISP, SMALLTALK, РЕФАЛ), языки логического программирования (PROLOG), языки ПЗ (OPS 5, KRL, FRL), интегрированные программные среды, содержащие арсенал инструментальных средств для создания систем ИИ (KE, ARTS, GURU, G2), а также оболочки ЭС (BUILD, EMYCIN, EXSYS Professional, ЭКСПЕРТ), которые позволяют создавать прикладные ЭС, не прибегая к программированию.

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.

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

Мы, в нашем курсе, интеллектом будем называть способность мозга решать задачи путем приобретения, запоминания и целенаправленного преобразования знаний в процессе обучения на опыте и адаптации к разнообразным обстоятельствам.

3. Основные понятия ИИС.

Понятие данные, знания и информация отражают три аспекта предметной области сознаний субъекта.

1. Синтаксический
2. Семантический
3. Прагматический

Синтаксический аспект действительности реализует данные, которые представляют собой записанные посредством какого-либо носителя, факты и их зависимости. Факты предметной области могут быть представлены в виде описаний на естественном языке, в виде графических диаграмм и математических формул.

Знания представляют собой данные, осмысленные или понятые субъектом, которые запоминаются для последующего целенаправленного использования. Знания представляют семантический аспект предметной области.

Информация – приращение знаний субъекта. Информация – новые и полезные данные, осмысливаемые на основе имеющегося знания. Информация – отражение прагматического аспекта предметной области.

ИИС – это ИС, которая основана на концепции использования БЗ для генерации алгоритмов прикладных задач различных классов в зависимости от конкретных информационных потребностей пользователей.

4. Классификация ИИС.

Выделяют 4 основных признака интеллектуальности ИС:

1. Развитые коммуникативные способности, т.е. способы взаимодействия конечного пользователя с системой;
2. Умение решать сложные плохо формализуемые задачи, которые требуют построения оригинального алгоритма решения в зависимости от конкретной ситуации, характеризующейся неопределенностью и динамичностью исходных данных и знаний;
3. Самообучаемость, т.е. умение системы автоматически извлекать знания из накопленного опыта и применять их для решения задач;
4. Адаптивность – адекватное отражение действительности, способность системы к развитию в соответствии с объективными изменениями области знаний.

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

Под интеллектуальным интерфейсом будем понимать интерфейс, обеспечивающий непосредственное взаимодействие конечного пользователя и ПК при решении задачи в составе человеко-машинной системы, выполняющий 3 группы функций:

• обеспечение для пользователя возможности постановки задачи для ЭВМ путем сообщения только ее условия (без задания программы решения);

• обеспечение для польз-ля возможности формирования сред решения задачи с использованием только терминов и понятий из области профессиональной деятельности пользователя, естественных форм представления информации;

• обеспечение гибкого диалога с использованием разных средств, в том числе не регламентируемых заранее, с коррекцией возможных ошибок пользователя.

Гипертекстовые системы

В основе гипертекстного подхода к изложению знания лежит так называемая "метафора гипертекста", суть которой в том, что в памяти человека знания "упакованы" в виде отдельных идей и фактов, между которыми установлены логико-смысловые связи. ГС реализует такую структуру на физических носителях памяти.

Примерами гипертекста служат толковые словари и энциклопедии, состоящие из статей, в которых содержатся ссылки на другие статьи.

1.2. Лекция № 2 (2 часа).

Тема: «Знания и данные. Извлечение знаний»

1.2.1. Вопросы лекции:

1. Свойства знаний и отличие знаний от данных.
2. Типы знаний: декларативные и процедурные, экстенсиональные и интенсиональные.
3. Нечеткие знания.
4. Источники экспертных знаний, извлечение и структурирование знаний, стадии приобретения знаний, автоматизированное приобретение знаний.

1.2.2. Краткое содержание вопросов:

1. Свойства знаний и отличие знаний от данных.

Основным отличительным признаком СИИ является работа со знаниями. Если для, обычных программ представление данных алгоритма определяется на уровне описания языка программирования, то для СИИ представление знаний выливается в проблему, связанную со многими вопросами: что такое знания, какие знания хранить в системе в виде базы знаний (БЗ), в каком виде и сколько, как их использовать, пополнять и т. д.

В отличие от данных знания обладают следующими свойствами:

1) Внутренней интерпретируемостью — вместе с информацией в БЗ представлены информационные структуры, позволяющие не только хранить знания, но и использовать их;
2) Структурированностью — выполняется декомпозиция сложных объектов на более простые и установление связей между ними;
3) Связанностью — отражаются закономерности относительно фактов, процессов, явлений и причинно-следственные отношения между ними;
4) Активностью — знания предполагают целенаправленное использование информации, способность управлять информационными процессами по решению определенных задач.

Все эти свойства знаний в конечном итоге должны обеспечить возможность СИИ моделировать рассуждения человека при решении прикладных задач — со знаниями тесно связано понятие процедуры получения решений задач (стратегии обработки знаний). В системах обработки знаний такую процедуру называют механизмом вывода, логическим выводом или машиной вывода. Принципы построения механизма вывода в СИИ определяются способом представления знаний и видом моделируемых рассуждений.

2. Типы знаний: декларативные и процедурные, экстенсиональные и интенсиональные.

Обрабатываемую на ЭВМ информацию следует разделять на процедурную и декларативную. Процедурную информацию составляют программы для решения тех или иных задач, декларативную – данные, с которыми эти программы работают. Стандартной формой представления информации в памяти ЭВМ является, как правило, машинное слово. В современных ЭВМ для представления данных и команд используются одинаковые по числу разрядов машинные слова. Одинаковое число разрядов в машинном слове для команд и для данных позволяет рассматривать их в ЭВМ в качестве одинаковых информационных единиц и выполнять сходные операции над ними. Содержимое памяти образует информационную базу.

В 1970-е годы с развитием теории баз данных происходит формирование основных моделей представления данных: сетевой, иерархической и реляционной. В рамках следующего поколения моделей данных происходит постепенное слияние данных и знаний. В современных развитых моделях представления данных выделяют интенсиональные и экстенсиональные представления.

В экстенсиональную часть входят конкретные факты, касающиеся предметной области. В интенсиональную часть входят схемы связей между атрибутами. Фактически

экстенсиональные представления описывают конкретные объекты, события, процессы и явления для рассматриваемой предметной области. Интенсиональные представления фиксируют закономерности и связи, которым эти конкретные объекты, события, процессы и явления обязаны в данной проблемной области удовлетворять. Экстенсиональные представления относятся к данным. Относительно интенсиональных представлений говорят либо как о схемах баз данных, либо как о знаниях о проблемной области.

3. Нечеткие знания.

Ряд понятий человеческих знаний оказывается трудно, а иногда и невозможно описать количественно, используя детерминированные или стохастические методы. Трудности возникают при создании моделей не полностью определенных, неточных, нечетких знаний. Это связано с тем, что человеческому мышлению присуща лингвистическая неопределенность; знания и понятия, которыми оперирует человек, часто имеют качественную природу, они ситуативны, бывают неполными. Для формализации знаний такого типа используется аппарат теории нечетких множеств, создание которого связано с именем известного американского ученого Л. Заде.

4. Источники экспертных знаний, извлечение и структурирование знаний, стадии приобретения знаний, автоматизированное приобретение знаний.

Рассматривая методы приобретения знаний, будем использовать следующие термины: извлечение, получение, формирование, приобретение знаний и обучение БЗ. Определим сущность указанных терминов. Под извлечением знаний будем понимать процесс приобретения материализованных знаний из текстологических источников информации с помощью некоторой совокупности методов и процедур, позволяющих переходить от знаний в текстовой форме к их аналогам для ввода в базу знаний СИИ. Получение знаний – это процесс приобретения вербализуемых и невербализуемых знаний эксперта, основанный на использовании непосредственно им самим или инженером по знаниям приемов, процедур, методов и инструментальных средств.

Формирование знаний – это процесс автоматического приобретения (порождения) системой искусственного интеллекта или инструментальным средством нового и полезного знания из исходной и текущей информации, которое в явном виде не формируют эксперты, в целях освоения новых процедур решения прикладных задач на основе использования различных моделей машинного обучения. Под приобретением знаний будем понимать процесс, основанный на переносе знаний из различных источников в базу знаний путем использования различных методов, моделей, алгоритмов и инструментальных средств.

Понятие получение знаний соотносится с понятиями извлечение, приобретение, формирование знаний как часть-целое.

Обучение базы знаний – это процесс ввода (переноса) приобретенных знаний в СИИ на основе применения совокупности методов, приемов и процедур в целях ее заполнения, расширения и модификации. Термин обучение рассматривается как свойство БЗ, как совокупность методов, приемов и процедур ввода знаний в БЗ и как процесс переноса знаний в СИИ.

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

Методы извлечения знаний. Они состоят из текстологических методов и методов автоматической обработки текстов.

Текстологические методы предназначены для получения инженером по знаниям знаний из материализованных источников, в качестве которых выступают монографии, учебники, статьи, методики, инструкции и другие носители профессиональных знаний. Текстологические методы, несмотря на их простоту и тривиальность, являются наименее разработанными. Эти методы основываются не только на выявлении и понимании смысла

текста, но и на выделении базовых понятий и отношений, т. е. формировании семантической (понятийной) структуры ПрО.

Процесс понимания является сложным и не формализуемым, на него существенно влияют когнитивный стиль инженера по знаниям и его интеллектуальные характеристики. В инженерии знаний разработана методика анализа текстов в целях извлечения и структурирования знаний. Методика предусматривает овладение инженером по знаниям микроструктурой текста, вычленение ключевых слов (компрессию или сжатие текста) и последующее формирование поля знаний.

Сжатие текста служит методологической основой для использования текстологических процедур извлечения знаний. Текстологические методы являются самыми трудоемкими и применяются, как правило, на начальном этапе создания СИИ.

Значительное развитие получили методы извлечения знаний при применении современных информационных технологий, в частности гипертекстовой технологии.

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

Эти новые концепции работы со знаниями создают предпосылки для решения проблемы эффективности процесса приобретения знаний.

Усилия исследователей в области инженерии знаний направлены на создание формальных методов извлечения знаний. К их числу можно отнести метод автоматической обработки текстов на основе статистической обработки семантических единиц. Метод и программные средства автоматизированного извлечения знаний из текстов базируются на формальных процедурах обнаружения в текстах семантических единиц различной выраженности.

Семантические единицы получаются путем статистической обработки текстов, в основе которой лежат универсальные механизмы определения частотных характеристик терминов. Задача извлечения знаний решается в два этапа: сначала формируется терминологическая сеть (поле знаний), а затем определяется ассоциативная близость терминов на основе статистически определенной меры ассоциации. Достоинство рассмотренного метода состоит в автоматическом выявлении значимых слов и связей с учетом статистической информации о гипертексте в целом.

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

Методы получения экспертных знаний. К ним относятся следующие методы: коммуникативные (пассивные и активные), основанные на прямом диалоге экспертов и инженеров по знаниям как без использования СИИ, так и с применением СИИ (технологии окон, меню); психосемантики и тестирования БЗ.

Коммуникативные методы получения знаний рассматриваются как разновидности интервьюирования. Для них характерны следующие основные особенности:

1) Не имеют формального определения и носят качественный характер. Полученные с их помощью знания несут на себе отпечаток самонаблюдений эксперта и субъективную интерпретацию инженера по знаниям.

2) Требуют словесного выражения экспертом своих знаний, что является непростой задачей. Неточность и неадекватность словесных описаний мыслительных процессов и применяемых эвристических приемов, используемых при решении задач, ведут к серьезным последствиям.

3) Сложность выражения процедурных знаний при их словесном описании.

4) Крайняя сложность явного описания знаний, которые являются результатом компиляции и автоматизма процессов мышления, а также интуиции эксперта. В психологии

доказано, что интуиция на самом деле является способностью распознавать образы. Однако словесное описание способности к распознаванию образов дать крайне трудно.

5) Трудоемкость организации и неэффективность взаимодействия инженера по знаниям и эксперта. На них приходятся большие интеллектуальные нагрузки, связанные с вербализацией знаний, управлением процессом коммуникации и необходимостью освоения, анализа и документирования больших объемов новых знаний.

Коммуникативные методы получения знаний отличаются своей низкой эффективностью. Так, при непосредственном взаимодействии инженера по знаниям и эксперта теряется до 76% информации.

Один из путей совершенствования процесса приобретения знаний состоит в разработке методов, позволяющих передать часть функций, выполняемых инженером по знаниям, самому эксперту или СИИ.

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

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

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

Главный вопрос, на который должны ответить методы формирования знаний, состоит в следующем: как от частного (примера) перейти к общему (обобщениям)?

Базисом всех методов формирования знаний является индукция, которая лежит в основе получения общих выводов из совокупности частных утверждений.

1.3. Лекция № 3 (2 часа).

Тема: «Представление знаний в интеллектуальных системах»

1.3.1. Вопросы лекции:

1. Модели представления знаний в системах ИИ.
2. Формальные логические модели представления знаний.
3. Правила-продукции. Структура правил-продукций. Типы ядер правил-продукций и варианты их интерпретаций. Методы логического вывода: прямой и обратный. Стратегии выбора правил при логическом выводе.

1.3.2. Краткое содержание вопросов:

1. Модели представления знаний в системах ИИ.

Модели представления знаний – это одно из важнейших направлений исследований в области искусственного интеллекта. Почему одно из важнейших? Да потому, что без знаний искусственный интеллект не может существовать в принципе. Действительно, представьте себе человека, который абсолютно ничего не знает. Например, он не знает даже таких элементарных вещей как: для того, чтобы не умереть от голода, необходимо периодически есть; не обязательно из одного края города в другой идти пешком, если для этих целей можно воспользоваться общественным транспортом.

Таких примеров удастся привести еще много, но уже сейчас можно легко ответить на следующий вопрос: «Поведение такого человека может считаться разумным?». Конечно же, нет. Именно поэтому, при создании систем искусственного интеллекта особенное внимание уделяется моделям представления знаний.

На сегодняшний день разработано уже достаточное количество моделей. Каждая из них обладает своими плюсами и минусами, и поэтому для каждой конкретной задачи

необходимо выбрать именно свою модель. От этого будет зависит не столько эффективность выполнения поставленной задачи, сколько возможность ее решения вообще.

Отметим, что модели представления знаний относятся к прагматическому направлению исследований в области искусственного интеллекта. Это направление основано на предположении о том, что мыслительная деятельность человека – «черный ящик». При таком подходе не ставится вопрос об адекватности используемых в компьютере моделей представления знаний тем моделям, которыми пользуется в аналогичных ситуациях человек, а рассматривается лишь конечный результат решения конкретных задач.

Рассмотрим три наиболее часто используемые и популярные на сегодняшний день модели представления знаний: продукционные модели – модели основанные на правилах, позволяют представить знание в виде предложений типа: «ЕСЛИ условие, ТО действие». Продукционные модели обладает тем недостатком, что при накоплении достаточно большого числа правил, они начинают противоречить друг другу; сетевые модели или семантические сети – как правило, это граф, отображающий смысл целостного образа. Узлы графа соответствуют понятиям и объектам, а дуги – отношениям между объектами; фреймовые модели – основывается на таком понятии как фрейм (англ. frame – рамка, каркас). Фрейм – структура данных для представления некоторого концептуального объекта. Информация, относящаяся к фрейму, содержится в составляющих его слотах. Слоты могут быть терминальными либо являться сами фреймами, т.о. образуя целую иерархическую сеть.

2. Формальные логические модели представления знаний.

Основная идея при построении логических моделей знаний заключается в следующем – вся информация, необходимая для решения прикладных задач, рассматривается как совокупность фактов и утверждений, которые представляются как формулы в некоторой логике. Знания отображаются совокупностью таких формул, а получение новых знаний сводится к реализации процедур логического вывода. В основе логических моделей знаний лежит понятие формальной теории, задаваемое картежем:

S~=~<A,~F,~Ax,~R>

A – счетное множество базовых символов (алфавит);

F – множество, называемое формулами;

Ax – выделенное подмножество априори истинных формул (аксиом);

R – конечное множество отношений между формулами, называемое правилами вывода.

Основные достоинства логических моделей знаний:

1) В качестве «фундамента» здесь используется классический аппарат математической логики, методы которой достаточно хорошо изучены и формально обоснованы;
2) Существуют достаточно эффективные процедуры вывода, в том числе реализованные в языке логического программирования «Пролог»;
3) В базах знаний можно хранить лишь множество аксиом, а все остальные знания получать из них по правилам вывода.

В логических моделях знаний слова, описывающие сущности предметной области, называются термами (константы, переменные, функции), а слова, описывающие отношения сущностей – предикатами.

Предикат – логическая N-арная пропозициональная функция, определенная для предметной области и принимающая значения либо истинности, либо ложности. Пропозициональной называется функция, которая ставит в соответствие объектам из области определения одно из истинностных значений («истина», «ложь»). Предикат принимает значения «истина» или «ложь» в зависимости от значений входящих в него термов.

Способ описания предметной области, используемый в логических моделях знаний, приводит к потере некоторых нюансов, свойственных естественному восприятию человека, и поэтому снижает описательную возможность таких моделей.

Сложности возникают при описании «многосортных» миров, когда объекты не являются однородными. Так, высказывания:

«2 + 2 = 4»

«Москва – столица России»

имеют одно и то же значение «истина», но разный смысл. С целью преодоления сложностей и расширения описательных возможностей логических моделей знаний разрабатываются псевдофизические логики, логики, оперирующие с нечеткостями, эмпирическими кванторами, обеспечивающие индуктивные (от частного к общему), дедуктивные (от общего к частному) и традуктивные (на одном уровне общности) выводы. Такие расширенные модели, объединяющие возможности логического и лингвистического подходов, принято называть логико-лингвистическими моделями предметной области.

3. Правила-продукции. Структура правил-продукций. Типы ядер правил-продукций и варианты их интерпретаций. Методы логического вывода: прямой и обратный. Стратегии выбора правил при логическом выводе.

Процедура логического вывода в системах, основанных на продукционных моделях, в принципе не сложная. Как правило, она включает следующие части:

1) Рабочую память (базу данных) – фактические данные, описывающие возможное и текущее состояние предметной области – хранящуюся в оперативной памяти;
2) Базу продукционных правил, содержащую все допустимые зависимости между фактами предметной области и хранящуюся в долговременной памяти;
3) Механизм логического вывода.

Механизм логического вывода обеспечивает формирование заключений, воспринимая вводимые факты как элементы правил, отыскивая правила, в состав которых входят введенные факты, и актуализируя те части продукций, которым соответствуют введенные факты. Теоретической основой построения механизма логического вывода служит теория машины Поста.

Механизм логического вывода выполняет функции поиска в базе правил, последовательного выполнения операций над знаниями и получения заключений. Существует два способа проведения таких заключений – прямые выводы и обратные выводы.

Пусть имеется совокупность продукций в виде цепочек правил:

$$A \rightarrow B;\ B \rightarrow C;\ C \rightarrow D;\ D \rightarrow E;$$
$$F \rightarrow G;\ G \rightarrow H;\ H \rightarrow D;$$

Прямым выводам (прямой цепочке рассуждений) соответствует движение от посылок к следствиям.

Механизм логического вывода, использующий прямые выводы, в качестве образца выбирает введенный в базу данных (рабочую память) факт А и если при сопоставлении он согласуется с посылкой правила, то делается заключение В, которое тоже помещается в базу данных как факт, описывающий состояние предметной области. Последовательно выводятся новые результаты, начиная с уже известных. Однако отсутствие связи между фактами С и I может привести к обрыву процедуры и конечный результат Е не может быть получен. Это считается основным недостатком прямых механизмов логического вывода и требует от пользователя знания всей структуры модели предметной области. Особенно явно этот недостаток проявляется при включении в базу знаний новых фактов и правил: если они не связаны в цепочку с имеющимися фактами, то они становятся

балластом – механизм логического вывода никогда их не найдет. С этой точки зрения использование обратной цепочки рассуждений предпочтительнее.

Обратным выводам (обратной цепочке рассуждений) соответствует движение от цели (факта, который требуется установить) к предпосылкам. В обратном механизме логического вывода работа начинается от поставленной цели. Если цель А согласуется с консеквентом (заключением) продукции, то антецедент (посылка) принимается за подцель и делается попытка подтверждения истинности этого факта. Процесс повторяется до тех пор, пока не будут просмотрены все правила, имеющие в качестве заключения требуемый факт.

Так, в приведенном примере движение от заключения Е приводит к необходимости подтверждения факта D. Факт D может подтвердиться, если подтверждается I. Если I не подтверждается, то механизм логического вывода отыщет правило, связывающее Dc H и перейдет на анализ второй цепочки правил. Дойдя до правила F right~G, система запросит базу данных (рабочую память) или пользователя о справедливости факта F. Если факт F подтверждается, то происходит возвратное движение по правилам, все факты актуализируются (считаются справедливыми) и цель достигается успешно. В противном случае система явно указывает причину недоказанности выводов, что, в отличие от прямой цепочки рассуждений, облегчает работу пользователя.

Функцией, реализующей работу механизма логического вывода, является рекурсивная процедура сопоставления с образцом.

Рекурсия (лат. «recurso» – бегу назад, спешу обратно, возвращаюсь) – способ решения задач, заключающийся в разбиении исходной задачи на подзадачи. Если подзадача есть уменьшенный вариант исходной задачи, то способ ее разбиения и решения идентичен примененному к исходной задаче. Последовательное разбиение приводит к задаче, решаемой непосредственно. Это решение служит основанием для решения подзадачи верхнего уровня и т. д., пока первоначальная задача не будет решена.

Пример рекурсивных рассуждений:

Как найти льва в пустыне? Для этого следует выполнить следующие шаги:

1) По периметру пустыни поставить забор (чтобы лев не убежал).
2) Поймать льва в выделенном пространстве. Если лев не пойман, то перейти к п. 3, иначе - к п. 5.
3) Выделенное пространство разделить забором на две равные части (в два раза сократить пространство поиска).
4) Выбрать одно из подпространств и перейти к п. 2.
5) Завершение, цель достигнута

В заключении отметим, что в практике наиболее часто встречаются механизмы логического вывода, опирающиеся на обратную цепочку рассуждений. Это обусловлено их более надежной работой (практически всегда имеется возможность найти цепочку рассуждений от конца до начала) и большей производительностью, что становится особенно заметно при большом количестве продукций.

Вывод в формальной логической системе является процедурой, которая из заданной группы выражений выводит отличное от заданных семантически правильное выражение. Эта процедура, представленная в определенной форме, и является правилом вывода. Если группа выражений, образующая посылку, является истинной, то должно гарантироваться, что применение правила вывода обеспечит получение истинного выражения в качестве заключения.

Наиболее часто используются два метода. Первый – метод правил вывода, или метод естественного (натурального) вывода, названный так потому, что используемый тип рассуждений в исчислении предикатов приближается к обычному человеческому рассуждению. Второй – метод резолюций. В его основе лежит исчисление резольвент.

В этой статье рассматривается метод правил вывода. В логике предикатов используется правило, которое из двух выражений A и $A \rightarrow B$ выводит новое выражение B.

В разной литературе можно встретить разные названия метода правил вывода, например, правила дедуктивных выводов или более часто modus ponens. Принцип работы правил вывода хорошо иллюстрирует следующий пример:

«Если известно, что высказывание «A» влечет (имплицирует) высказывание «B», а также известно, что высказывание «A» истинно, то, следовательно, «B» истинно»

В логике предикатов имеются универсальные правила, оперирующие с формулами, содержащими свободные переменные. Решение задач (получение выводов) в логических моделях может основываться на применении подобных правил к исходной совокупности истинных предикатов как доказательство правильности какого-либо составного предиката. Такой способ получения решения называется неаксиоматическим или другими словами – натуральным, естественным и совпадает со способами вывода в продукционных моделях.

Поскольку ответ получается как заключение из комбинации уже существующих логических формул, то по аналогии с выводами в продукционных моделях его можно назвать прямым (обратным) выводом. Однако всегда следует учитывать, что в формальной логике причинно-следственные отношения игнорируются.

Суть процедуры вывода заключается в рекурсивном применении подстановки известных значений в составной предикат. При этом принципиально гарантируется, что доказательство истинности результата можно проверить формальной процедурой.

Если в формальной логической модели механизм логического вывода использует метод правил вывода, то есть основания эту модель отнести к продукционным или логико-лингвистическим.

Пример: вывод решения в логической модели на основе правила вывода – modus ponens.

Даны утверждения:

- «Сократ – человек»;
- «Человек – это живое существо»;
- «Все живые существа смертны».

Требуется доказать утверждение «Сократ смертен».

Решение:

Шаг 1. Представим высказывания в предикатной форме:

УТВЕРЖДЕНИЕ	ПРЕДИКАТНАЯ ФОРМА
«Человек - это живое существо»	$\forall(X)$(Человек(X) → Живое_существо(X))
«Сократ - человек»	Человек(Сократ)
«Все живые существа смертны»	$\forall(Y)$(Живое_существо(Y) → Смертно(Y))

Шаг 2. На основе правила вывода (modus ponens) и подстановки (Сократ/X) в первом предикате получим утверждение:

«Сократ – это живое существо»

Шаг 3. На основе правила вывода (modus ponens) и подстановки (Сократ/Y) в третьем предикате получим утверждение:

«Сократ – смертен»

1.4. Лекция №4 (2 часа).

Тема: «Представление знаний в интеллектуальных системах»

1.4.1. Вопросы лекции:

1. Семантические сети. Основные понятия семантических сетей. Типы отношений в семантических сетях. Абстрактные и конкретные сети.
2. Фреймы и объекты. Основные понятия фрейма: слоты, присоединенные процедуры-слуги и процедуры-демоны, наследование свойств. Связь понятия фрейма и объекта в объетно-ориентированном программировании. Сети фреймов. Принципы обработки данных в сети фреймов.

1.4.2. Краткое содержание вопросов:

1. Семантические сети. Основные понятия семантических сетей. Типы отношений в семантических сетях. Абстрактные и конкретные сети.

Однозначное определение семантической сети в настоящее время отсутствует. В инженерии знаний под ней подразумевается граф, отображающий смысл целостного образа. Узлы графа соответствуют понятиям и объектам, а дуги – отношениям между объектами. Формально сеть можно задать в следующем виде:

$$H = < I, C, G >$$

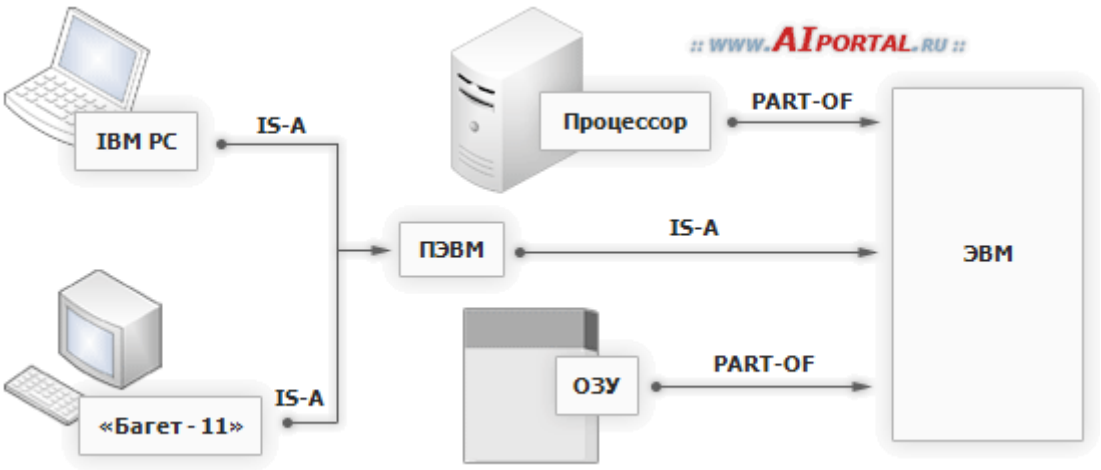
I – множество информационных единиц;
C – множество типов связей между информационными единицами;
G – отображение, задающее конкретные отношения из имеющихся типов C между элементами I.

Семантическая сеть как модель наиболее часто используется для представления декларативных знаний. С помощью этой модели реализуются такие свойства системы знаний, как интерпретируемость и связность, в том числе по отношениям IS-A и PART-OF. За счет этих свойств семантическая сеть позволяет снизить объем хранимых данных, обеспечивает вывод умозаключений по ассоциативным связям.

Одной из первых известных моделей, основанных на семантической сети, является TLC-модель (Teachaple Languge Compre-hender – доступный механизм понимания языка), разработанная Куиллианом в 1968 году. Модель использовалась для представления семантических отношений между концептами (словами) с целью описания структуры долговременной памяти человека в психологии.

Как правило, различают экстенсиональные и интенсиональные семантические сети. Экстенсиональная семантическая сеть описывает конкретные отношения данной ситуации. Интенсиональная – имена классов объектов, а не индивидуальные имена объектов. Связи в интенсиональной сети отражают те отношения, которые всегда присущи объектам данного класса.

Примером семантической сети может служить фрагмент описания вычислительной техники, показанный на рисунке.



Пример семантической сети
С помощью такой сети, используя отношение IS-A и PART-OF, можно вывести факты: «Багет-11» – это ЭВМ; IBM PC имеет процессор и т.д. Для отображения процедурных знаний используются процедурные семантические сети. В этом случае факты, отношения и процедуры представлены как вершины, а связи объединяют их в единое понятие.

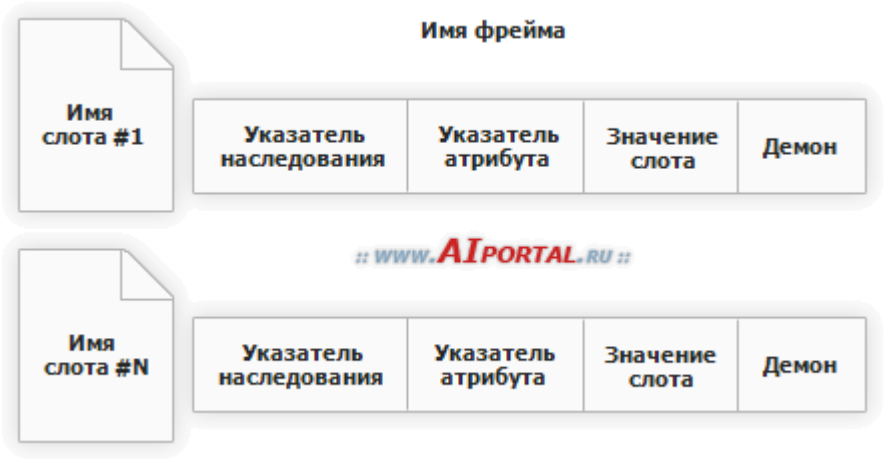
2. Фреймы и объекты. Основные понятия фрейма: слоты, присоединенные процедуры-слуги и процедуры-демоны, наследование свойств. Связь понятия фрейма и объекта в объетно-ориентированном программировании. Сети фреймов. Принципы обработки данных в сети фреймов.

Фреймовая модель основана на концепции Марвина Мински (Marvin Minsky) – профессора Массачусетского технологического института, основателя лаборатории искусственного интеллекта, автора ряда фундаментальных работ. Фреймовая модель представляет собой систематизированную психологическую модель памяти человека и его сознания.

Фрейм (англ. frame – рамка, каркас) – структура данных для представления некоторого концептуального объекта. Информация, относящаяся к фрейму, содержится в составляющих его слотах.

Слот (англ. slot – щель, прорезь) может быть терминальным (листом иерархии) или представлять собой фрейм нижнего уровня.

Каждый фрейм, как показано на рисунке ниже, состоит из произвольного числа слотов, причем несколько из них обычно определяются самой системой для выполнения специфических функций, а остальные определяются пользователем.

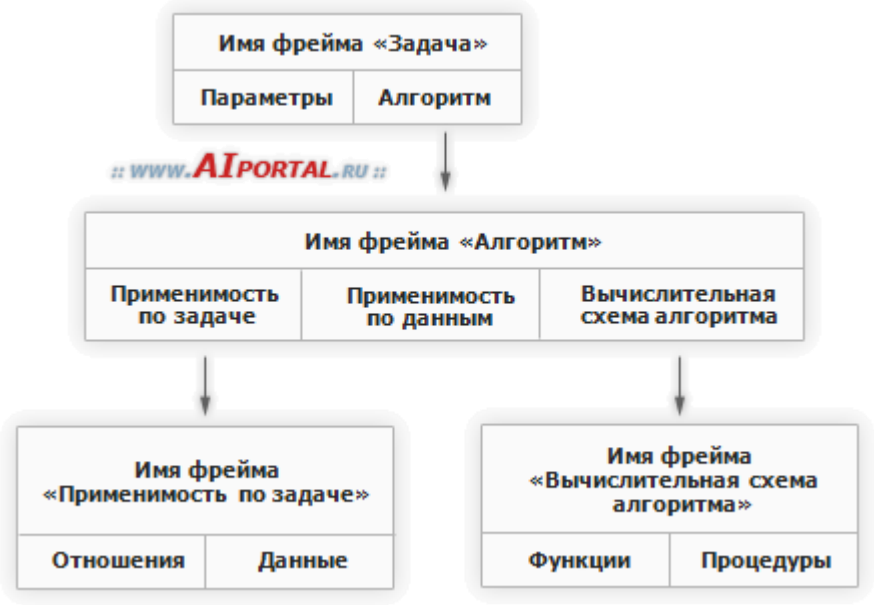


Пояснение:

1) Имя фрейма (имя фрейма) – это идентификатор, присваиваемый фрейму. Фрейм должен иметь имя, единственное в данной фреймовой модели (уникальное имя);
2) Имя слота (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, но в некоторых случаях оно может иметь специфический смысл;
3) Указатель наследования – только для фреймовых моделей иерархического типа; они показывают, какую информацию об атрибутах слотов во фрейме верхнего уровня наследуют слоты с такими же именами во фрейме нижнего уровня;
4) Указатель атрибутов – указатель типа данных слота. К таким типам относятся: FRAME (указатель), INTEGER (целое), REAL (вещественное), BOOL (булево), LISP (присоединенная процедура), TEXT (текст), LIST (список), TABLE (таблица), EXPRESSION (выражение) и другие;
5) Значение слота – значение, соответствующее типу данных слота и удовлетворяющее условиям наследования;

6) Демон – процедура, автоматически запускаемая при выполнении некоторого условия. Демоны запускаются при обращении к конкретному слоту фреймовой модели. Например, демон IF-NEEDED запускается, если в момент обращения к слоту его значение не было установлено, IF-ADDED запускается при подстановке в слот значения, IF-REMOVED запускается при стирании значения слота.

Пример фреймовой модели иерархического типа представлен на рисунке ниже:



Фреймы образуют иерархию. Иерархия во фреймовых моделях порождает единую многоуровневую структуру, описывающую либо объект, если слоты описывают только свойства объекта, либо ситуацию или процесс, если отдельные слоты являются именами процедур, присоединенных к фрейму и вызываемых при его актуализации.

Формально фрейм – это тип данных вида:

$$F = < N, S_1, S_2, S_3 >$$

N – имя объекта;

S1 – множество слотов, содержащих факты, определяющие декларативную семантику фрейма;

S2 – множество слотов, обеспечивающих связи с другими фреймами (каузальные, семантические и т. д.);

S3 – множество слотов, обеспечивающих преобразования, определяющие процедурную семантику фрейма.

Фреймы подразделяются на:

1) Фрейм-экземпляр – конкретная реализация фрейма, описывающая текущее состояние в предметной области;
2) Фрейм-образец – шаблон для описания объектов или допустимых ситуаций предметной области;
3) Фрейм-класс – фрейм верхнего уровня для представления совокупности фреймов образцов.

Состав фреймов и слотов в каждой конкретной фреймовой модели может быть разный, однако в рамках одной системы целесообразно единое представление для устранения лишнего усложнения.

Разнотипные объекты или объекты, соответствующие концепции «множественности миров», заключающейся, к примеру, в том, что лошадь – животное бескрылое для одного (реального) мира и одновременно крылатое (Пегас в мифическом мире) для другого, могут описываться отличающимися друг от друга фреймами.

В целом фреймовая модель допускает представление всех свойств декларативных и процедурных знаний. Глубина вложенности слотов во фрейме (число уровней) зависит от предметной области и языка, реализующего модель.

1.5. Лекция № 5 (2 часа).

Тема: «Общая структура и схема функционирования ЭС».

1.5.1. Вопросы лекции:

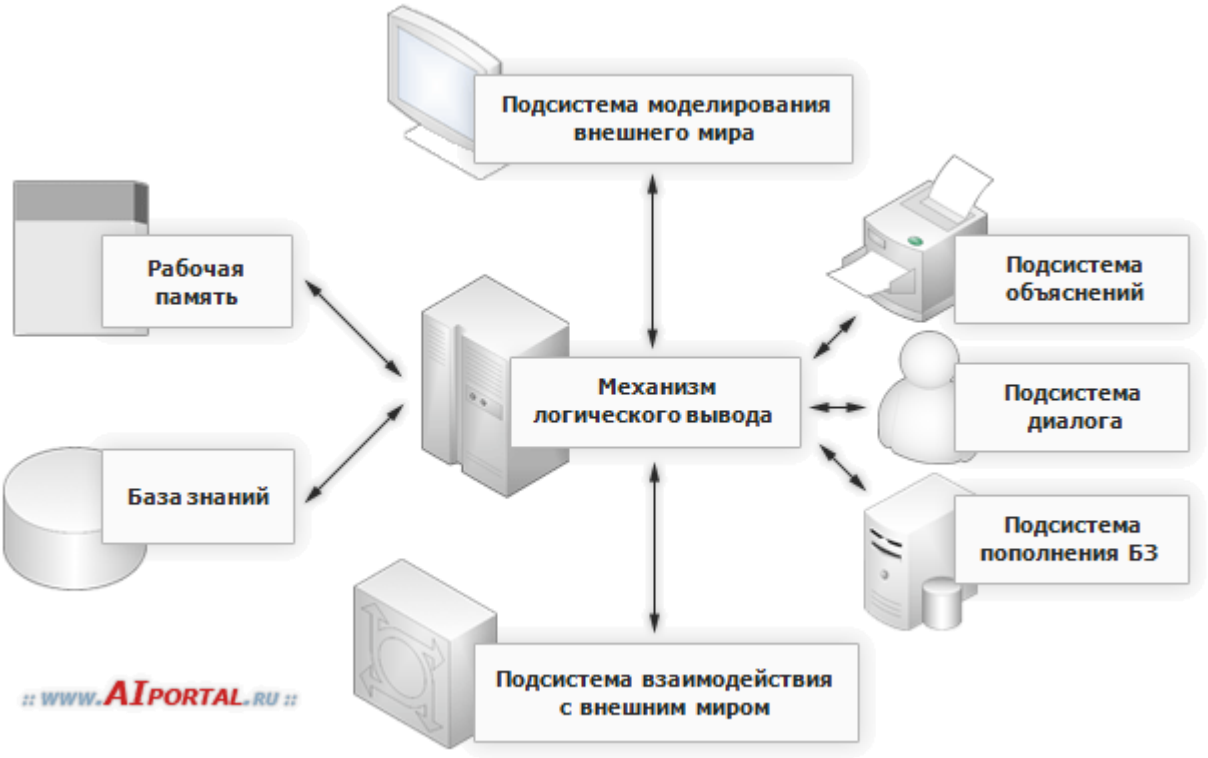
1. Общая структура и схема функционирования ЭС.
2. Основные режимы работы экспертных систем.
3. Объяснительные способности ЭС.

1.5.2. Краткое содержание вопросов:

1. Общая структура и схема функционирования ЭС.

Выделяют два типа экспертных систем: статические и динамические. Статические экспертные системы используются в тех приложениях, где можно не учитывать изменения окружающего мира, происходящие за время решения задачи. Первые экспертные системы, получившие практическое использование, были статическими. Динамические экспертные системы по сравнению со статическими содержат дополнительно два следующих компонента: подсистему моделирования внешнего мира и подсистему взаимодействия с внешним миром.

На рисунке ниже представлена каноническая структура экспертной системы динамического типа:



Пояснение к рисунку:

1) Механизм логического вывода, называемый также интерпретатором, решателем;
2) Рабочую память (РП), называемую также рабочей базой данных (БД);
3) Базу знаний (БЗ);
4) Подсистему приобретения и пополнения знаний;
5) Подсистему объяснения;

6) Подсистему диалога;
7) Подсистему взаимодействия с внешним миром.

Механизм логического вывода (МЛВ) предназначен для получения новых фактов на основе сопоставления исходных данных из рабочей памяти и знаний из базы знаний. Механизм логического вывода во всей структуре экспертной системы занимает наиболее важное место. Он реализует алгоритмы прямого и/или обратного вывода и формально может быть представлен четверкой:

$$< V,S,K,W >$$

V – процедура выбора из базы знаний и рабочей памяти правил и фактов;
S – процедура сопоставления правил и фактов, в результате которой определяется множество фактов к которым применимы правила для присвоения значений;
K – процедура разрешения конфликтов, определяющая порядок использования правил, если в заключении правила указаны одинаковые имена фактов с разными значениями;
W – процедура, осуществляющая выполнение действий, соответствующих полученному значению факта (заключению правила).

Рабочая память предназначена для хранения исходных и промежуточных фактов решаемой в текущий момент задачи. Как правило, размещается в оперативной памяти ЭВМ и отражает текущее состояние предметной области в виде фактов с коэффициентами уверенности (КУ) в истинности этих фактов.

Следующий элемент в структуре экспертной системы не менее важен, чем механизм логического вывода. Это – база знаний. База знаний предназначена для хранения долгосрочных фактов, описывающих рассматриваемую область, правил, описывающих отношения между этими фактами и других типов декларативных знаний о предметной области. Кроме правил и фактов, образующих декларативную часть базы знаний, в нее может входить процедурная часть – множество функций и процедур, реализующих оптимизационные, расчетные и другие требуемые алгоритмы.

Экспертные системы относятся к классу интеллектуальных систем, основывающихся на понимании факта. Другими словами экспертные системы основываются на знаниях специалиста-эксперта о предметной области. Высококачественный опыт наиболее квалифицированных специалистов, доступный для всех пользователей системы, становится фактором, резко повышающим качество принимаемых решений для организации, использующей экспертные системы в целом.

Подсистема приобретения и пополнения знаний автоматизирует процесс наполнения экспертной системы знаниями, осуществляемый пользователем-экспертом, и адаптации базы знаний системы к условиям ее функционирования. Адаптация экспертной системы к изменениям в предметной области реализуется путем замены правил или фактов в базе знаний.

Подсистема объяснения объясняет, как система получила решение задачи (или почему она не получила решения) и какие знания она при этом использовала, что облегчает эксперту тестирование системы и повышает доверие пользователя к полученному результату. Возможность объяснять свои действия является одним из самых важных свойств экспертной системы, так как:

1) Повышается доверие пользователей к полученным результатам;
2) Облегчается отладка системы;
3) Создаются условия для пользователей по вскрытию новых закономерностей предметной области;
4) Объяснение полученных выводов может служить средством поиска точки в парето-оптимальном множестве решений.

Структура экспертной системы была бы неполной без подсистемы диалога. Подсистема диалога ориентирована на организацию дружественного интерфейса со всеми категориями пользователей как в ходе решения задач, так и в ходе приобретения знаний и объяснения результатов работы.

2. Основные режимы работы экспертных систем.

Экспертная система работает в двух режимах: режиме приобретения знаний и в режиме консультаций (называемом также режимом решения или режимом пользования экспертной системой).

В режиме приобретения знаний общение с экспертной системой осуществляет эксперт. В этом режиме эксперт, используя компонент приобретения знаний, наполняет систему знаниями, которые позволяют экспертной системе в режиме консультаций самостоятельно (без эксперта) решать задачи из 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 способы манипулирования с данными, характерные для рассматриваемой области.

В режиме консультации общение с экспертной системой осуществляет конечный пользователь, которого интересует результат и (или) способ его получения. Необходимо отметить, что в зависимости от назначения экспертной системы пользователь может не быть специалистом в данной проблемной области (в этом случае он обращается к экспертной системе за результатом, не умея получить его сам), или быть специалистом (в этом случае пользователь может сам получить результат, но он обращается к экспертной системе с целью либо ускорить процесс получения результата, либо возложить на экспертную систему рутинную работу). Следует подчеркнуть, что термин «пользователь» является многозначным, так как использовать экспертную систему кроме конечного пользователя может и эксперт, и инженер по знаниям, и программист.

В режиме консультации данные о задаче пользователя после обработки их диалоговым компонентом поступают в рабочую память. Решатель на основе входных данных из рабочей памяти, общих данных о проблемной области и правил из базы знаний формирует решение задачи.

Хорошо построенная экспертная система должна иметь возможность самообучаться на решаемых задачах, пополняя автоматически свою базу знаний результатами полученных выводов и решений.

3. Объяснительные способности ЭС.

Объяснительные способности ЭС позволяют пользователям уяснять смысл того или иного ответа, формируемого системой, и проводить оценку этой информации для окончательного принятия решений

1.6. Лекция №6 (2 часа).

Тема: «Этапы построения ЭС».

1.6.1. Вопросы лекции:

1. Этапы построения ЭС.
2. Взаимодействие пользователя с ЭС.

1.6.2. Краткое содержание вопросов:

1. Этапы построения ЭС.

При разработке экспертных систем часто используется концепция быстрого прототипа. Суть её в следующем: поначалу создается не экспертная система, а её прототип, который обязан решать узкий круг задач и требовать на свою разработку незначительное время. Прототип должен продемонстрировать пригодность будущей экспертной системы для данной предметной области, проверить правильность кодировки фактов, связей и стратегий рассуждения эксперта. Он также дает возможность инженеру по знаниям привлечь эксперта к активной роли в разработке экспертной системы. Размер прототипа – несколько десятков правил.

На сегодняшний день сложилась определенная технология разработки экспертных систем, включающая 6 этапов.

Этап 1. Идентификация

Определяются задачи, которые подлежат решению. Планируется ход разработки прототипа экспертной системы, определяются: нужные ресурсы (время, люди, ЭВМ и т.д.), источники знаний (книги, дополнительные специалисты, методики), имеющиеся аналогичные экспертные системы, цели (распространение опыта, автоматизация рутинных действий и др.), классы решаемых задач и т.д. Этап идентификации – это знакомство и обучение коллектива разработчиков. Средняя длительность 1-2 недели.

На этом же этапе разработки экспертных систем проходит извлечение знаний. Инженер по знаниям помогает эксперту выявить и структурировать знания, необходимые для работы экспертной системы, с использованием различных способов: анализ текстов, диалоги, экспертные игры, лекции, дискуссии, интервью, наблюдение и другие. Извлечение знаний – это получение инженером по знаниям более полного представления о предметной области и методах принятия решения в ней. Средняя длительность 1-3 месяца.

Этап 2. Концептуализация

Выявляется структура полученных знаний о предметной области. Определяются: терминология, перечень главных понятий и их атрибутов, структура входной и выходной информации, стратегия принятия решений и т.д. Концептуализация – это разработка неформального описания знаний о предметной области в виде графа, таблицы, диаграммы либо текста, которое отражает главные концепции и взаимосвязи между понятиями предметной области. Средняя длительность этапа 2-4 недели.

Этап 3. Формализация

На этапе формализации все ключевые понятия и отношения, выявленные на этапе концептуализации, выражаются на некотором формальном языке, предложенном (выбранном) инженером по знаниям. Здесь он определяет, подходят ли имеющиеся инструментальные средства для решения рассматриваемой проблемы или необходим выбор другого инструментария, или требуются оригинальные разработки. Средняя длительность 1-2 месяца.

Этап 4. Реализация

Создается прототип экспертной системы, включающий базу знаний и другие подсистемы. На данном этапе применяются следующие инструментальные средства: программирование на обычных языках (Паскаль, Си и др.), программирование на специализированных языках, применяемых в задачах искусственного интеллекта (LISP, FRL, SmallTalk и др.) и др. Четвертый этап разработки экспертных систем в какой-то степени является ключевым, так как здесь происходит создание программного комплекса, демонстрирующего жизнеспособность подхода в целом. Средняя длительность 1-2 месяца.

Этап 5. Тестирование

Прототип проверяется на удобство и адекватность интерфейсов ввода-вывода, эффективность стратегии управления, качество проверочных примеров, корректность базы знаний. Тестирование – это выявление ошибок в выбранном подходе, выявление ошибок в реализации прототипа, а также выработка рекомендаций по доводке системы до промышленного варианта.

Этап 6. Опытная эксплуатация

Проверяется пригодность экспертной системы для конечных пользователей. По результатам этого этапа может потребоваться существенная модификация экспертной системы.

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

2. Взаимодействие пользователя с ЭС.

Конкретная экспертная система создается в результате совместной работы инженера по знаниям и эксперта. Взаимодействие пользователя с ЭС осуществляется через интерфейс пользователя на близком к естественному или профессиональному языку предметной области непроцедурном языке. При этом производится трансляция предложений на язык представления знаний (ЯПЗ) экспертной системы. Описание запроса на ЯПЗ поступает в решатель, в котором на основе знаний из базы выводится решение поставленного запроса в соответствии с некоторой стратегией выбора правил. С помощью подсистемы объяснений производится отображение промежуточных и окончательных выводов, объяснение применяемой мотивировки.

1.7. Лекция №7 (2 часа).

Тема: «Технология разработки экспертных систем»

1.7.1. Вопросы лекции:

1. Технология разработки экспертных систем.
2. Планирование в интеллектуальных системах.
3. Методы поиска решений в ЭС: поиск в пространстве состояний, редукция, дедуктивный вывод.

1.7.2. Краткое содержание вопросов:

1. Технология разработки экспертных систем.

Существует, по крайней мере, четыре значительно отличающихся друг от друга подхода к созданию экспертных систем:

1) Подход, базирующийся на поверхностных знаниях;
2) Структурный подход;
3) Подход, базирующийся на глубинных знаниях;
4) Смешанный подход, базирующийся на использовании поверхностных и глубинных знаний.

1. Подход, базирующийся на поверхностных знаниях

Применяется к сложным задачам, которые не могут быть точно описаны. Этот подход заключается в получении от эксперта фрагментов знаний (часто эвристических), которые релевантны решаемой задаче. При этом не предпринимается никаких попыток систематического или глубинного изучения области, что предопределяет использование поиска в пространстве состояний в качестве универсального механизма вывода. Обычно в экспертных системах, использующих данный подход, в качестве способа представления выбираются правила. Условие каждого правила определяет образец некоторой ситуации, при соблюдении 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 сопоставится ни с одним правилом. Данный подход с успехом применяется к широкому классу приложений, однако он оказывается неэффективным в тех приложениях, когда задача может быть заранее структурирована или при решении задачи может быть использована некоторая модель.

2. Структурный подход

Структурный подход к построению экспертных систем обусловлен тем, что для ряда приложений применение только техники поверхностных знаний не обеспечивает решения задачи. Действительно, использование поиска в качестве механизма вывода в неструктурированной базе знаний может приводить к ненадежным и (или) некачественным решениям. Структурный подход к построению экспертных систем подобен структурному программированию. Однако применительно к экспертным системам речь не идет о том, что структурирование должно довести задачу до алгоритма (как в традиционном программировании), а предполагается, что часть задачи решается с помощью поиска.

Структурный подход в различных приложениях целесообразно сочетать с поверхностным или глубинным.

3. Глубинный подход

В глубинном подходе компетентность экспертной системы базируется на модели той проблемной среды, в которой эта экспертная система работает. Модель может быть определена различными способами (декларативно, процедурно). Необходимость в ряде приложений использовать модели вызвана стремлением исправить несовершенство поверхностного подхода, возникающего при отсутствии правил, удовлетворяющих текущей ситуации в рабочей памяти. Глубинные экспертные системы кроме возможностей поверхностных обладают способностью при возникновении неизвестной ситуации определить с помощью некоторых общих принципов, справедливых для области экспертизы, какие действия следует выполнить.

Глубинный (модельный) подход требует явного описания структуры и взаимоотношений между различными сущностями области. При этом подходе необходимо использовать инструментальные средства, обладающие мощными моделирующими возможностями: объекты с присоединенными процедурами, иерархическое наследование свойств, активные знания (программирование, управляемое данными), передача сообщений объектам (объектно-ориентированное программирование) и т.п.

4. Смешанный подход

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

2. Планирование в интеллектуальных системах.

Плани́рование — оптимальное распределение ресурсов для достижения поставленных целей, деятельность (совокупность процессов), связанная с постановкой целей (задач) и действий в будущем. С точки зрения математики, планирование — это функция, одним из аргументов которой является время.

3. Методы поиска решений в ЭС: поиск в пространстве состояний, редукция, дедуктивный вывод.

Методы решения задач, основанные на сведении 'их к поиску, зависят от особенностей предметной области, в которой решается задача, и от требований, предъявляемых пользователем к решению. Особенности предметной области:

1) Объем пространства, в котором предстоит искать решение;
2) Степень изменяемости области во времени и пространстве (статические и динамические области);
3) Полнота модели, описывающей область, если модель не полна, то для описания области используют несколько моделей, дополняющих друг друга;
4) Определенность данных о решаемой задаче, степень точности (ошибочности) и полноты (неполноты) данных.

Задача поиска в пространстве состояний обычно формулируется в теоретико-графовой интерпретации.

Пусть задана тройка (S_0, F, S_T), где S_0 - множество начальных состояний (условия задачи), F - множество операторов задачи, отображающих одни состояния в другие, S_T - множество конечных (целевых) состояний (решений задачи).

Цель: определять такую последовательность операторов, которая преобразует начальные состояния в конечные.

Процесс решения в виде графа G= (X, Y), где X= {x_0, x_1, ...} - множество (в общем случае бесконечное) вершин графа, состояний, а Y - множество, содержащее пары вершин (x_i, x_j), (x_i, xj) $\in$ X. Если каждая пара (x_i, x_j) не упорядочена, то ее называют ребром, а граф - неориентированным. Если для каждой пары (x_i, x_j) задан порядок

(направление), то пару (x_i, x_j) называют дугой (ориентированным ребром), а граф называют ориентированным (направленным). Вершины пары (x_i, x_j) называют концевыми точками ребра (дуги).

Поиск в пространстве состояний естественно представить в виде ориентированного графа. Наличие пары (x_i, x_j) свидетельствует о существовании некоторого оператора f ($f \in F$), преобразующего состояние, соответствующее вершине xi, в состояние xj. Для некоторой вершины xi выделяем множество всех направленных пар (xi, xj) $\in Y$, т.ь. множество дуг, исходящих из вершины x_i, (родительской вершины), и множество вершин (называемых дочерними вершинами), в которые эти дуги приводят. Множество дуг, исходящих из вершины x_i, соответствует множеству операторов, которые могут быть применены к состоянию, соответствующему вершине x_i.

В множестве вершин X выделяют подмножество вершин $X_0 \subseteq X$, соответствующее множеству начальных состояний (S_o), и подмножество вершин $X_т \subseteq X$, соответствующее множеству конечных (целевых) состояний (S_T). Множество $X_т$ может быть задано как явно, так и неявно, т.е. через свойства, которыми должны обладать целевые состояния.

Отметим, что граф С может быть задан явно и неявно. Неявное задание графа G стоит в определении множества $X_0 \subseteq X$ (соответствующего множеству начальных состояний) и множества операторов, которые, будучи применимы к некоторой вершине графа, дают все ее дочерние вершины.

Итак, граф G задает пространство состояний, т.е. пространство, в котором осуществляется поиск решения. Построение пространства осуществляется с помощью следующего процесса. Берется некая вершина $x_0 \subseteq X$, к ней применяются все возможные операторы, порождающие все дочерние вершины.

Этот процесс называют процессом раскрытия вершин. Если получена целевая вершина, то она не раскрывается. Процесс построения пространства состояний заканчивается, когда все нераскрытые вершины являются целевыми, или терминальными (т.е. вершинами, к которым нельзя применить никаких операторов). В связи с тем, что пространство состояний может содержать бесконечное количество вершин, на практике процесс порождения пространства ограничивают либо временем, либо объемом памяти.

На практике требуется обеспечить полноту поиска, т.е. Организовать поиск так, чтобы все целевые вершины были найдены, если они существуют.

Надежным способом обеспечения полноты является полный перебор всех вершин.

Для задания процесса перебора необходимо определить порядок, в котором будут перебираться вершины графа. Обычно выделяют два основных способа поиска:

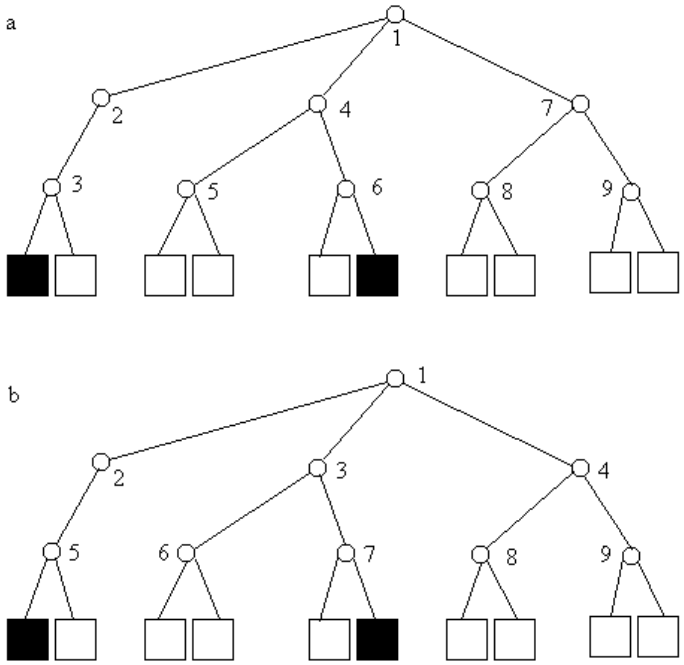


Рис.1. Пространство состояний, построенное поиском в глубину (а) и поиском в ширину (б).

Поиск в глубину (сначала раскрывается та вершина, которая была построена самой последней). Рис.3.1.а поиск в ширину. (Вершины раскрываются в том же порядке, в котором они порождаются.) Рис.3.1.б.

Целевые вершины помечены черными квадратами, а терминальные - белыми квадратами. При использовании каждого из способов могут быть найдены все решения. При переборе всего пространства оба метода будут анализировать одинаковое количество вершин, однако метод поиска в ширину будет требовать существенно больше памяти, так как он запоминает все пути поиска (а не один, как при поиске в глубину).

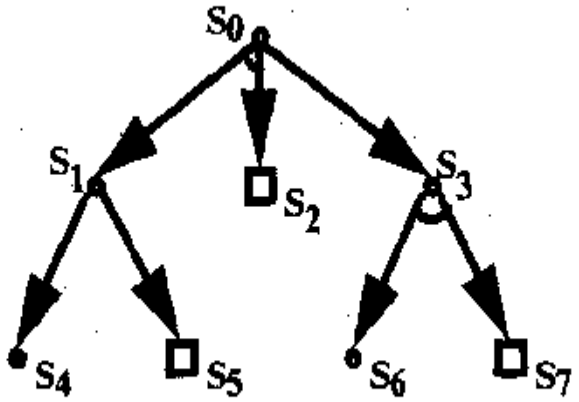
Поиск методом редукции

При поиске методом редукции решение задачи сводится к решению совокупности образующих ее подзадач. Этот процесс повторяется для каждой подзадачи до тех пор, пока каждая из полученного набора подзадач, образующих решение исходной задачи, не будет иметь очевидное решение.

Процесс решения задачи разбиением ее на подзадачи можно представить в виде специального направленного графа G, называемого И/ИЛИ-графом; Каждой вершине этого графа ставится в соответствие описание некоторой задачи (подзадачи). В графе выделяют два типа вершин: конъюнктивные вершины и дизъюнктивные вершины.

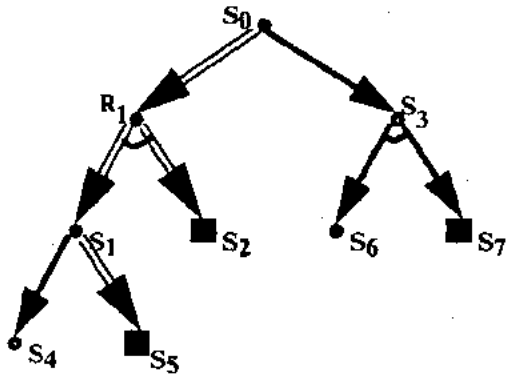
Решение задачи при поиске методом редукции (при поиске в И/ИЛИ-графе) сводится к нахождению в И/ИЛИ-графе решающего графа.

Цель процесса поиска в И/ИЛИ-графе - показать, что начальная вершина разрешима, т.е. для этой вершины существует решающий граф. Определение разрешимой вершины в И/ИЛИ-графе можно сформулировать рекурсивно следующим образом:



Графическое представление процесса разбиения задачи на подзадачи

- Конечные (целевые) вершины разрешимы, так как их решение известно по исходному предположению.
- Вершина ИЛИ разрешима тогда и только тогда, когда разрешима по крайней мере одна из ее дочерних вершин.
- Вершина И разрешима тола и только тогда, когда разрешима каждая из ее дочерних вершин.



Пример И/ИЛИ-графа

Решающий граф определяется как подграф из разрешимых вершин, который показывает, что начальная вершина разрешима (в соответствии с приведенным выше определением). На рис. 3.3. разрешимые вершины зачернены, а неразрешимые оставлены белыми.

Для графа И/ИЛИ, так же как для поиска в пространстве состояний, можно определить поиск в глубину и поиск в ширину как в прямом, так и в обратном направлении. На рис. 3.4, приведен пример поиска в ширину (рис. 3.4. а) и поиска в глубину (рис. 3.4, б). На рисунке вершины пронумерованы в том порядке, в котором они раскрывались, конечные вершины обозначены квадратами, разрешимые вершины зачернены, дуги решающего графа выделены двойными линиями.

1.8. Лекция №8 (2 часа).

Тема: «Нейрокомпьютерные системы».

1.8.1. Вопросы лекции:

1. Понятие нейрокомпьютерной системы.
2. Биологический прототип НС. Модели искусственных нейронов.
3. Персептрон. Обучение персептрона.
4. Многослойный персептрон. Метод обратного распространения ошибки.

1.8.2. Краткое содержание вопросов:

1. Понятие нейрокомпьютерной системы.

В последние десятилетия в мире бурно развивается новое направление ИИ, специализирующаяся на искусственных нейронных сетях (НС).

Нейрокомпьютеры являются предметом исследований сразу нескольких дисциплин, поэтому единое определение нейрокомпьютера можно дать только с учетом различных точек зрения, адекватных разным направлениям науки.

Математическая статистика. Нейрокомпьютеры - это системы, позволяющие сформировать описания характеристик случайных процессов и совокупности случайных процессов, имеющих в отличие от общепринятого, сложные, зачастую неизвестные функции распределения.

Математическая логика и теория автоматов. Нейрокомпьютеры - это системы, в которых алгоритм решения задачи представлен логической сетью элементов частного вида - нейронов с полным отказом от булевских элементов типа И, ИЛИ, НЕ.

Теория управления. В качестве объекта управления выбирается частный случай, хорошо формализуемый объект - многослойная нейронная сеть, а динамический процесс ее настройки представляет собой процесс решения задачи.

Вычислительная математика. В отличие от классических методов решения задач нейрокомпьютеры реализуют алгоритмы решения задач, представленные в виде нейронных сетей.

Что позволяет разрабатывать алгоритмы, потенциально более параллельные, чем любая другая их физическая реализация. Множество нейросетевых алгоритмов решения задач составляет новый перспективный раздел вычислительной математики, условно называемый нейроматематикой.

Вычислительная техника. Нейрокомпьютер - это вычислительная система с архитектурой MSIMD, в которой реализованы два принципиальных технических решения:

- Упрощен до уровня нейрона процессорный элемент однородной структуры и резко усложнены связи между элементами;
- Программирование вычислительной структуры перенесено на изменение весовых связей между процессорными элементами.

Общее определение: Нейрокомпьютерная система - это вычислительная система с архитектурой аппаратного и программного обеспечения, адекватной выполнению алгоритмов, представленных в нейросетевом логическом базисе.

Таким образом нейрокомпьютерная система может быть представлена моделями как программного, так и аппаратного исполнения.

В основе такой системы лежит понятие нейронной сети (НС), которая позволяет решать => виды задач:

автоматизация процессов распознавания образов	адаптивное управление	аппроксимация функционалов	прогнозирование	создание ЭС	другие

Например, с помощью НС можно:

- Предсказывать показатели биржевого рынка,
- Выполнять распознавание оптических или звуковых сигналов,
- Создавать самообучающиеся системы, способные управлять автомашиной при парковке или синтезировать речь по тексту.

Определение: НС – это громадный распределенный параллельный процессор, состоящий из элементарных единиц обработки информации (нейронов), которые накапливают экспериментальные знания и представляют их для последующей обработки.

Существуют мнения, что связь НС с биологией слаба и зачастую несущественна, однако функционирование НС часто напоминает человеческое познание.

Выделим 2 сходства НС с мозгом человека:

- Знания поступают в НС из окружающей среды и используется в процессе обучения;
- Для накопления знаний применяются связи между нейронами, называемые синоптическим весами.

Биологический нейрон

Нервную систему человека можно рассмотреть как 3^x-ступенчатую.

Центром этой системы является мозг, представленный сетью нейронов (нервов). Рецепторы преобразовывают сигналы от тела и из окружающей среды в электрические импульсы, передаваемые в НС (мозг). Эффекторы преобразовывают электрические импульсы, сгенерированные НС в выходные сигналы.

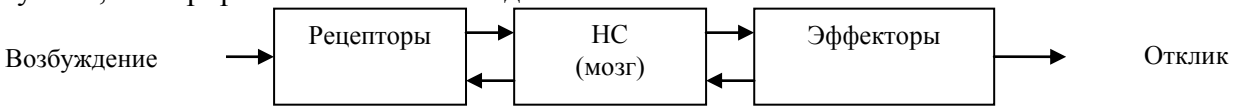


Рисунок 1 – Блочная диаграмма для нервной системы

Существует множество форм и размеров нейронов, в зависимости от того, в какой части мозга они находятся. Самый распространенный тип нейронов коры головного мозга – пирамидальная клетка.

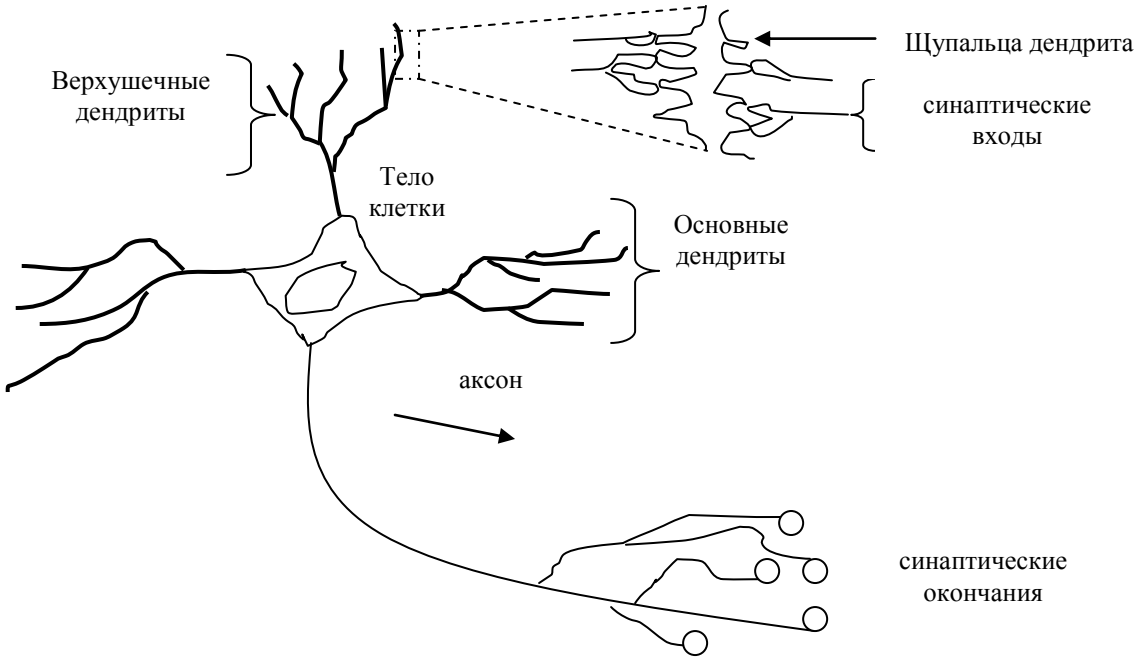


Рисунок 2 - Биологический нейрон

Дендриты идут от тела нервной клетки к другим нейронам, где они принимают сигналы в точках соединения, называемых синапсами. Принятые синапсом входные сигналы подводятся к телу нейрона. Здесь они суммируются, причем одни входы стремятся возбудить нейрон, другие – воспрепятствовать его возбуждению. Когда суммарное возбуждение в теле нейрона превышает некоторый порог, нейрон возбуждается, посылая по аксону сигнал другим нейронам.

Синапсы – это элементарные структурные и функциональные единицы, которые передают импульсы между нейронами.

Аксоны – линии передачи, имеют более гладкую поверхность, тонкие границы и большую длину.

Дендриты – зоны приема, имеют неровную поверхность с множеством окончаний (название дано из-за сходства с деревом).

Модели искусственных нейронов

Рассмотрим нелинейную модель искусственного нейрона, лежащую в основе НС.

Основу каждой НС составляют простые, в большинстве случаев – однотипные, элементы (ячейки), имитирующие работу нейронов мозга.

Каждый нейрон характеризуется своим текущим состоянием по аналогии с нервными клетками головного мозга, которые могут быть возбуждены или заторможены. Он обладает группой синапсов – однонаправленных входных связей, соединенных с выходами других нейронов, а также имеет аксон – выходную связь данного нейрона, с которой сигнал (возбуждения или торможения) поступает на синапсы следующих нейронов.

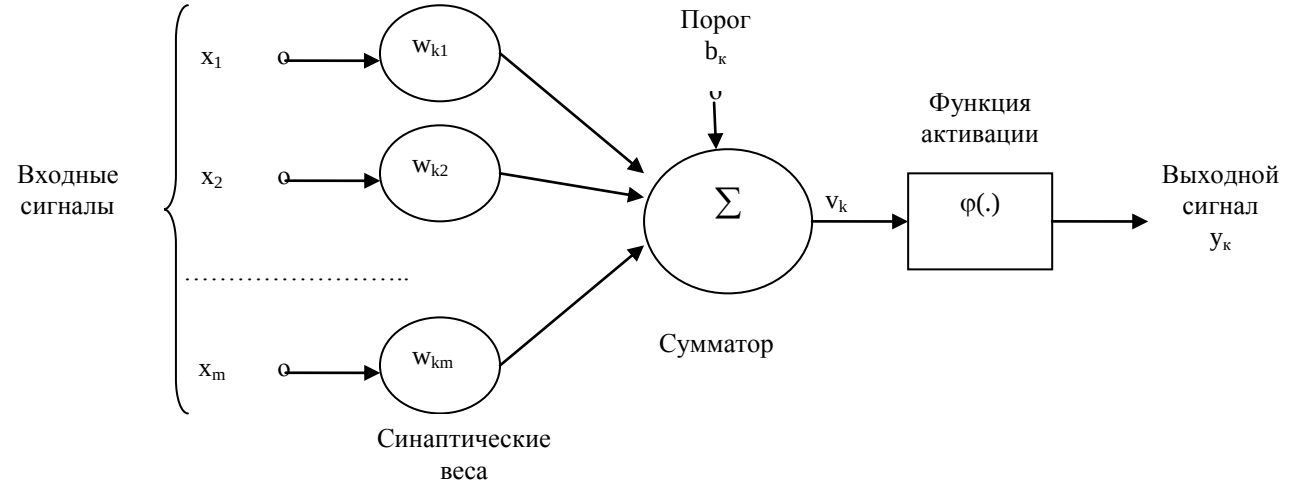


Рисунок 3 – Модель искусственного нейрона

В этой модели можно выделить 3 основных элемента.

1. Набор *синапсов* или *связей*, каждый из которых характеризуется своим *весом* или *силой*.

В частности, сигнал x_j на входе синапса j, связанного с нейроном k , умножается на вес w_{kj}, где индексы относятся: k – рассматриваемому нейрону, j – ко входному окончанию синапса, с которым связан данный вес.

В отличие от синапсов мозга в данном случае вес может принимать как «+», так и «-» значения.

2. *Сумматор* – складывает входные сигналы, взвешенные относительно соответствующих синапсов нейрона. Описывается в виде линейной комбинации.

3. *Функция активации* (*сжатия*) ограничивает амплитуду выходного сигнала нейрона. Обычно нормализованный диапазон амплитуд выхода нейрона лежит в интервале [0,1] или [-1,1].

Кроме того, в модель включен пороговый элемент – эта величина отражает увеличение или уменьшение входного сигнала, подаваемого на функцию активации.

В мат. представлении функц-е нейрона k можно описать => парой урав-й:

$$u_k = \sum_{j=1}^{m} w_{kj} \cdot x_j \ , \tag{1}$$

$$y_k = \varphi(u_k + b_k), \tag{2}$$

где $x_1, x_2, \ldots, x_m$ – входные сигналы;

$w_{k1}, w_{k2}, \ldots, w_{km}$ – синаптические веса нейрона k;

u_k – линейная комбинация входных воздействий;

$\varphi(\cdot)$ – функция активации;

y_k – выходной сигнал нейрона.

В модели рис.3 *постсинаптический потенциал* или *индуцированное локальное поле* или *потенциал активации* обозначен v_k для нейрона k зависит от того, какое зн-е принимает порог b_k «+» или «-» и вычисляется =>

$$v_k = u_k + b_k . \tag{3}$$

Порог b_k - внешний параметр искусственного нейрона k. Принимая во внимание (3), уравнения (1) и (2) можно преобразовать к => виду:

$$v_k = \sum_{j=0}^{m} w_{kj} \cdot x_j , \tag{4}$$

$$y_k = \varphi(v_k), \tag{5}$$

В выражении (4) появился новый синапс, входной сигнал которого фиксирован, т.е. x_0=+1, а его вес w_{k0}= b_k.

Типы функций активации

Функции активации, представленные в формулах как $\varphi(v)$, определяют выходной сигнал нейрона в зависимости от индуцированного локального поля v. Можно выделить 3 основных типа функций активации.

1. *Функция единичного скачка*, или пороговая функция

$$\varphi(v) = \begin{cases} 1, если\ v \geq 0; \\ 0, если\ v < 0 \end{cases}, \tag{6}$$

В технической литературе эта форма (6) наз-ся *функцией Хэвисайда*.

Соответственно выходной сигнал нейрона k такой функции можно представить как

$$y_k = \begin{cases} 1, если\ v_k \geq 0; \\ 0, если\ v_k < 0 \end{cases}, \text{ где} \tag{7}$$

$v_k = \sum_{j=1}^{m} w_{kj} \cdot x_j + b_k$ - индуцированное локальное поле нейрона

Модель (7) наз-ся *моделью Мак-Каллока-Питца*.

2. *Кусочно – линейная функция*, описывается так

$$\varphi(v) = \begin{cases} 1, если\ v \geq +\dfrac{1}{2} \\ |v|, если\ -\dfrac{1}{2} < v < +\dfrac{1}{2}, \\ 0, если\ v \leq -\dfrac{1}{2} \end{cases} \tag{8}$$

где коэффициент усиления в линейной области оператора предполагается =1. Эту функцию активации м. р-ть как *аппроксимацию нелинейного усилителя*.

3. *Сигмоидальная функция* – самая распространенная ф-я, исп-я для создания НС, она быстро возрастает и поддерживает баланс м\у линейным и нелинейным поведением.

Примером сигмоидальной ф-ции м. служить логистическая ф-ция, задаваемая => выражением

$$\varphi(v) = \frac{1}{1 + \exp(-av)}, \tag{9}$$

где a – *параметр наклона* сигмоидальной ф-ции.

Преимущества сигмоида по отношению к пороговой ф-ции:

- принимает бесконечное множество значений в диапазоне от 0 до 1 (пороговая – только значения 0 и 1);
- когда *a* достигает ∞, сигмоидальная ф-я вырождается в пороговую;

- сигмоидальная функция является дифференцируемой на всей оси абсцисс, что используется в некоторых алгоритмах обучения (пороговая нет),

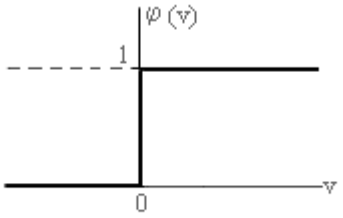
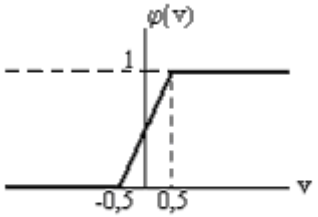
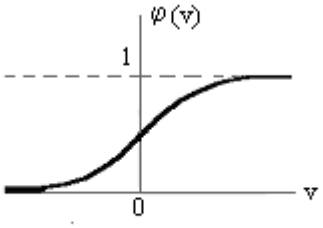
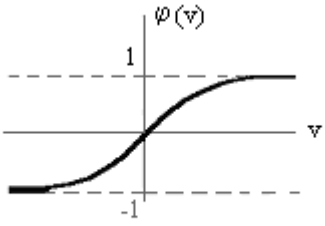
$$\varphi'(v) = \alpha \cdot \varphi(v) \cdot (1 - \varphi(v)).$$

Замечание: Область знаний функций активации, определенных формулами (6), (8) и (9), представляет собой отрезок [0, 1]. Однако иногда требуется функция активации с обл. зн-й [-1, 1]. Тогда пороговую функцию м. определить =>

$$\varphi(v) = \begin{cases} 1, если\ v > 0 \\ 0, если\ v = 0 \\ -1, если\ v < 0 \end{cases} \quad (10)$$

Эта функция обычно называются *сигнум*. В данном случае сигмоидальная функция будет иметь форму гиперболического тангенса.

Графическое представление типов ф-ций активации

Функция единичного скачка	Кусочно – линейная функция
	
Сигмоидальная функция или сигмоид	Гиперболическая форма сигмоида -сигнум
	

3. Персептрон. Обучение персептрона.

Персептрон – простейшая форма НС, предназначенная для классификации линейно-разделимых сигналов. Состоит из одного нейрона с настраиваемыми синхронными весами и порогом (модель нейрона Мак – Каллока – Питца). Предложил Розенблатт.

Персептрон обучается с учителем. Это означает, что д\б задано *обучающее множество*, т.е. мн-во пар векторов $\{x(n), d(n)\}$, где

$\{x(n)\} = \{+1, x_1(n), x_2(n), ..., x_m(n)\}$ - формализованное условие задачи (вектор-строка размерности $m+1$),

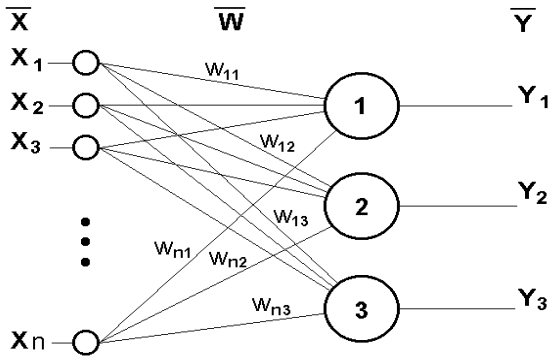
$\{d(n)\} = \{d_1(n), d_2(n), ..., d_m(n)\}$ - желаемый отклик (известное решение для этого условия),

m – кол-во эл-в в обучающем мн-ве.

Задача обучения персептрона ставится так: подобрать такие зн-я параметров сети, чтобы ошибка была *min* для данного обучающего множества $\{x(n), d(n)\}$.

Р-м трехнейронный персептрон, т.е. такую сеть, нейроны кот, имеют активационную ф-цию в виде ед. скачка.

На *m* входов поступают некие сигналы, проходящие по синапсам на 3 нейрона, образующие ед. слой этой НС и выдающие 3 выходных сигнала:



$$y_j = \varphi\left[\sum_{i=1}^{m} x_i \cdot w_{ij}\right], \qquad j=1...3 \qquad (4.1)$$

Очевидно, что все весовые коэф-ты синапсов одного слоя нейронов м. свести в матрицу ***W***, в кот, каждый эл-т w_{ij} задает величину i-ой синап-ой связи j-ого нейрона.

Т.о., процесс, происходящий в НС, м\б записан в матричной форме:

$$Y = \varphi(XW), \qquad (4.2)$$

где ***X*** и ***Y*** – соответственно входные и выходные сигнальные векторы,

$\varphi(u)$ – активационная функция, применяемая поэлементно к компонентам вектора ***U***.

Предложенный Ф.Розенблаттом метод обучения состоит в итерационной подстройке матрицы весов, последовательно уменьшающей ошибку в выходных векторах.

Алгоритм обучения персептрона

Шаг 0	Проинициализировать эл-ты весовой м-цы (небол. случайными зн-ями).
Шаг 1	Подать на входы один из входных векторов, кот. сеть д. научиться различать, и вычислить ее выход.
Шаг 2	Вычисляется вектор ошибки по (15), делаемой сетью на выходе. Дальнейшая идея состоит в том, что изменение вектора весовых коэф-тов в области малых ошибок д\б пропорционально ошибке на выходе и =0, если ошибка =0.
Шаг 3	Вектор весов модифицируется по формуле (17)
Шаг 4	Шаги 1—3 повторяются для всех обучающих векторов. Один цикл послед-ного предъявления всей выборки называется **эпохой.** Обучение завершается по истечении нескольких эпох: а) когда итерации сойдутся, т.е. вектор весов перестает изменяться, или б) когда полная, просум-я по всем векторам абс. ошибка станет < некот. малого зн-я.

Подробное объяснение алгоритма

Подаем на вход *персептрона* такой вектор x, для кот. уже известен правильный ответ. Если выходной сигнал *персептрона* совпадает с правильным ответом, то никаких действий предпринимать не надо. В случае ошибки, необходимо обучить *персептрон* правильно решать данный пример. Ошибки м\б 2 типов.

1 тип ошибки: на выходе *персептрона* — 0, а правильный ответ — 1.

Для того чтобы *персептрон* выдавал правильный ответ, необ-мо, чтобы сумма $\sum_{i=1}^{m} x_i \cdot w_{ij}$ стала больше. Поскольку переменные принимают значения 0 или 1, ув-е суммы м\б достигнуто за счет увеличения весов w_i. Однако нет смысла увеличивать веса при переменных x_i, которые равны нулю. Т.о., следует ув-ть веса w_i при тех переменных x_i, кот. равны 1.

Первое правило. Если на выходе *персептрона* получен 0, а правильный ответ =1, то необ-мо ув-ть веса связей м\у одновременно активными нейронами. При этом выходной *персептрон* считается активным.

2 тип ошибки: на выходе *персептрона* — 1, а правильный ответ =0. Для обучения правильному реш-ю данного примера следует ум-ть $\sum_{i=1}^{m} x_i \cdot w_{ij}$.

След-но, необ-мо ум-ть веса связей w_i при тех переменных, кот. =1 (т.к. нет смысла ум-ть веса связей при равных нулю переменных~ x_i). Необ-мо также провести эту процедуру для всех активных нейронов предыдущих слоев. В результате получаем второе правило.

Второе правило. Если на выходе *персептрона* получена единица, а правильный ответ =0, то необ-мо ум-ть веса связей между одновременно активными нейронами.

Т.о., процедура обучения сводится к последовательному перебору всех примеров обучающего мн-ва с применением правил обучения для ошибочно решенных примеров. Если после очередного цикла предъявления всех примеров окажется, что все они решены правильно, то процедура обучения завершается.

Нерассмотренными остались два вопроса:

1 - о сходимости процедуры обучения

2 - на сколько нужно ув-ть (ум-ть) веса связей при применении правил обучения.

Ответ на первый вопрос дают следующие теоремы.

Теорема о сходимости персептрона.

Если существует вектор параметров *w*, при котором *персептрон* правильно решает все примеры обучающей выборки, то при обучении *персептрона* по вышеописанному алгоритму решение будет найдено за конечное число шагов.

Теорема о "зацикливании" персептрона.

Если не существует вектора параметров w, при котором *персептрон* правильно решает все примеры обучающей выборки, то при обучении *персептрона* по данному правилу через конечное число шагов вектор весов начнет повторяться.

4. Многослойный персептрон. Метод обратного распространения ошибки.

Проблемы, связанные с использованием однослойного персептрона

Р-м проблемы, которые остались открытыми после работ Ф.Розенблатта. Часть из них была впоследствии решена, некоторые остались без полного теоретического решения.

1. Практическая проверка условия линейной разделимости множеств.
2. Сколько шагов потребуется при итерационном обучении?
3. Как влияет на обучение последовательность предъявления образов в течение эпохи обучения?
4. Имеет ли вообще -правило преимущества перед простым перебором весов, т.е. является ли оно конструктивным алгоритмом быстрого обучения?
5. Каким будет качество обучения, если обучающая выборка содержит *не все* возможные пары векторов? Какими будут ответы персептрона на *новые* вектора?

Персептрон – простейшая форма НС, предназначенная для классификации линейно-разделимых сигналов.

Доказанная Розенблаттом теорема о сходимости обучения по -правилу говорит о том, что персептрон способен обучится любому обучающему набору, который *он способен представить*.

Линейная разделимость и персептронная представляемость

Каждый нейрон персептрона яв-ся формальным пороговым эл-м, принимающим ед. зн-я в случае, если суммарный взвешенный вход больше некоторого порогового значения:

$$y_j = \begin{cases} 1, & \sum_i W_{ij} x_i > \theta_j \\ 0, & \sum_i W_{ij} x_i \le \theta_j \end{cases}$$

Таким образом, при заданных значениях весов и порогов, нейрон имеет определенное значение выходной активности для каждого возможного вектора входов. Мн-во вх, векторов, при кот, нейрон активен (y=1), отделено от мн-ва векторов, на кот, нейрон пассивен (y=0) *гиперплоскостью*, уравнение которой есть, суть:

$$\sum_i W_{ij} x_i - \theta_j = 0$$

Следовательно, нейрон способен *отделить* (иметь различный выход) только такие два мн-ва векторов входов, для кот, имеется гиперплоскость, отсекающая одно мн-во от другого. Такие мн-ва наз-ют *линейно разделимыми*.

Р-м это понятие на пр-ре.

Пусть имеется нейрон, для которого вх, вектор содержит только две булевые компоненты (x_1, x_2), определяющие плоскость. На данной плоскости возможные значения векторов отвечают вершинам единичного квадрата. В каждой вершине определено *требуемое* значение активности нейрона 0 (на рис. - белая точка) или 1 (черная точка). Требуется определить, существует ли такой набор весов и порогов нейрона, при кот, этот нейрон сможет отделить точки разного цвета?

На рис 4.2 представлена одна из ситуаций, когда этого сделать *нельзя* вследствие линейной неразделимости множеств белых и черных точек.

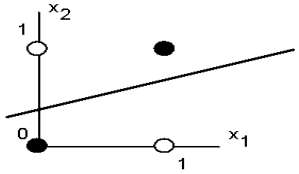


Рис. 4.2. Белые точки не могут быть отделены одной прямой от черных.

Требуемая активность нейрона для этого рисунка определяется таблицей, в которой не трудно узнать задание логической функции “исключающее или”.

X 1	X 2	
0	0	
1	0	
0	1	
1	1	

Линейная неразделимость множеств аргументов, отвечающих различным значениям функции означает, что функция “исключающее или”, столь широко использующаяся в логических устройствах, не м\б представлена формальным нейроном.

Столь скромные возможности нейрона и послужили основой для критики персептронного направления Ф.Розенблатта со стороны М.Минского и С.Пейперта.

При увел-и числа аргументов *относительное* число ф-ций, кот, обладают свойством лин., раздел-ти резко уменьшается. А значит и резко сужается класс функций,

который может быть реализован персептроном. Соответствующие данные приведены в => таблице:

Число переменных N	Полное число возм. логических ф-ций ($= 2^{2^N}$)	Из них лин. разделимых ф-ций
1	4	4
2	16	14
3	256	104
4	65536	1882
5	> 1000000000	94572

В начале 70-х годов, это ограничение было преодолено путем введения нескольких слоев нейронов.

Особенности строения биологических сетей подталкивают исследователя к использованию более сложных, и в частности, иерархических архитектур.

На возможность построения таких архитектур указал еще Ф.Розенблатт, однако им не была решена проблема обучения.

Многослойный персептрон представляет собой сеть, состоящую из нескольких последовательно соединенных слоев формальных нейронов МакКаллока и Питтса.

Обучение многослойного персептрона

Проблема обучения многослойного персептрона

Возникает вопрос - почему для обучения многослойного персептрона нельзя применить уже известное дельта-правило Розенблатта?

Ответ состоит в том, что для прим-я метода Розенблатта необходимо знать не только тек. выходы нейронов, но и требуемые *правильные* зн-я.

В случае многослойной сети эти правильные значения имеются только для нейронов *выходного* слоя. Требуемые значения выходов для нейронов скрытых слоев неизвестны, что и ограничивает применение дельта-правила.

Возможные варианты решения проблемы

1. Разработка наборов выходных сигналов, соответствующих входным, для каждого слоя НС, что, конечно, является очень трудоемкой операцией и не всегда осуществимо.

2. Динамическая подстройка весовых коэффициентов синапсов, в ходе которых выбираются, как правило, наиболее слабые связи и изменяются на малую величину в ту или иную сторону, а сохраняются только те изменения, которые повлекли умен-е ошибки на выходе всей сети. Очевидно, что данный метод "тыка", несмотря на свою кажущуюся простоту, требует громоздких рутинных вычислений.

3. Распределение сигналов ошибки от выходов НС к ее входам, в направлении, обратном прямому распределению сигналов в обычном режиме работы. Так называемый метод ОРО (обработка распределения ошибки).

Т.к. вывод алгоритма ОРО слишком громоздок, то для упрощения понимания мат. выкладок р-м => понятия и обоз-я:

1. Пусть *E(w)*- непр-но диф-мая функция стоимости, зависящая от некоторого неизвестного вектора весовых коэффициентов, тогда $\nabla E(w) = \left[\frac{\partial E}{\partial w_1}, \frac{\partial E}{\partial w_2}, ..., \frac{\partial E}{\partial w_m}\right]^T$ - *вектор градиента* функции стоимости.

2. Под *функциональным сигналом* будем понимать – входной сигнал (стимул), поступающий в сеть и передаваемый вперед от нейрона к нейрону по всей сети. Такой сигнал достигает конца сети в виде выходного сигнала.

Составим таблицу используемых обозначений.

Таблица 1 - Используемые обозначения

Обоз-я	Действия
i, *j* и *k*	Будем относить к различным нейронам сети. Когда сигнал проходит по сети слева направо, считается, что нейрон *j* нах-ся на один слой правее нейрона *i*, а нейрон *k* - еще на один слой правее нейрона *j*, если последний принадлежит скр. слою.
n	Номер итерации, соотв-ет *n* – му обучающему образу(примеру), поданному на вход сети.
E(n)	Тек.сумма квадратов ошибок (или энергия ошибки) на итерации *n*.
E_{av}	Ср. зн-е *E(n)* по всем зн-ям *n* (т.е. по всему обуч. мн-ву) наз-ся ср. энергией ошибки.
$e_j(n)$	Описывает сигнал ошибки на выходе нейрона *j* на итерации *n*.
$d_j(n)$	Желаемый отклик нейрона *j*.
$y_j(n)$	Описывает функциональный сигнал, генерируемый на выходе нейрона *j* на итерации *n*.
$w_{ji}(n)$	Синап.вес, связывающий выход нейрона *i* со входом нейрона *j* на итерации *n*.
$\Delta w_{ji}(n)$	Коррекция, применяемая к весу $w_{ji}(n)$ на шаге *n*.
$v_j(n)$	Индуцированное локальное поле(т.е. взвешенная сумма всех син.входов + порог) нейрона *j* на итерации *n*. Это зн-е передается в ф-цию активации, связанную с нейроном *j*.
$\varphi_j(\cdot)$	Ф-ция активации, соответ-щая нейрону *j* и описывающая нелин. взаимосвязь вх. и вых. сигналов этого нейрона.
$x_i(n)$	*i*–й эл-т вх. вектора.
$o_k(n)$	*k*–й эл-т вых.вектора.
m_l	Размерность (кол-во узлов) слоя *l* многосл. персептрона; *l* =1,2,…,L, где L – глубина сети. Т.о. m_0 - размерность вх.слоя; m_1 - размерность $1^{го}$ скрытого слоя; m_L - размерность вых.слоя.

Метод обратного распространения ошибок

Известно:

- что сигнал ошибки вых.нейрона j на итерации n (соотв-щей n – му пр-ру обучения) опр-ся соотн-ем:

$$e_j(n) = d_j(n) - y_j(n). \tag{6.1}$$

- тек.зн-е энергии ошибки нейрона j определяется как $\frac{1}{2}e_j^2(n)$. Соответственно тек.зн-е общей энергии ошибки сети выч-ся путем сложения этих величин по всем нейронам выходного слоя, т.к. это видимые нейроны, то м. записать:

$$E(n) = \frac{1}{2}\sum_{j \in C} e_j^2(n), \tag{6.2}$$

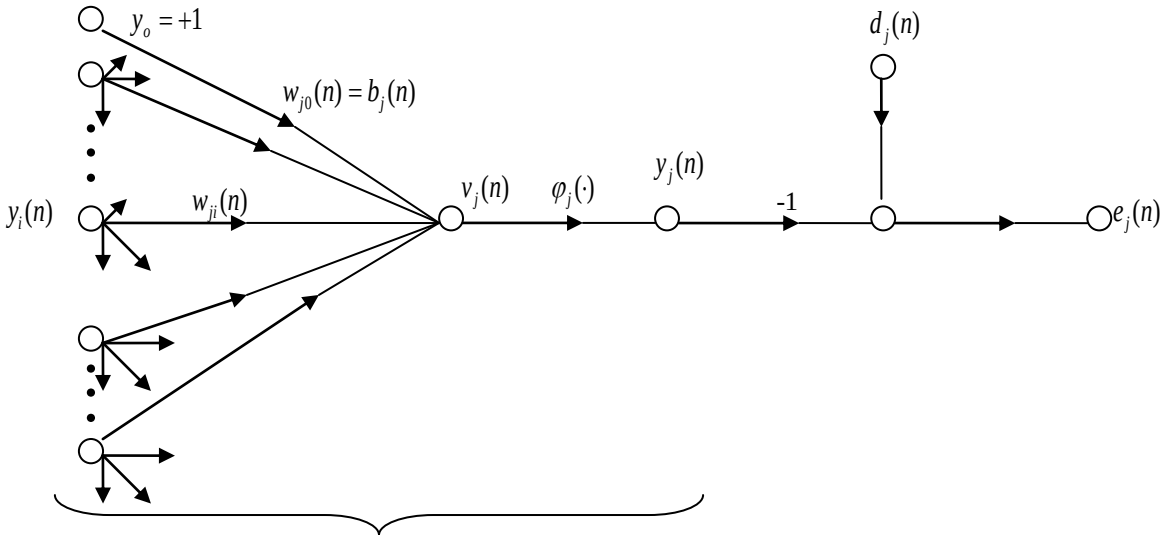
где множество С включает все нейроны выходного слоя НС.

Пусть N – общее число образов в обученном множестве (т.е. мощность этого множества), тогда энергия среднеквадратической ошибки вычисляется как нормализованная по N сумма всех зн-й энергии ошибки $E(n)$:

$$E_{av}(n) = \frac{1}{N}\sum_{n=1}^{N} E(n). \tag{6.3}$$

Для данного множества энергия E_{av} – представляет собой функцию стоимости – меру эффективности обучения. А значит цель прочеса обучения многослойного персептрона является настройка свободных параметров НС с минимизации величины E_{av}.

Р-м граф передачи сигнала в пределах некоторого нейрона j.



На данном рис. изображен нейрон j, на который поступает поток сигналов от нейронов, расположенных в предыдущем слое, тогда инд. лок. поле, полученное на входе ф-и активации, связанной с данным нейроном будет:

$$v_j(n) = \sum_{i=0}^{m} w_{ji}(n) y_i(n), \tag{6.4}$$

где m – общее число входов(за искл. порога) нейрона j.

Функциональный сигнал на выходе нейрона j на итерации n будет:

$$y_j(n) = \varphi_j(v_j(n)). \tag{6.5}$$

Тогда алгоритм ОРО состоит в применении к син.весу $w_{ji}(n)$ коррекции $\Delta w_{ji}(n)$, пропорциональной частной производной $\frac{\partial E(n)}{\partial w_{ji}(n)}$.

В соотв- и с правилом цепочки, градиент м. представить =>

$$\frac{\partial E(n)}{\partial w_{ji}(n)} = \frac{\partial E(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial w_{ji}(n)}. \quad (6.6)$$

Диф-я обе части ур-я (6.2) по $e_j(n)$, получаем:

$$\frac{\partial E(n)}{\partial e_j(n)} = e_j(n). \quad (6.7)$$

Диф-я обе части ур-я (6.1)по $y_j(n)$, получим:

$$\frac{\partial e_j(n)}{\partial y_j(n)} = -1. \quad (6.8)$$

Диф-я обе части ур-я (6.5)по $v_j(n)$, получим:

$$\frac{\partial y_j(n)}{\partial v_j(n)} = \varphi_j'(v_j(n)), \quad (6.9)$$

где штрих справа от имени ф-ции обоз-ет диф-е по аргументу.

И, наконец, диф-я обе части ур-я (6.4)по $w_{ji}(n)$, получим:

$$\frac{\partial v_j(n)}{\partial w_{ji}(n)} = y_i(n). \quad (6.10)$$

Подставим результаты (6.7)-(6.10)в выражение (6.6), окончательно получим:

$$\frac{\partial E(n)}{\partial w_{ji}(n)} = -e_j(n)\varphi_j'(v_j(n))y_i(n). \quad (6.11)$$

Согласно дельта-правила коррекция весовых коэф-в опр-ся:

$$\Delta w_{ji}(n) = -\eta \frac{\partial E(n)}{\partial w_{ji}(n)}. \quad (6.12)$$

Исп-е знака «-» в (6.12) связано с реализацией градиентного спуска в пр-ве весов.

След-но, подставляя (6.11) в (6.12), получим:

$$\Delta w_{ji}(n) = \eta \delta_j(n) y_i(n), \quad (6.13)$$

где $\delta_j(n)$ - локальный градиент, кот. указ-ет на требуемое измен-е син.веса и выч-ся:

$$\delta_j(n) = -\frac{\partial E(n)}{\partial v_j(n)} = e_j(n)\varphi_j'(v_j(n)) \quad (6.14)$$

1.9. Лекция №9 (2 часа).

Тема: «Генетические алгоритмы и моделирование биологической эволюции».

1.9.1. Вопросы лекции:

1. Основные понятия, принципы и предпосылки генетических алгоритмов.
2. Пример работы простого генетического алгоритма

1.9.2. Краткое содержание вопросов:

1. Основные понятия, принципы и предпосылки генетических алгоритмов.

Генетические Алгоритмы (ГА) *– это адаптивные методы функциональной оптимизации, основанные на компьютерном* ***имитационном моделировании биологической эволюции***.

Подобно НС ГА основывается на свойствах биологического прототипа.

Основные принципы ГА были сформулированы Голландом (Holland, 1975), и хорошо описаны во многих работах и на ряде сайтов в Internet.

В настоящее время существует ряд теорий биологической эволюции (Ж.-Б.Ламарка, П.Тейяра де Шардена, К.Э.Бэра, Л.С.Берга, А.А.Любищева, С.В.Мейена и др.), однако, ни одна из них не считается общепризнанной. Наиболее известной и популярной,

конечно, является теория Чарльза Дарвина, которую он представил в работе "Происхождение Видов" в 1859 году.

Эта теория, как и другие, содержит довольно много ***нерешенных проблем***, отметим некоторые наиболее известные из них:

1.Как это ни парадоксально, но несмотря на то, что сам Чарльз Дарвин назвал свою работу "Происхождение Видов" но как раз именно *происхождения видов* она и не объясняет. Дело в том, что возникновение нового вида "по алгоритму Дарвина" является крайне маловероятным событием, т.к. для этого требуется случайное возникновение в одной точке пространства и времени сразу не менее 100 особей нового вида, т.е. особей, которые могли бы иметь плодовитое потомство. При меньшем кол-ве особей вид обречен на вымирание. Поэтому процесс видообразования на основе ***случайных*** мутаций д\б бы занять несуразно много времени.

2. Кроме того, "алгоритм Дарвина" не объясняет явной ***системности*** в многообразии возникающих форм.

3. Кроме того, Дарвин не смог показать ***механизм наследования***, при кот. поддерж-ся и закрепляется изменчивость.

Не смотря на свои недостатки, ***именно теория Дарвина традиционно и моделируется в ГА***, хотя, конечно, это не исключает возможности моделирования и других теорий эволюции в ГА.

Теория Дарвина применима не к отд. особям, а к ***популяциям –*** бол. кол-ву особей ***одного вида***, т.е. способных давать плодовитое потомство, находящейся в определенной статичной или динамичной вн. среде.

В основе модели эволюции Дарвина лежат случайные ***изменения*** отдельных материальных эл-в живого организма при переходе от поколения к поколению. Целесообразные изменения, которые облегчают выживание и производство потомков в данной конкретной внешней среде, сохраняются и передаются потомству, т.е. ***наследуются.*** Особи, не имеющие соответствующих приспособлений, погибают, не оставив потомства или оставив его меньше, чем приспособленные (считается, что количество потомства пропорционально степени приспособленности). Поэтому в рез-те ***естественного отбора*** возникает популяция из наиболее приспособленных особей, которая может стать основой нового вида.

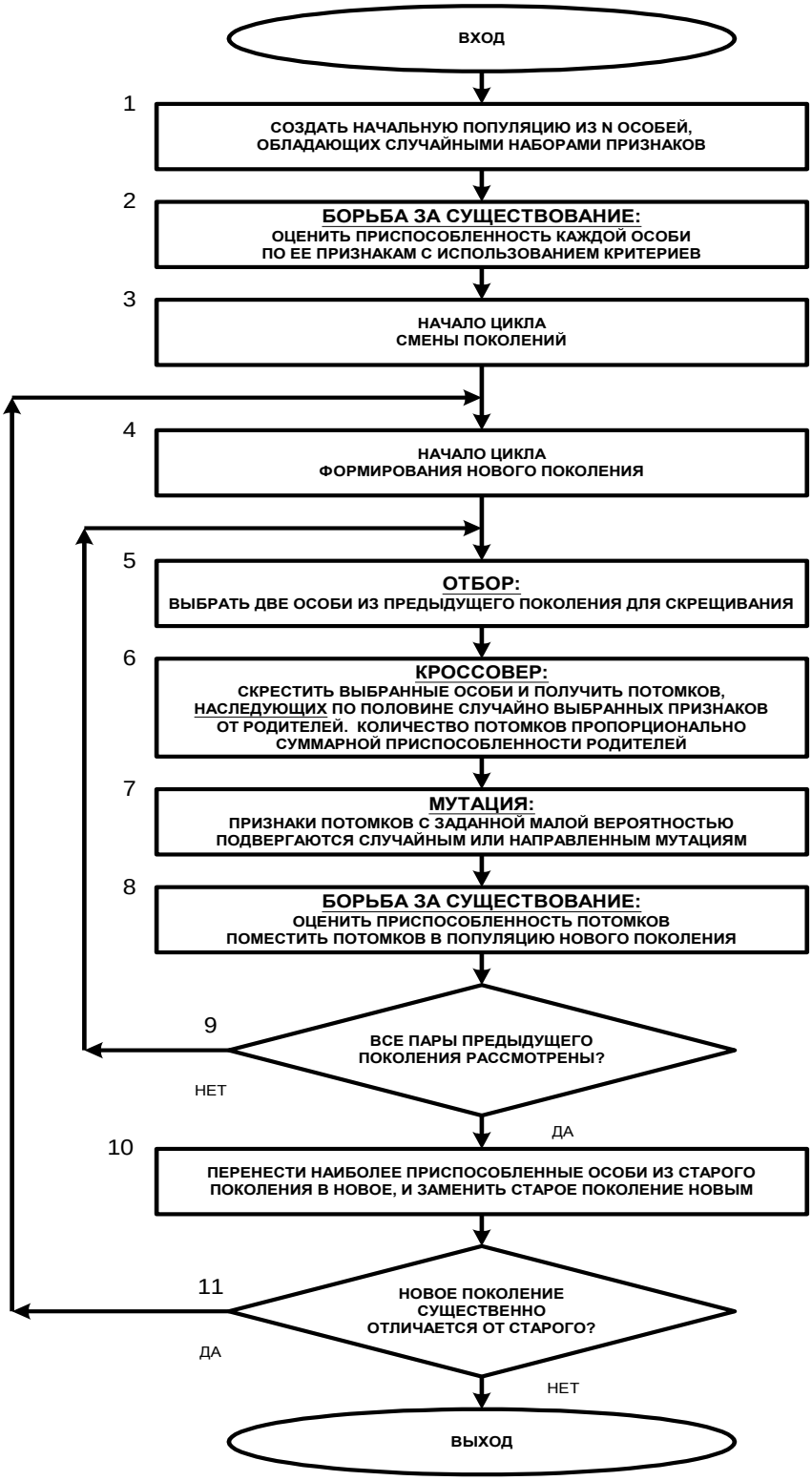
Естественный отбор происходит в условиях ***конкуренции*** особей популяции, а иногда и различных видов, друг с другом за различные ***ресурсы***, такие, н-р, как пища или вода. Кроме того, члены популяции одного вида часто конкурируют за привлечение брачного партнера. Те особи, которые наиболее приспособлены к окружающим условиям, будут иметь относительно больше шансов воспроизвести потомков. Слабо приспособленные особи либо совсем не произведут потомства, либо их потомство будет очень немногочисленным. Это означает, что гены от высоко адаптированных или приспособленных особей будут распределятся в увеличиваться количестве потомков на каждом последующем поколении.

Таким образом, каждый конкретный **ГА** ***представляют <u>имитационную модель</u> некоторой определенной теории биологичкой эволюции или ее варианта.***

Вместе с тем необходимо отметить, что сами исследователи биологической эволюции пока еще не до конца определились с критериями и методами определения степени существенности для поддерживаемой ими теории эволюции тех или иных биологических процессов, которые собственно и моделируются в генетических алгоритмах.

2 Пример работы простого генетического алгоритма

На рисунке 85 приведен пример простого генетического алгоритма.



Простой генетический алгоритм

Работа ГА представляет собой итер-ый процесс, который продолжается до тех пор, пока поколения не перестанут существенно отличаться друг от друга, или не пройдет заданное кол-во поколений или заданное время. Для каждого поколения реализуются отбор, кроссовер (скрещивание) и мутация. Рас-м этот алгоритм.

Шаг 1: генерируется начальная популяция, состоящая из ***N*** особей со случайными наборами признаков.

Шаг 2 (борьба за существование): вычисляется *абсолютная приспособленность* каждой особи популяции к условиям среды *f(i)* и суммарная приспособленность особей

популяции, характеризующая приспособленность всей популяции. Затем при ***пропорциональном отборе*** для каждой особи вычисляется ее *относительный вклад в суммарную приспособленность популяции* $P_s(i)$, т.е. относительные ее абсолютные приспособленности *f(i)* к суммарной приспособленности всех особей популяции (1):

$$P_s(i) = \frac{f(i)}{\sum_{i=1}^{N} f(i)} \quad (1)$$

В выражении (1) сразу обращает на себя внимание возможность сравнения абсолютной приспособленности *i-й* особи *f(i)* не с суммарной приспособленностью всех особей популяции, а со средней абсолютной приспособленностью особ и популяции (2):

$$\bar{f} = \frac{1}{N}\sum_{i=1}^{N} f(i) \quad (2)$$

Тогда получим (3):

$$P(i) = \frac{f(i)}{\bar{f}} = \frac{f(i)}{\frac{1}{N}\sum_{i=1}^{N} f(i)} \quad (3)$$

Если взять логарифм по основанию 2 от выражения (3), то получим ***количество информации, содержащееся в признаках особи о том, что она выживет и даст потомство*** (4).

$$I(i) = Log_2 \frac{f(i)}{\bar{f}} \quad (4)$$

Формально ***приспособленность особи представляет собой количество инф-и, содержащееся в ее фенотипе о продолжении ее генотипа в последующих поколениях.*** Т.к. кол-во потомства особи пропорционально ее приспособленности, то м. считать, что *если это кол-во инф-и:*

– *положительно,* то данная особь выживает и дает потомство, численность кот, пропорциональна этому количеству информации;

– *равно нулю,* то особь доживает до половозрелого возраста, но потомства не дает (его численность равна нулю);

– *меньше нуля,* то особь погибает до достижения половозрелого возраста.

2. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ ЛАБОРАТОРНЫХ РАБОТ

2.1 Лабораторная работа №1 (2 часа).

Тема: «Представление знаний в интеллектуальных системах. Методы работы со знаниями**»**

2.1.1 Цель работы: изучить представление знаний в интеллектуальных системах, изучить методы работы со знаниями

2.1.2 Задачи работы:

1. Проведите анализ представленных определений искусственного интеллекта.
2. Какие сложные задачи решает искусственный интеллект?
3. Представьте определение СИИ.

2.1.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. ПЭВМ

2.1.4 Описание (ход) работы:

Продукционные правила – это правила, имеющие форму: ЕСЛИ «Условие» – ТО «Событие». Продукционные правила описывают знания в виде взаимосвязей типа: «причина» – «следствие», «явление» – «реакция», «признак» – «факт» и.т.п. Конкретизация продукционных правил меняется в зависимости от сущности представляемых знаний.

Например:

- ЕСЛИ «Температура в реакторе превышает 120 ° С » ТО «Снизить подачу топлива на 5%»;
- ЕСЛИ «Вышел из строя вентилятор кондиционера» ТО «Температура в помещении повышается»;

Продукционное представление знаний с человеческой точки зрения является прямым описанием логических выводов при решении конкретных задач. Совокупность знаний о конкретной предметной области в этом случае представляется соответствующим набором продукционных правил, который образует базу знаний. При построении продукционных правил допустимо использование логических операторов И, ИЛИ, например:

- ЕСЛИ «Температура в реакторе превышает 120 ° С » И «Температура хладагента превышает 90 ° С » ТО «Прекратить подачу топлива»;
- ЕСЛИ «Температура в реакторе превышает 90 ° С » ИЛИ «Температура хладагента превышает 60 ° С » ТО «Снизить подачу топлива на 40%».

Недостатком языка продукционных правил можно считать отсутствие явных связей между правилами и целями, к достижению которых необходимо стремится. Таким образом, для активизации одного из продукционных правил необходимо проверка всей продукционной базы знаний, что при больших объемах информации приводит к существенным затратам временных и технических ресурсов интеллектуальной системы. Возможность решения этой проблемы заключается в разработке перспективных продукционных баз знаний, в которых одни продукционные правила могут активировать и дезактивировать другие продукционные правила, влияя на количество перебираемых правил в текущем цикле и, следовательно, на выбор пути достижения цели управления.

Отличительной чертой и основным преимуществом продукционной базы знаний является простота анализа, дополнения, модификации и аннулирования определенных продукционных правил. Помимо этого, представление знаний в таком синтаксически однотипном виде существенно облегчает техническую реализацию системы использования знаний. Вследствие этого в настоящее время продукционные базы

знаний получили наибольшее распространение в интеллектуальных технических системах.

Знаниями можно назвать описания отношений между абстрактными понятиями и сущностями, являющимися конкретными объектами реального мира. Изначально семантические сети разрабатывались как модели долговременной человеческой памяти в психологии, но впоследствии эта модель перекочевала в инженерию знаний. В семантической сети абстрактные понятия и отношения между ними описываются в виде узлов и дуг. Сущности и понятия в такой сети являются узлами, а отношения между ними – дугами. Атрибуты семантических сетей можно разделить на лингвистические (объект, условие, место, инструмент, цель и т.п.), атрибутивные (форма, размер, цвет и т.п.), характеристические (род, время, наклонение и т.п.), логические (да, нет, отрицание, объединение и т.п.).

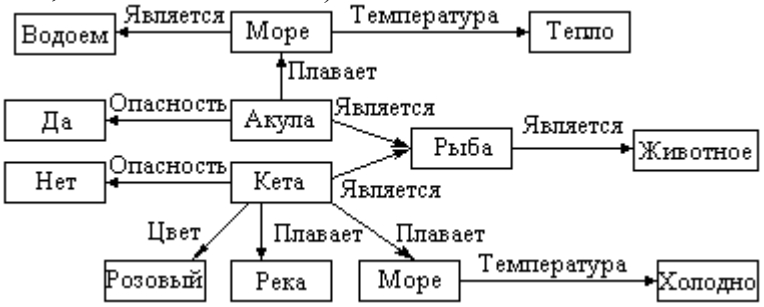


Рис.1.5. Семантическое представление знаний биолога

Допустим, фрагмент знаний ихтиолога о биологии рыб можно описать следующей семантической сетью (рис.1.5). В качестве другого примера рассмотрим представление знаний, содержащихся в высказывании: «Робот сверлит отверстие в детали с помощью сверла 10» (рис.1.6).

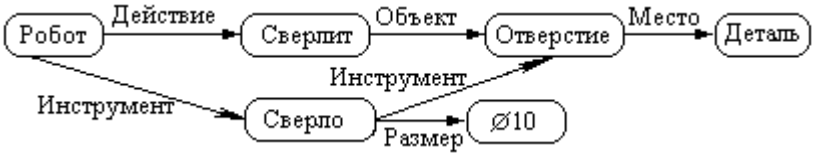


Рис.1.6. Семантическое представление технического знания

Недостаток семантических сетей – дублирование информации при построении сетей и смешение групп знаний, относящихся к различным ситуациям. Например, семантическая сеть, представленная на рис.1.5, имеет дубляж понятия «море», а отношение «температура» может использоваться не только для описания среды обитания животных. Выходом из данной ситуации стала наметившаяся в последнее время тенденция к построению разделенных семантических сетей.

Основным преимуществом семантических сетей является то, что они имитируют понимание и использование человеком естественного языка, что позволяет применять их при техническом моделировании рассуждений, доказательстве теорем, построении незаданных явно причинно-следственных связей и лингвистических конструкций, т.е. семантические сети позволяют реализовать устройства, имитирующие мыслительные акты более высокого уровня по сравнению с продукционными правилами. Представление знаний в виде семантических сетей широко используется в интеллектуальных системах интерпретации естественного языка и автоматического машинного перевода, в диалоговых вопросно-ответных системах естественного человеко-машинного общения, в блоках логической интерпретации систем технического зрения.

Приобретением знаний называется выявление знаний из источников и преобразование их в нужную форму, а также перенос в базу знаний ИС. Источниками знаний могут быть книги, архивные документы, содержимое других баз знаний и т. п., т. е. некоторые *объективизированные знания,* переведенные в форму, которая делает их доступными для потребителя. Другим типом знаний являются *экспертные знания,*

которые имеются у специалистов, но не зафиксированы во внешних по отношению к нему хранилищах. Экспертные знания являются *субъективными.* Еще одним видом субъективных знаний являются *эмпирические знания.* Такие знания могут добываться ИС путем наблюдения за окружающей средой (если у ИС есть средства наблюдения).

Ввод в базу знаний объективизированных знаний не представляет особой проблемы, выявление и ввод субъективных и особенно экспертных знаний достаточно трудны. Чтобы разработать методологию приобретения субъективных знаний, получаемых от эксперта, надо четко различать две формы репрезентации знаний. Одна форма связана с тем, как и в каких моделях хранятся эти знания у человека-эксперта. При этом эксперт не всегда осознает полностью, как репрезентированы у него знания. Другая форма связана с тем, как инженер по знаниям, проектирующий ИС, собирается их описывать и представлять. От степени согласованности этих двух форм репрезентации между собой зависит эффективность работы инженера по знаниям.

В когнитивной психологии изучаются формы репрезентации знаний *(когнитивные структуры знаний)* характерные для человека. Примерами могут служить [Хафман, 1986]: представление класса понятий через его элементы (например, понятие "птица" репрезентируется рядом чайка, воробей, скворец, ...)

представление понятий класса с помощью базового прототипа, отражающего наиболее типичные свойства объектов класса (например, понятие "птица" репрезентируется прототипом нечто с крыльями, клювом, летает ...) представление с помощью признаков (для понятия "птица", например, наличие крыльев, клюва, двух лап, перьев).

Кроме понятий репрезентируются и отношения между ними. Как правило, отношения между понятиями определяются процедурным способом, а отношения между составляющими понятий (определяющими структуру понятия) -декларативным способом. Наличие двух видов описаний заставляет в моделях представления знаний одновременно иметь оба компонента, например семантическую сеть и продукционную систему, как это представлено в когнитивной модели [Anderson, 1983].

При приобретении знаний важную роль играют так называемое *поле знаний*в котором содержатся основные понятия, используемые при описании предметной области, и свойства всех отношений, используемых для установления связей

между понятиями. Поле знаний связано с концептуальной моделью проблемной области, в которой еще не учтены ограничения, которые неизбежно возникают при формальном представлении знаний в базе знаний. Переход от описания некоторой области в поле знаний к описанию в базе знаний аналогичен переходу от концептуальной модели базы данных к ее логической схеме, когда уже зафиксирована система управления базой данных. Важно отметить, что переход непосредственна к формальным представлениям в базе знаний без этапа концептуального описания в поле знаний приводит к многочисленным ошибкам, что замедляет процесс формирования базы знаний ИС.

Возможны три режима взаимодействия инженера по знаниям с экспертом-специалистом: *протокольный анализ, интервью и игровая имитация профессиональной деятельности.*Протокольный анализ заключается в фиксации (например, путем записи на магнитную ленту "мыслей вслух" эксперта во время решения проблемы и в последующем анализе полученной информации. В режиме интервью инженер по знаниям ведет с экспертом активный диалог, направляя его в нужную сторону. При игровой имитации эксперт помещается в ситуации, похожие на те в которых протекает его профессиональная деятельность. Наблюдая за его действиями в различных ситуациях, инженер по знаниям, формирует свои соображения об экспертных знаниях, которые впоследствии могут быть уточнены с экспертом в режиме интервью. Принципы игровой имитации нашли применение в разнообразных деловых играх, специальных тренажерах.

Каждый из упомянутых способов извлечения знаний имеет свои преимущества и недостатки. Так, при анализе протоколов инженеру по знаниям нелегко отделить понятия, важные для включения в словарь предметной области, от тех, которые при "мыслях вслух" появляются случайно. Кроме того, в протоколах обнаруживаются пробелы, когда рассуждение эксперта как бы прерывается и продолжается уже на основе пропущенных шагов вывода. Заполнение подобных лакун возможно лишь в режиме интервью. Таким образом, во всех трех подходах к извлечению знаний из экспертов необходим этап интервью, что делает его одним из важнейших методов приобретения знаний.

Существует не менее двух десятков стратегий интервьюирования. Наиболее известны три: разбиение на ступени, репертуарная решетка и подтверждение сходства,

При разбиении на ступени эксперту предлагается назвать наиболее важные, по его мнению, понятия предметной области и указать между ними отношения структуризации, т. е. отношения типа "род-вид", "элемент-класс", "целое- часть" и т. п. Эти понятия используются на следующем шаге опроса как базовые. Стратегия нацелена на создание иерархии понятий предметной области, выделение в понятиях тесно связанных между собой групп-*гаксонов* (кластеров) .

Стратегия репертуарной решетки направлена на выявление характеристических свойств понятий, позволяющих отделять одни понятия от других. Методика состоит в предъявлении эксперту троек понятий с предложением назвать признаки для каждых двух понятий, которые отделяли бы их от третьего. Так как каждое понятие входит в несколько троек, то на основании такой процедуры происходит уточнение объемов понятий и формируются "симптокомплексы" понятий, с помощью которых эти понятия могут идентифицироваться в базе знаний.

Стратегия подтверждения сходства состоит в том, что эксперту предлагается установить принадлежность каждой пары понятий из предметной области к некоторому отношению сходства (толерантности). Для этого эксперту задается последовательность достаточно простых вопросов, цель которых заключается в уточнении того понимания сходства, которое вкладывает эксперт в утверждение о сходстве двух понятий предметной облает.

Процесс взаимодействия инженера по знаниям (аналитика) с экспертом-специалистом включает три основных этапа.

1. Подготовительный этап. Для успеха общения оба участника должны тщательно подготовиться к диалогу или игре. Желательно, чтобы эксперт был не только компетентным специалистом, но и заинтересованным (морально или материально) лицом в достижении конечной цели-построении ИС. Он должен быть доброжелателен к аналитику и уметь объяснять свои знания (наилучший случай когда эксперт имеет опыт преподавательской работы).

Аналитику необходимо: глубоко познакомиться со специальной литературой по предметной области" чтобы не задавать очень "глупых" вопросов (просто "глупые" вопросы бывают чрезвычайно полезны), а также увеличить количество "пакетов ожиданий" [Шенк и др., 1987]; уметь слушать и грамотно задавать вопросы; настроиться на роль "ученикам, а не "экзаменатора"; разбираться в моделях когнитивной психологии, а также в моделях представления знаний, чтобы из знаний эксперта выделять четкие структуры.

В любой совместной деятельности большое значение имеют психологические качества исследователей, такие как личность, манера поведения, стиль научного мышления. Существуют различные классификации научных работников. В качестве примера приведем следующую: инициатор - быстро реагирует на перспективные проблемы, т. е. один из первых ощущает необходимость решения проблемы с элементами неопределенности; диагност-способен к быстрой оценке сильных и слабых сторон решения задачи, эрудит-наделен исключигельной памятью, отличается повышенным вниманием к деталям и стремлением к упорядоченности; ремесленник - способен

воплощать в жизнь плохо оформленные идеи других; эстет - стремится исследовать проблемы, приводящие к изящным решениям, не склонен к кропотливому труду; методолоза интересован методологическими аспектами исследований; независимый-стремится к индивидуальному решению проблем; фанатик-самоотверженно увлечен своей научной проблемой, того же требует и от окружающих.

Принадлежность научного работника к тому или иному типу определяется с помощью косвенных методик (тестов личности, интеллекта, когнитивных стилей.проектных методик). Автоматизация опроса и получения психологического портрета испытуемого реализована, например, в системе АВТАНТЕСТ [Гаври-лова. 1984].

Для роли эксперта наиболее предпочтительны инициатор эрудит, диагност и ремесленник (в паре с аналитиком-эрудитом), а для роли аналитика-диагност, методолог, эрудит, инициатор. При этом наилучшее сочетание дают сочетания разных типов. Благодаря различиям в подходах к решению задачи, в точках зрения, стиле мышления восприятия, памяти и т. п. участники в такой паре с разных сторон подходят к поставленной цели, в результате увеличивается общее количество гипотез, идей, альтернативных вариантов, а следовательно, обогащается поле знаний. Однако не все сочетания даже из приемлемых типов улучшают взаимодействие, а некоторые типы (например, фанатик, эстет, независимый.ремесленник) часто слабо приспособлены для творческого взаимодействия, что приводит к возникновению скрытых и явных конфликтов, которые усложняют процесс продуктивного общения.

Важное значение имеет также лидерство в паре. В ходе любого диалога одна сторона обычно занимает позицию ведущей, чаще эту роль берет интервьюер, т. е. аналитик. Роль лидера в диалоге позволяет аналитику направлять и систематизировать процесс создания поля знания, не давая эксперту "размыть" или излишне детализовать процесс. С другой стороны, догматизм и настойчивость могут привести к неадекватному полю. Имеет место также эффект "фасада", т. е. желание эксперта не ударить "в грязь лицом" перед аналитиком, и отсюда генерирование неподтвержденных гипотез.

2. Установление "общего кода". Для создания лингвистического альянса взаимодействия участники взаимодействия должны пытаться сократить "расстоянием между объектом (т. е. исследуемой предметной областью) и аналитиком. Необходимо определить главные понятия, т. е. выработать словарную основу базы знаний; уровень детализации; взаимосвязи между понятиями.

3. Гносеологический этап. На этом этапе происходит выяснение закономерностей, присущих предметной области, условий достоверности и истинности утверждений, структурирование за счет введения отношений и т. п. Этот этап является определяющим во взаимодействии аналитика и эксперта, В процессе анализа игры или диалога вербализуется и формализуется знание эксперта и зачастую для него самого порождается новое знание. Репрезентация внешнего мира в его памяти получает материальное воплощение в форме поля знаний.

В процессе извлечения знаний сначала желательно получить от эксперта поверхностные знания такие например, как репрезентация признаков), постепенно переходя к глубинным структурам и более абстрактным понятиям (таким, например, как прототипы).

При формировании поля знаний учитываются особенности эмпирического знания: модальность, противоречивость, неполнота и т. д,

Аналитик должен за частным всегда видеть общее, т. е. строить цепочки 1факт - обобщенный факт - эмпирический закон - теоретический закон". Центральное звено цепочки - формализация эмпирики. При этом иногда основным на этапе формализации становится не извлечение "слепых" непонятных связей, а понимание внутренней структурной связи понятий предметной области. Искусство аналитика состоит в стремлении к созданию ясной и понятной модели проблемной области.

Следует также учитывать, что эксперты в проблемной области не всегда опираются на логические рассуждения. В их представлениях о проблемной области и методах решения задач, характерных для нее, широкое применение находят ассоциативные рассуждения и рассуждения правдоподобия. Опишем примерную методику работы с экспертом по формированию поля знаний.

Подготовительный этап

1. Четкое определение задач проектируемой системы (сужение поля знаний): определение, что на входе и выходе; определение режима работ, консультации, обучение и др.

2. Выбор экспертов: определение количества экспертов; выбор уровня компетентности (не всегда хорошо выбирать самый высокий уровень сразу); определение способов и возможности заинтересовать экспертов в работе; тестирование экспертов.

3. Знакомство аналитика со специальной литературой в предметной области

4. Знакомство аналитика и экспертов (в дальнейшем для простоты будем считать, что эксперт один).

5. Знакомство эксперта с популярной литературой по искусственному интеллекту (желательно, но необязательно).

6. Попытка аналитика создать поле знаний первого приближения априорным знаниям из литературы (прототип поля знаний).

Основной этап

1. "Накачка" поля знаний: а) в зависимости от предметной области выбор способа интервьюирования; б) протоколирование мыслей вслух или запись на магнитофон рассуждении эксперта (аналитик по возможности не должен пока вмешиваться в рассуждения).

2. "Домашняя работа". Попытка аналитика выделить некоторые причинно-следственные связи в рассуждениях эксперта; построение словаря предметной области (возможно, на карточках) и подготовка вопросов к эксперту.

3. "Подкачка" поля зрения. Обсуждение с экспертом прототипа поля знаний и домашней работы, а также ответы на вопросы аналитика.

4. Формализация концептуальной модели.

5. Построение поля знаний второго приближения.

Системы приобретения знаний от экспертов

Одно из первых рассмотрении интервью как метода инженерии знаний проведено в [Newel 1972], Проблемы, возникающие при извлечении экспертных знаний, некоторые психологи связывают с так называемой когнитивной защитой. В [Kelly, 1985] была развита теория человеческого познания, основанная на понятии "персональных конструктов", которые человек создает и пытается приспособить к реалиям мира. В [Bose, 1984] теория персональных конструктов использована для создания системы извлечения экспертных знаний и показала свою способность успешно преодолевать когнитивную защиту, т. е. нежелание экспертов достичь четкого и осознанного ими истолкования основных понятий, отношений между понятиями и приемов решения задач в интересующей инженера по знаниям проблемной области.

Методы интервьюирования эксперта предметной области знаний с использованием нескольких различных стратегий применены при создании системы TEIRESIAS [Davis, 1982]. В [KahnetaL, 1984] выделено восемь различных стратегий интервью, в [KahnetaL, 1985] на основе этих стратегий исследуется возможность автоматического

интервьюирования. Автоматизации метода протокольного анализа посвящены работы [Waterman, 1971, 1973; Krippendorf, 1980].

В [Kahnetal. 1985] на примере диагностической системы MORE; описана техника интервьюирования, направленная на выяснение следующих сущностей, гипотез, симптомов, условий, связей и путей. Гипотеза - событие идентификация которого имеет своим результатом диагноз. Симптом-событие, являющееся следствием существования гипотезы, наблюдение которого приближает последующее принятие гипотезы. Условие - событие или некоторое множество событий, которое не является непосредственно симптоматическим для какой-либо гипотезы, но которое может иметь диагностическое значение для некоторых других событий. Связи-соединения сущностей (в том числе, других связей). Путь- выделенный тип связи, который соединяет гипотезы с симптомами. В соответствии с этим используются следующие стратегии интервью: дифференциация гипотез, различение симптомов, симптомная обусловленность, деление пути и др.

Дифференциация гипотез направлена на поиск симптомов, которые обеспечивают более точное различение гипотез. Наиболее мощными в этом смысле являются те симптомы, которые происходят из одного диагностируемого события, Различение симптомов выявляет специфические характеристики симптома, которые, с одной стороны, идентифицируют его как следствие некоторой гипотезы, с другой-противопоставляют другим. Симптомная обусловленность направлена на выявление негативных симптомов, т. е. симптомов, отсутствие которых имеет больший диагностический вес, чем их присутствие. Деление пути обеспечивает нахождение симптоматических событий, которые лежат на пути к уже найденному симптому. Если такой симптом существует, то он имеет большое диагностическое значение, чем уже найденный.

Аналогичные стратегии интервьюирования эксперта использованы при создании инструментальной диагностической системы ИДИС [Голубев и др., 1987].

В системе KRITON [DiederichetaL, 1987] для приобретения знаний используются два источника: эксперт с его знаниями, полученными на практике (эти знания, как правило, неполны, отрывочны, плохо структурированы); книжные знания, документы, описания инструкции (эти знания хорошо структурированы и фиксированы традиционными средствами). Для извлечения знаний из первого источника в KRITON применена техника интервью, использующая стратегии репертуарной решетки и разбиения на ступени. При этом применяется прием переключения стратегий: если при предъявлении тройки семантически связанных понятий эксперт не в состоянии назвать признак, отличающий два из них от третьего, система запускает стратегию разбиения на ступени и предпринимает попытку выяснения таксономической структуры этих понятий с целью выявления признаков, их различающих.

Для выявления процедурных знаний эксперта в KRITON применен метод протокольного анализа. Он осуществляется в пять шагов. На первом шаге протокол делится на сегменты на основании пауз, которые делает эксперт в процессе записи. Второй шаг-семантический анализ сегментов, формирование высказываний для каждого сегмента. На третьем шаге из текста выделяются операторы и аргументы. Далее делается попытка поиска по образцу в базе знаний для обнаружения переменных в высказываниях (переменная вставляется в высказывание, если соответствующая ссылка в тексте не обнаружена). На последнем шаге утверждения упорядочиваются в соответствии с их появлением в протоколе.

Анализ текста используется в KRITON для выявления хорошо структурированных знаний из книг, документов, описаний, инструкций.

В [Morik, 1987] описан метод выявления модели предметной области. Первая фаза-формирование инженером знаний грубой модели предметной области путем определения предикатов и сортов их возможных аргументов и сообщения системе фактов об области, выразимых этими предикатами. Система выявляет свойства предикатов и устанавливает отношения между ними, структурируя таким образом предметную область. На второй

фазе с помощью метазнаний (общих структур), отражающих особенности человеческого мышления, осуществляется проверка соответствия фактов предикатам, индуктивный вывод правил из фактов, вывод правил из других правил.

В системах SIMER и ДИАПС [Осипов. 1987; OsipovetaL, 1987] основным методом приобретения знаний является автоматизированное интервьюирование эксперта, которое управляется знаниями, приобретенными системой. В системах SIMER и ДИАПС не выявляется предварительная модель области. Все объекты (события) и их атрибуты определяются в режиме прямого интервьюирования эксперта. Предполагается только, что на множестве объектов могут быть заданы ряд отношений из известного (конечного) множества: "элемент-множество", "часть - целое", "пример - прототип", отношения структурного сходства объектов, структурной иерархии и некоторые другие. Все отношения попарно различаются формальными свойствами. Так, отношений структурного сходства не обладает транзитивностью, но симметрично. Отношение структурной иерархии.напротив, не обладает симметричностью, однако транзитивно. На выяснение этих и ряда других свойств отношений и объектов направлено интервью.

В частности, для установления структурного сходства на первой фазе интервью для каждого вновь вводимого понятия эксперту предлагается указать (с помощью меню) те понятия предметной области, с которыми может быть связано данное (без спецификации отношения). Затем в процессе интервью для каждой пары понятий (из выделенных на первой фазе) связь специфицируется, устанавливаются свойства и тип отношения, в число элементов которого включается исследуемая пара. Так, для включения некоторой пары понятий *X* и У. о которых эксперт сообщил, что *X* влияет на У (например*X* увеличивает возможность У), в число элементов некоторого отношения *Я,* обладающего среди прочих свойств симметричностью, необходимо задать эксперту вопрос: "Увеличивает ли У возможность ?". При положительном ответе на этот вопрос (и если прочие свойства уже установлены и удовлетворяют определению отношения *Я)* пара *(X,* У) включается в *R,* Для установления структурного сходства и структурной иерархии понятий используются стратегии подтверждения сходства и разбиения на ступени.

В модели имеются метапроцедуры и метаправила, которые проверяют корректность модели, используют формальные свойства отношений для пополнения модели и генерируют правила.

Сформулируем основные этапы реализации системы приобретения знаний.

1. Интервью для определения актуальной области, в которой происходит процесс решения интересующей проблемы, и расчленение ее на автономные области.

2. Автоматизированное интервью для выявления и формирования декларативной модели предметной области.

3. Протокольный анализ к выявленным на предыдущем этапе понятиям и отношениям предметной области для пополнения модели процедурными знаниями.

(этапы 2 и 3 можно использовать попеременно до тех пор, пока модель не достигнет нужной полноты).

4. Протокольный анализ для попонения декларативных знаний модели.б. Проверка полноты модели. Обычно протокольный анализ выявляет пустоты в модели. Имеется в виду случай, когда понятия, использованные в "мыслях вслух", недостаточно описаны. В этом случае интервью и протокольный анализ повторяются.

2.2 Лабораторная работа №2 (2 часа).

Тема: «Формальные логические модели знаний**.** Продукционные модели представления знаний**»**

2.2.1 Цель работы: изучить формальные логические модели знаний

2.2.2 Задачи работы:

1. Научиться решать задачи связанные, с формализацией знаний.

2. Проектирование экспертных систем в заданной предметной области с помощью продукционной модели.

2.2.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. ПЭВМ

2.2.4 Описание (ход) работы:

Формализация знаний основана на системе исчисления **предикатов первого порядка**, которая в свою очередь основывается на исчислении высказываний. Высказыванием называется предложение, принимающее только два значения: истина или ложь. Например: «Иван студент». Из простых высказываний с помощью слов: и, или, не, если – то, могут формироваться более сложные высказывания.

Иван студент и Татьяна студентка;

Иван студент или Татьяна студентка.

Логика высказываний оперирует логическими связями между высказываниями, то есть решает вопросы типа:

Можно ли на основе высказывания А получить высказывание В?

Истинно ли высказывание В при истинности высказывания А?

Элементарные высказывания, т.е. те, которые нельзя разделить на частичные, могут рассматриваться как переменные логического типа, над которыми разрешены следующие логические операции: отрицание (┐); конъюнкция или логическое умножение (^); дизъюнкция или логическое сложение (˅); импликация (→); эквивалентность (↔). Исчисление высказываний позволяет формализовать лишь малую часть множества рассуждений, поскольку этот аппарат не позволяет учитывать внутреннюю структуру высказывания, которая существует в естественных языках.

Пример 1. Пусть сформулированы следующие высказывания:

P: Все люди смертны;

Q: Сократ – человек;

R: Сократ – смертен.

Можно составить формулу:

(P^Q) → R

Однако эта формула не является общезначимой, поскольку относится только к Сократу. Кроме того, высказывание R не выводится из P и Q, то есть при его отсутствии невозможно записать импликацию. Для достижения общезначимости Q необходимо разделить на две части: «Сократ» (субъект) и «человек» (свойство субъекта), что можно записать в виде некоторой функции:

человек (Сократ)

или в общем случае

человек (x)

Такая запись имеет внутреннюю структуру, т.к. значение высказывания является функцией его компонент, не является элементарным высказыванием и называется предикатом первого порядка.

Исчисление предикатов первого порядка – это формальный язык, используемый для представления отношений между объектами и для выявления новых отношений между объектами на основе существующих [7,10]. Алфавит языка исчисления предикатов первого порядка включает переменные, константы, предикаты, логические операции, функции, кванторы ($\forall, \exists$). Конструкцией предложений в языке исчисления предикатов первого порядка управляют синтаксические правила.

Терм – это переменная, константа или результат применения функции к терму, например, a, x, f (x). Предложения языка исчисления предикатов первого порядка есть формулы, определенные следующим образом:

1. Если P – n-арный предикат (предикат от n аргументов) и t_1, t_2, …, t_n – термы, тогда P (t_1, t_2, …, t_n) – атомическая формула (атом).
2. Атом – это правильно построенная формула.
3. Если F_1 и F_2 – атомы, то $F_1 \wedge F_2$, $F_1 \vee F_2$, $F_1 \rightarrow F_2$, $\neg F_1$ – тоже атомы.
4. Если F – формула и x – не связанная квантором переменная в F, тогда $\forall$ x (F) и $\exists$ x (F) – также атомы.

Чтобы избежать неоднозначности, необходимо определять формулы, в которых все переменные квантованы, т.е. связаны кванторами, например, $\forall$ x $\exists$ y ЛЮБИТ (x,y).

Такая формула называется замкнутой. Замкнутая формула имеет единственное истинное значение. Формула $\exists$ y ЛЮБИТ (x,y) является незамкнутой или открытой.

Для построения модели некоторой предметной области следует описать известные факты на языке логики предикатов и, используя ее результаты, построить систему, способную на основе имеющихся фактов строить некоторые новые предложения и отвечать на поставленные вопросы.

Пример 2. Пусть заданы предикаты:

E (x) – «x» въезжает в строку;

$\vee$ (x) – «x» высокопоставленное лицо;

S (x,y) – «y» обыскивает «x»;

C (y) – «y» - таможенник;

P (x) – «x» способствует провозу наркотиков.

Тогда произвольные предложения на естественном языке могут быть записаны в виде:

1. Таможенники обыскивают всех, кто въезжает в страну, кроме высокопоставленных лиц:

$\forall$ x (E (x) $\wedge \neg \vee$ (x) $\rightarrow$ ($\exists$ y (S (x,y) $\wedge$ C (y)))).

2. Некоторые люди, въезжавшие в страну и способствовавшие провозу наркотиков, были обысканы исключительно людьми, способствовавшими провозу наркотиков:

$\exists$ x (E (x) $\wedge$ P (x) $\wedge$ ($\forall$ y (S (x,y) $\rightarrow$ P (y)))).

3. Никто из высокопоставленных лиц не способствовал провозу наркотиков:

$\forall$ x (P (x) $\rightarrow \neg \vee$ (x)).

4. Некоторые таможенники способствуют провозу наркотиков:

$\exists$ x (P (x) $\wedge$ C (x)).

Задача состоит в том, чтобы, признав фактами предложения 1, 2, 3, доказать, что предложение 4 является истинным.

Для машинного решения вышеприведенной задачи используется методика автоматического формирования суждений, или метод дедукции. При этом последовательно реализуются процедуры: исключение знаков импликации; ограничение области действия знака отрицания; переименование переменных; вынесение кванторов в начало формулы; исключение кванторов и др. При автоматизации вывода доказательств методами исчисления предикатов требуется определить ряд процедур для выбора правил, позволяющих предотвратить «комбинаторный взрыв» и обеспечить проведение немонотонных рассуждений. Решением стало создание декларативных (непроцедурных) языков программирования, в частности Пролога. Программирование на Прологе состоит из этапов:

- объявление некоторых фактов об объектах и отношениях между ними;
- определения некоторых правил об объектах и отношениях между ними;
- формулировки вопросов об объектах и отношениях между ними.

Реально исчисление предикатов первого порядка в промышленных ЭС практически не используется. Формально-логическая модель представления знаний применима в основном в исследовательских системах, т.к. предъявляет очень высокие требования и ограничения к предметной области.

Рассмотрим применение аппарата нечеткой логики на примере оценки надежности поставщика, в котором кроме фактора финансового состояния учитывается и фактор формы собственности. Пусть государственное предприятие не имеет задолженность с уверенностью 60 и предполагается, что его рентабельность удовлетворительна с уверенностью 80. Фрагмент множества правил имеет следующий вид:

Правило 1: *Если Задолженность = "нет" и Рентабельность = "удовл." То Финансовое_состояние = "удовл." cf 100*

Правило 2: *Если Финансовое_состояние = "удовл." То Надежность += "есть" cf 90*

Правило 3: *Если Предприятие = "государств." То Надежность+ = "есть" cf 50*

Результат выполнения первого правила: *cf(посылки) = min(60,80) = 60, cf(Фин_сост.="удовл.") = 60*100/100 = 60.*

Результат выполнения второго правила: *cf(Надежность="есть") = 60*90/100 = 54*

Результат выполнения третьего правила: *cf(Надежность="есть") = 54 + 50 - 54*50/100 = 67*

Динамические модели. Моделирование рассуждений человека, как правило, не сводится только к прямой или обратной аргументации. Сложные проблемы решаются путем выдвижения во времени нескольких гипотез с анализом подтверждающих фактов и непротиворечивости следствий. Причем для многоцелевых проблемных областей происходит увязка гипотез по общим ограничениям. При этом возможны задержки в принятии решений, связанные со сбором подтверждающих фактов, доказательством подцелей, входящих в ограничения.

Следовательно, для подобных динамических проблем важна рациональная организация памяти системы для запоминания и обновления получаемых промежуточных результатов, обмен данными между различными источниками знаний для достижения нескольких целей, изменение стратегий вывода с выдвижения гипотез (прямая аргументация) к их проверке (обратная аргументация). Целям построения таких гибких механизмов вывода служит применение технологии "доски объявлений", через которую в результате осуществления событий источники знаний обмениваются сообщениями.

В целях динамического реагирования на события некоторые продукционные модели используют специальные правила-демоны, которые формулируются следующим образом: "Всякий раз, как происходит некоторое событие, выполнить некоторое действие". Например:

Всякий раз, как становится известным значение переменной "Поставщик",

Выполнить набор правил "Финансовый анализ предприятия"

В программном средстве GURU подобное правило будет записано следующим образом:

IF: KNOWN("Поставщик") = true

THEN: CONSULT FIN_AN

Для динамических экспертных систем характерна также обработка времени как самостоятельного атрибута аргументации логического вывода:

Если в течение дня уровень запаса понизился больше, чем на 50 % То выполнить набор правил "Выбор поставщика для поставки"

Общим недостатком всех формализмов представления знаний, основанных на правилах, является недостаточно глубокое отражение семантики проблемной области, что может сказываться на гибкости формулирования запросов пользователей к экспертным

системам. Этот недостаток снимается в объектно-ориентированных методах представления знаний.

Изучение базы знаний ЭС лучше всего начать с анализа концептуальной схемы, поскольку концептуальная схема дает наиболее общее представление о структуре базы знаний. Рассмотрим в качестве примера базу знаний ЭС прогнозирования продолжительности жизни человека. На рис. 1 представлена концептуальная схема этой ЭС. На этой схеме показаны связи между объектами, которые используются в процессе логических выводов. Терминальные объекты выделены на схеме жирными рамками.

«Продолжительность» жизни, являющаяся целью консультации, определяется «основной продолжительностью» и некоторым «фактором», который может привести к увеличению или сокращению «продолжительности» жизни по сравнению с «основной продолжительностью» в зависимости от того, будет значение этого «фактора» положительным или отрицательным.

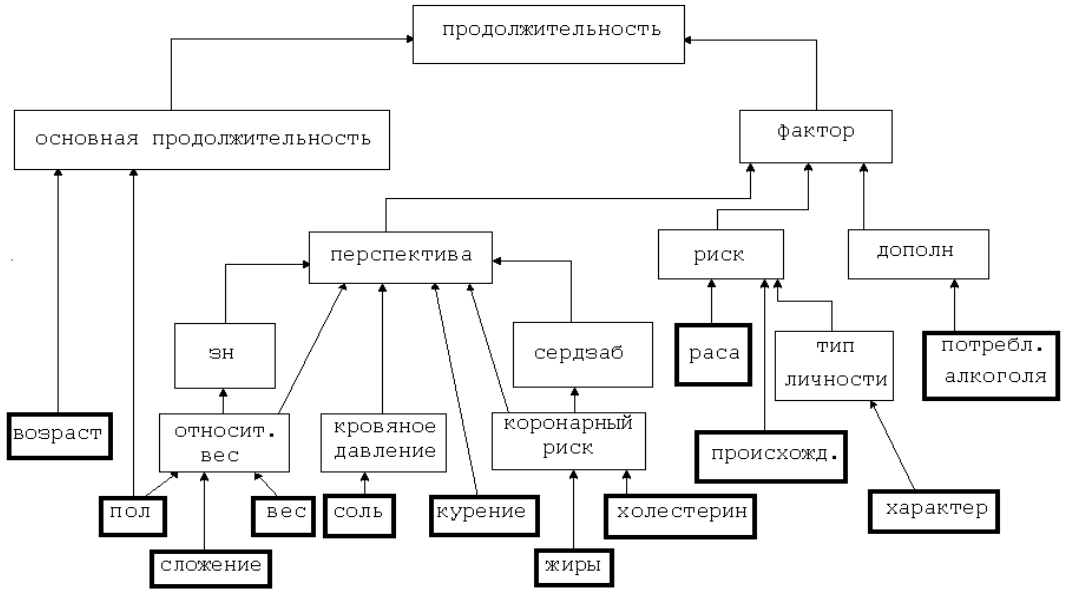


Рис. 1. Концептуальная схема ЭС

Рис. 1. Концептуальная схема ЭС

Как видно из рис. 1, концептуальная схема дает представление не только о структуре базы знаний, но и о подходе к решению задач ЭС. Однако на концептуальной схеме указываются только имена объектов и не указываются их значения, поэтому для более полного представления процесса решения задач необходимо использовать граф И/ИЛИ.

На рис. 2 представлен фрагмент графа И/ИЛИ, соответствующего концептуальной схеме, показанной на рис. 1. На рис. 2 в кружках указаны номера правил, а в прямоугольниках – условия и заключения правил. На графе И-вершины помечены дугой.

Для построения графа И/ИЛИ найдите в базе знаний правило, указанное преподавателем, и представьте его в виде графа. Затем выберите одно из условий этого правила и найдите в базе знаний такие правила, у которых в заключении стоит та же пара <объект> = <значение>, что и в условии исходного правила. Добавьте граф этого правила к исходному графу.

После этого возьмите второе условие исходного правила и аналогичным образом продолжите другую ветвь графа И/ИЛИ. Продолжение каждой из ветвей строится до тех пор, пока в условиях правил не встретятся объекты, значения которых запрашиваются у пользователя в процессе консультации. Эти условия соответствуют терминальным вершинам графа И/ИЛИ.

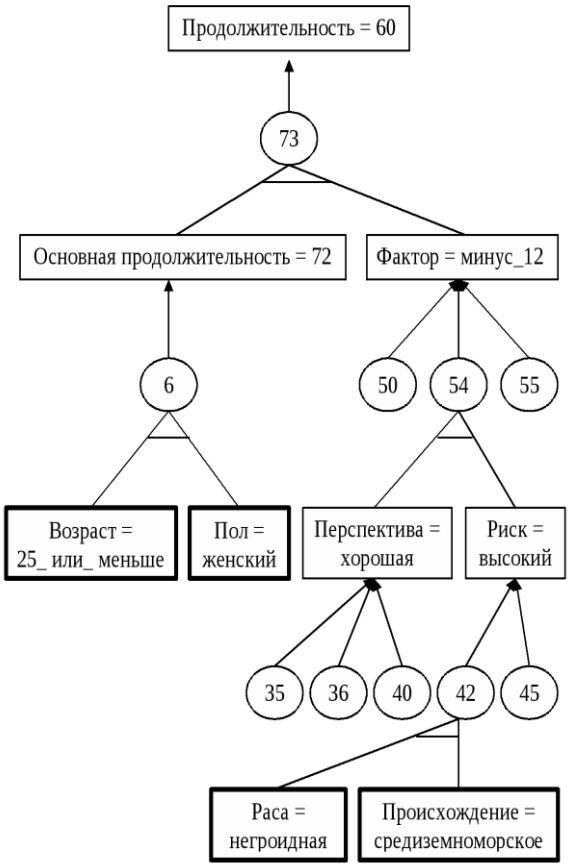


Рис. 2. Фрагмент графа И/ИЛИ для одной из гипотез

2.3 Лабораторная работа №3 (2 часа).

Тема: «Представление знаний семантическими сетями. Представление знаний фреймовыми моделями**»**

2.3.1 Цель работы: изучить представление знаний семантическими сетями, изучить фреймовые модели

2.3.2 Задачи работы:

1. Решение задач по представлению знаний семантическими сетями

2. Решение задач представления знаний фреймовыми моделями

2.3.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. ПЭВМ

2.3.4 Описание (ход) работы:

Общепринятого определения семантической сети не существует. Обычно под ней подразумевают систему знаний некоторой предметной области, имеющую определенный смысл в виде целостного образа сети, узлы которой соответствуют понятиям и объектам, а дуги - отношениям между объектами. При построении семантической сети отсутствуют ограничения на число связей и на сложность сети. Для того чтобы формализация оказалась возможной, семантическую сеть необходимо систематизировать.

Семантические сети систематизируют функции отношений между понятиями с помощью следующих признаков:

- множество — подмножество (типы отношений «абстрактное — конкретное», «целое — часть», «род — вид»);
- индексы (свойства, имена прилагательные в языке и т.п.);
- конъюнктивные связи (логическое И);
- дизъюнктивные связи (логическое ИЛИ);
- связи по ИСКЛЮЧАЮЩЕМУ ИЛИ;
- отношения «близости»;
- отношения «сходства — различия»;
- отношения «причина — следствие» и др.

При построении семантической сети отсутствуют ограничения на число элементов и связей. Поэтому систематизация отношений между объектами в сети необходима для дальнейшей формализации. Пример семантической сети представлен на рис.1

Для реализации семантических сетей существуют специальные сетевые языки: Lisp,PROLOGдля реализации системSIMER+MIRи др. Широко известны экспертные системы, использующие семантические сети в качестве языка представления знаний:PROSPECTOR,CASNET,TORUS.

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

Задание для выполнения.

Нарисовать семантическую сеть, описывающих предметную область.

1. Электрогидравлические усилители мощности
2. Гидравлические усилители мощности
3. Электрические усилители мощности
4. Датчики для САУ
5. Микропроцессоры
6. Микроконтроллеры
7. Индикаторы
8. Резисторы
9. Конденсаторы
10. Транзисторы
11. Тиристоры
12. Диоды
13. Интегральные микросхемы
14. Аналоговые микросхемы
15. Источники электропитания
16. Структура БИТТУ
17. Видеоигры
18. Видеокамеры
19. Вузы Балакова
20. Ваши преподаватели
21. Предприятия Балакова
22. Библиотека БИТТУ
23. Отечественные автомобили
24. Комплектующие ПК

Средства САПР, ориентированные на автоматизацию процедур структурного синтеза опираются на методы искусственного интеллекта (ИИ).
ИИ – это наука о знаниях, способах их получения, представления, переработки и использования в искусственных системах.
В системах ИИ для описания знаний применяют способы, основанные на понятиях фрейма и семантической сети. Фреймы – естественная форма представления сведений об элементах синтезируемых объектов в системах структурного синтеза. В настоящее время концепция фреймов быстро развивается и расширяется, благодаря развитию методов объектно-ориентированного программирования.
Фреймы – это структуры данных, в которой в определенном порядке представлены сведения о свойствах объекта.
Когда человек оказывается в новой ситуации, он извлекает из памяти ранее накопленные блоки знаний, имеющие отношение к текущей ситуации, и пытается применить их. Эти блоки знаний и представляют собой фреймы. Вероятно, знания человека организованы в виде сети фреймов, отражающих его прошлый опыт. Например: типовой номер в гостинице. Он имеет кровать, ванную комнату, шкаф для одежды, телефон и т.д. Детали каждого конкретного номера могут отличаться от приведенного описания. Но они легко уточняются, когда человек оказывается в конкретном номере: цвет обоев, положение выключателей.
Таким образом, любое представление о предмете, объекте, стереотипной ситуации у человека всегда обрамлено (отсюда frame – «рамка») характеристиками и свойствами объекта или ситуации.
Основной структурной единицей фрейма является слот – вложенная во фрейм структура данных, который представляется в виде:
‹имя слота›:{(Ai, vi)},{ri}
где Ai - имя признака, vi- его значение, ri- связь с другими слотами.
Слоты – это некоторые незаполненные подструктуры фрейма, после заполнения которых конкретными данными, фрейм будет представлять ту или иную ситуацию, явление или объект предметной области. При конкретизации фрейма ему и его слотам присваиваются конкретные имена и происходит заполнение слотов.
В качестве значений слотов могут выступать имена других фреймов, что обеспечивает построение сети фреймов.
В общем виде фрейм выглядит следующим образом:

‹Имя фрейма›:
[‹роль 1›] (‹имя слота 1› : ‹значение слота 1›);
[‹роль 2›] (‹имя слота 2› : ‹значение слота 2›);
…….
[‹роль n›] (‹имя слота n› : ‹значение слота n›).

В общем случае структура данных фрейма может содержать более широкий набор информации, в который входят следующие атрибуты.
Имя фрейма. Оно 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 для выполнения специфических функций, примерами которых являются: слот-указатель родителя данного фрейма, 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. Оно должно быть уникальным в пределах фрейма.
Значение слота. Оно должно соответствовать указанному типу данных и условию наследования. Значением слота могут быть числа или математические соотношения,

тексты на естественном языке или программы, правила вывода или ссылки на другие слоты данного фрейма или других фреймов.

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	80000	Число
Адрес	ДОМ_АДРЕС	Фрейм

Каждый фрейм можно рассматривать как семантическую сеть, состоящую из выделенных вершин и связей. Верхний уровень фрейма представляет соответствующее понятие, а последующие уровни — терминальные слоты, которые содержат конкретные значения.

Например имеет место ситуация:

Студент Сидоров получил книгу Л.Н. Толстого «Воскресение» в библиотеке им. Н.В. Гоголя, расположенной в Москве».

Описание данной ситуации может быть представлено в виде фрейма:

ПОЛУЧЕНИЕ:

ОБЪЕКТ (КНИГА: (Автор, Л.Н. Толстой), (Название, Воскресение)); АГЕНТ (СТУДЕНТ: (Фамилия, Сидоров));

МЕСТО (БИБЛИОТЕКА: (Название, им. Н.В. Гоголя),

(Расположение, г. Москва)).

Здесь ОБЪЕКТ, АГЕНТ и МЕСТО - это роли, которые играют слоты соответственно КНИГА, СТУДЕНТ и БИБЛИОТЕКА в рамках фрейма ПОЛУЧЕНИЕ.

Данную ситуацию можно представить в виде семантической сети – формы представления знаний в виде совокупности понятий и отношений между ними в некоторой предметной области (рис. 1), где можно выделить три характерных уровня. На нулевом уровне представлены конкретные значения сущностей ПО (Толстой, Воскресение, Сидоров и т.д.), на первом - понятия, используемые для описания ПО (КНИГА, СТУДЕНТ, БИБЛИОТЕКА), и на втором - описываемая ситуация ПОЛУЧЕНИЕ. Связи между отдельными понятиями, участвующими в ситуации ПОЛУЧЕНИЕ, также имеют некоторые имена, которые выражают роли понятий в рамках данной ситуации.

Рис. 1 Семантическая сеть

Совокупность фреймов, моделирующая какую-либо предметную область, представляет собой иерархическую структуру, в которой фреймы соединяются 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, находящихся на более высоком уровне иерархии. Так, фрейм АФРИКАНСКИЙ СЛОН наследует от фрейма СЛОН значение СЕРЫЙ характеристики ЦВЕТ (рис. 2). Значения характеристик фреймов могут передаваться по умолчанию фреймам, находящимся ниже них в иерархии, но если последние содержат собственные значения данных характеристик, то в качестве

истинных принимаются именно они. Это обстоятельство позволяет легко учитывать во фреймовых системах различного рода исключения. В частности, во фрейме АЗИАТСКИЙ СЛОН значением слота ЦВЕТ будет КОРИЧНЕВЫЙ, а не СЕРЫЙ, которое могло бы в нем находиться, если бы предпочтение при выборе отдавалось не собственному значению, а наследуемому от фрейма СЛОН.

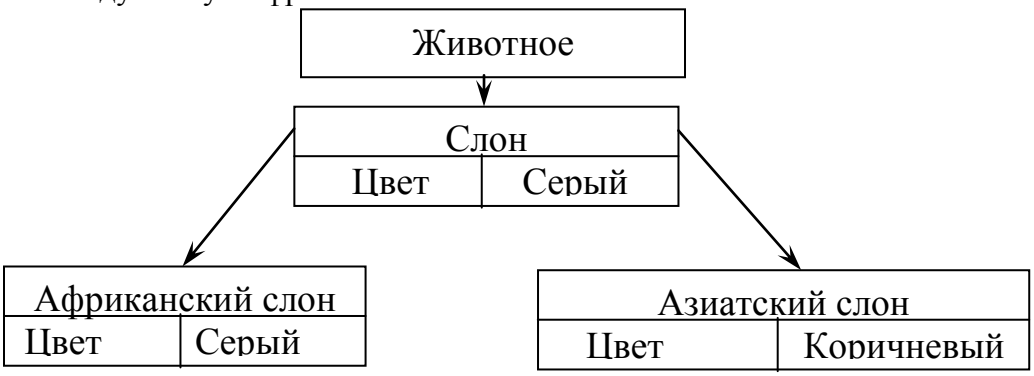


Рис.2 Иерархическая структура совокупности фреймов

Фреймы делят на различные группы – фреймы-описания и ролевые фреймы; символические и конкретные.

Фрейм-описание:

ФРУКТЫ: виноград (болгарский, 20 т)
яблоки (джонатан, 10 т)
вишня (владимирская, 200 кг)).

Ролевой фрейм:

ПЕРЕВЕЗТИ: что (прокат, 200кг)
откуда (Алчевск)
куда (Москву)
чем (железнодорожным транспортом)
когда (в ноябре 2008 г.)).

Символический фрейм:

‹РЕЗИСТОР; номинал = X1; мощность = X2; класс точности = X3; тип конструкции = X4; ГОСТ = X5›,

где X1 ÷ X5 – переменные, принимающие различные значения.

Конкретный фрейм:

‹АРМ; тип = АРМ2-05; заводской номер = 37; операционная система = Windows; назначение = АРМ конструктора; структура предприятия; вычислительная сеть САПР›,

где «структура предприятия» и «вычислительная сеть САПР» - ссылки на другие фреймы.

На рисунке 3 изображена простейшая иерархическая структура, в которой каждый фрейм имеет только один суперкласс.

Каждый подкласс или экземпляр класса наследует слоты своего суперкласса. Если подкласс (экземпляр класса) и суперкласс имеют слоты с совпадающими именами, то определения значений слотов, сделанные внутри подкласса (экземпляра класса), перекрывают определения суперкласса. Например, ответ на вопрос: «Способен ли пингвин Федя летать?» будет отрицательным. При поиске ответа на этот вопрос фрейм-экземпляр «пингвин Федя» наследует все слоты фрейма «пингвины». Значение слота «способность летать» фрейма «пингвины» перекрывает значение одноименного слота «птицы».

ЗАДАНИЕ ДЛЯ САМОСТОЯТЕЛЬНОГО ВЫПОЛНЕНИЯ:

Для выбранной самостоятельно предметной области разработать фреймы в виде взаимосвязанных таблиц, семантической сети и сложной иерархической структуры.
Электрогидравлические усилители мощности
Гидравлические усилители мощности
Электрические усилители мощности
Датчики для САУ
Микропроцессоры
Микроконтроллеры
Индикаторы
Резисторы
Конденсаторы
Транзисторы
Тиристоры
Диоды
Интегральные микросхемы
Аналоговые микросхемы
Источники электропитания
Структура БИТТУ
Видеоигры
Видеокамеры
Вузы Балакова
Ваши преподаватели
Предприятия Балакова
Библиотека БИТТУ
Отечественные автомобили
Комплектующие ПК

2.4 Лабораторная работа №4 (2 часа).

Тема: «Знакомство с оболочками ЭС**»**

2.4.1 Цель работы: приобрести навык работы с ЭС.

2.4.2 Задачи работы:

1. Работа с Малой Экспертной системой 2.0

2. Работа с оболочкой экспертной системой ESWin 1.32

.**2.4.3 Перечень приборов, материалов, используемых в лабораторной работе:**

1. ПЭВМ

2.4.4 Описание (ход) работы:

Задание 1. Знакомство с Малой Экспертной системой 2.0
1. Запустить программу «Малая Экспертная система 2.0».
2. Ознакомиться со справкой по системе.
3. Открыть Медицинскую базу знаний.
4. Ознакомиться с вопросами, которые задаются системой.
5. Провести консультацию.
6. Посмотреть другие базы знаний.
7. Зафиксировать результаты эксперимента в отчете.

Задание 2. Знакомство с оболочкой экспертной системой ESWin 1.32 для работы с продукционно-фреймовыми моделями.

1. Запустить программу.
2. Ознакомиться со справкой по системе.
3. Открыть базу знаний HEALTH.klb.
4. Выбрать цель. (*Решение ® Выбор цели*).
5. Провести консультацию. (*Решение ® Поиск решения*).
6. Посмотреть другие базы знаний (например: SCHOOL.klb, TEST.klb).
7. Изучить строение базы знаний
8. Зафиксировать результаты исследования в отчете.

Инструментальное ПО ESWin 1.32
для создания и эксплуатации экспертных систем

ПО ESWin предназначено для создания и эксплуатации советующих систем для решения различных задач, сводящихся к задачам принятия решений (диагностики, планирования, прогнозирования и т.п.).

ПО ESWin разработано на основе технологии гибридных экспертных систем с представлением знаний в виде фреймов, правил-продукций и лингвистических переменных, и возможностью разрабатывать и запускать специализированные программы в виде exe-файлов, а также, в процессе решения задач использовать данные из баз данных, доступ к которым осуществляется с помощью SQL-запросов, формируемых автоматически.

ПО ESWin позволяет создавать экспертные системы, ориентированные на решение задач диагностики, идентификации и классификации.

Ядром ПО является оболочка для разработчика

В состав инструментального ПО входят:

о экспертная оболочка для запуска экспертных систем ESWin с целью их отладки разработчиком экспертных систем,

о интерпретатор баз знаний ESWinUs для запуска экспертных систем конечным пользователем,

о редакторы баз знаний EdKB и KlbEdit, реализованные в разных стилях,

о программа для просмотра и диагностики целостности баз знаний KBView,

о программа для редактирования и оптимизации баз знаний KBOptim

Программы ESWin и KBView могут запускаться из редактора EdKB.

В качестве методов представления знаний использованы:

- правила-продукции с представлением нечеткости в виде коэффициентов достоверности с обратным логическим выводом,
- фреймы для описания структуры предметной области и диалога с пользователем,
- лингвистические переменные для описания нечетких понятий, входящих во фреймы.

С использованием ПО ESWin можно создавать эффективные советующие системы для решения, в частности, следующих задач:

- Оценка стоимости и трудозатрат для разработки WEB-сайта, информационной системы, локальной сети, рекламного ролика и т.п.,
- Выбор инструментального ПО для создания WEB-сайта, информационной системы, локальной сети, рекламного ролика и т.п.
- Выбор места отдыха в отпуск и получение рекомендаций о подготовке к поездке
- Выбор элементной базы и конструктивных решений для реализации специализированного контроллера, блока питания и т.п.
- Выбор стратегии и методов проведения рекламной кампании

Компоненты ПО ESWin

Пример базы знаний на языке ESWIn:

TITLE = ЭС для выбора метода представления знаний

FRAME = Цель
Метод представления знаний: ()
ENDF
FRAME = Тип
Решаемые задачи [Для решения каких задач планируется
использовать ЭС?]: (диагностика; проектирование;
планирование; мониторинг; прогнозирование)
ENDF
FRAME = Область
Применение [Какова область применения?]: (медицина; вычислительная техника; АСУТП; управление производством; юриспруденция)
ENDF
FRAME = Предметная область
Количество понятий [Количество понятий в предметной
области?]: (менее 10; от 10 до 100; от 100 до 1000; более
1000)
Необходимость структуризации: ()
Этапы принятия решений: (Один этап; Два этапа;
Несколько)
ENDF
FRAME=Действие
Parent:
Программа: ()
Сообщение: ()
Фрейм: ()
Удаление: ()
Запуск: ()
EndF
RULE 1
=(Предметная область. Количество понятий; менее 10)
DO
=(Предметная область.Необходимость структуризации; Нет) 100
ENDR
RULE 2
=(Предметная область. Количество понятий; более 1000)
DO
=(Предметная область.Необходимость структуризации; Да) 100
ENDR
RULE 3
=(Предметная область. Количество понятий; от 100 до 1000)
DO
=(Предметная область.Необходимость структуризации; Да) 90
ENDR
RULE 4
= (Область.Применение; медицина)
= (Тип.Решаемые задачи; диагностика)
DO
=(Метод представления знаний; Фреймы) 100
=(Метод представления знаний; Правила-продукции с представлением нечетких знаний) 90
ENDR
RULE 4

= (Область.Применение; вычислительная техника)
= (Тип.Решаемые задачи; проектирование)
DO
= (Метод представления знаний; Фреймы) 100
= (Метод представления знаний; Правила-продукции с представлением нечетких знаний) 70
= (Метод представления знаний; Семантические сети) 70
MS(Действие.Сообщение; Вероятно, требуется комбинация
предложенных методов представления знаний)
ENDR
RULE 5
=(Предметная область.Необходимость структуризации; Есть)
DO
= (Метод представления знаний; Фреймы) 100
ENDR
RULE 6
=(Предметная область.Этапы принятия решений; Несколько)
DO
= (Метод представления знаний; Правила-продукции) 100
ENDR

2.5 Лабораторная работа №5 (2 часа).

Тема: «Проектирование статической экспертной системы**»**

2.5.1 Цель работы: Построение ЭС

2.5.2 Задачи работы:

1. Диагностика неисправности автомобиля
2. Диагностика неисправности телевизора

2.5.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. ПЭВМ

2.5.4 Описание (ход) работы:

Спроектировать ЭС по диагностике неисправностей автомобиля и телевизора, в Малой экспертной системе.

2.6 Лабораторная работа №6 (2 часа).

Тема: «Методы поиска решений в ЭС. Поиск в пространстве состояний. Поиск в глубину. Поиск в ширину**»**

2.6.1 Цель работы: изучить методы поиска решений в ЭС

2.6.2 Задачи работы:

1.Выбор оптимального способа вложения денег в соответствии с потребностями инвестора

2. Профориентация школьника

3. Формирование набора документов для совершения нотариального действия (купля-продажа квартиры, оформление наследства, завещания и т.д.).

4. Выбор оптимального способа подключения к интернет (модем, локальная сеть, ADSL, GPRS, спутниковый канал).

2.6.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. ПЭВМ

2.6.4 Описание (ход) работы:

Вторым аспектом гипотезы о символьной системе является поиск решения задачи, поставленной перед интеллектуальной системой, среди всех возможных вариантов. Достаточно часто человек именно так решает проблемы, например, шахматист выбирает наилучший ход или врач рассматривает ряд возможных диагнозов. Такое поведение лежит в основе метода поиска решения в пространстве состояний.

Наиболее удобно пространство состояний представлять в виде графа, узлы которого соответствуют различным состояниям окружающего мира или состояниям решаемой задачи, а дуги – допустимым операциям, переводящим мир из одного состояния в другое, или шагам, выполняемым в процессе решения задачи. Практически любую задачу можно представить в виде такого графа, выбрав соответствующее описание для состояний и операций. В таком случае для решения поставленной задачи необходимо найти на графе пространства состояний путь от исходного состояния к требуемому.

Представление задачи в виде графа пространства состояний позволяет использовать теорию графов как для анализа структуры и степени сложности задачи, так и для разработки процедуры ее решения.

Швейцарский математик Леонард Эйлер разработал теорию графов для решения «задачи о Кенигсбергских мостах». На рис.2 изображена схема расположения города на двух берегах реки и двух островах, соединенных между собой семью мостами. Задача – найти такой маршрут обхода города, при котором каждый мост проходится только один раз. Обозначив берега реки (b_1, b_2), острова (o_1, o_2) и мосты (m_1, m_2, m_3, m_4, m_5, m_6, m_7), а также задав предикат *соединяет*(X,Y,Z), где X,Y – место, т.е. остров или берег; Z – мост, можно представить систему мостов в терминах предикатов: *соединяет* (o_1,o_2,m_1); *соединяет* (o_1,b_1,m_2); *соединяет* (o_1,b_1,m_3); *соединяет* (o_2,b_1,m_4); *соединяет* (o_1,b_2,m_5); *соединяет* (o_1,b_2,m_6); *соединяет* (o_2,b_2,m_7). Эквивалентное представление системы мостов в виде графа приведено на рис.2.

Чтобы доказать невозможность существования искомого маршрута Эйлер ввел понятие степени вершины графа, которая равна числу дуг, соединяющих данную вершину с другими вершинами. Эйлер показал, что если число вершин нечетной степени не равно нулю или двум, то маршрут невозможен, т.к. если вершин нечетной степени две, то маршрут можно начать в одной из них, а закончить в другой. Если вершин с нечетной степенью нет, то маршрут должен начинаться и заканчиваться в одной и той же вершине. Из рис.2 видно, что данное условие не выполняется, следовательно, искомый маршрут не существует.

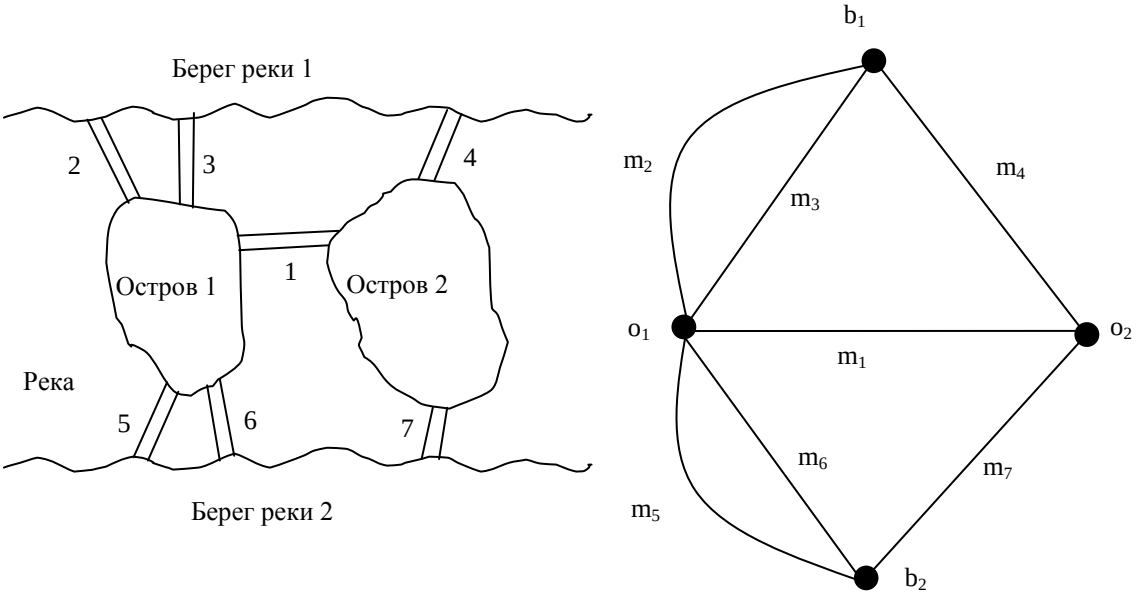


Рис.2 – Задача о кенигсбергских мостах

Следует отметить, что предикатное представление, описывая отношения между мостами, берегами и островами, не обеспечивает однозначную связь каждой вершины с дугами и, как следствие, не позволяет ввести понятие степени вершины. Следовательно, представление окружающего мира в виде графа повышает информативность по сравнению с предикатным представлением.

Напомним некоторые понятия теории графов. Граф – это множество вершин или узлов N_1, N_2, … N_n и дуг или ребер, соединяющих некоторые пары вершин. На графе состояний каждая вершина именована, а если имена двух вершин совпадают, то они считаются одинаковыми.

Дуги графа обычно идентифицируются именами вершин, которые они соединяют, т.е. обозначаются упорядоченной парой вершин, например, (N_2, N_3). Дуга может иметь собственную метку, характеризующую данную дугу, например, расстояние между узлами транспортной сети. Если каждой дуге приписано направление, то граф называется ориентированным, а пара вершин, соединяемых направленной дугой, называются соответственно родителем и потомком. У одной вершины-родителя может быть несколько вершин-потомков, которые называются братьями. Вершина, не имеющая потомков, называется концевой. Вершина, не имеющая предков, называется корнем, а граф с такой вершиной – корневой граф.

Упорядоченная последовательность вершин (N_1, N_2, … N_n), где каждая пара (N_i, N_{i+1}) является дугой, называется путем длины n-1 от вершины N_1, к вершине N_n. Если путь включает какую-то промежуточную вершину более одного раза, то путь содержит петлю. Две вершины называются связанными, если существует путь, содержащий эти вершины. Если в графе для каждой пары вершин существует единственный путь, то такой граф называется деревом и, как следствие, не содержит петель. В корневом дереве каждая вершина имеет единственного родителя. Пример корневого дерева – каталог файловой системы.

Пространство состояний задачи представляется четырьмя множествами:

- множество вершин графа *N*, определяющих возможные состояния задачи, возникающие в процессе ее решения;
- множество дуг между вершинами *A*, соответствующих возможным шагам в процессе решения задачи;
- множество начальных состояний задачи *S*;
- множество целевых состояний *D*.

Целевые состояния описываются набором некоторых измеряемых свойств состояний, встречающихся в процессе поиска, например, выигрышной комбинацией, или набором некоторых свойств путей, например, стоимостью перемещения по дугам.

Допустимым считается любой путь из вершины множества *S* в вершину множества *D*. Следует отметить, что множества *S* и *D* являются подмножествами множества *N*.

Одна из сложностей, возникающих при разработке алгоритма поиска допустимого пути на графе состояний, состоит в том, что одно и то же состояние может быть достигнуто разными путями, что может привести к возникновению петель. В результате возникает проблема выбора оптимального в некотором смысле пути. Например, граф пространства состояний игры в «крестики-нолики» корневой, т.к. имеет одно начальное состояние. Он не является деревом, т.к. очевидно, что некоторые состояния могут быть достигнуты разными путями, и не имеет петель, т.к. все дуги ориентированные и направлены от корневых вершин к потомкам. Такие графы часто встречаются при поиске в пространстве состояний.

Граф пространства состояний игры «8-головоломка» или игры в «пятнашки», приведенный на рис.3, тоже корневой, но, в отличие от предыдущего, может иметь циклы.

Цель игры – для начальной расстановки фишек с номерами от 1 до 8 найти такую последовательность их перемещения на пустое место, в результате которой фишки займут заданное положение, например, по возрастанию номеров слева направо и сверху вниз. Полное число возможных состояний достаточно велико (8!=40320), если учитывать симметричные отражения. Заметим, что определение очередного хода значительно удобнее задавать перемещением пустой клетки в одно из допустимых положений, как это показано на рис. 3.

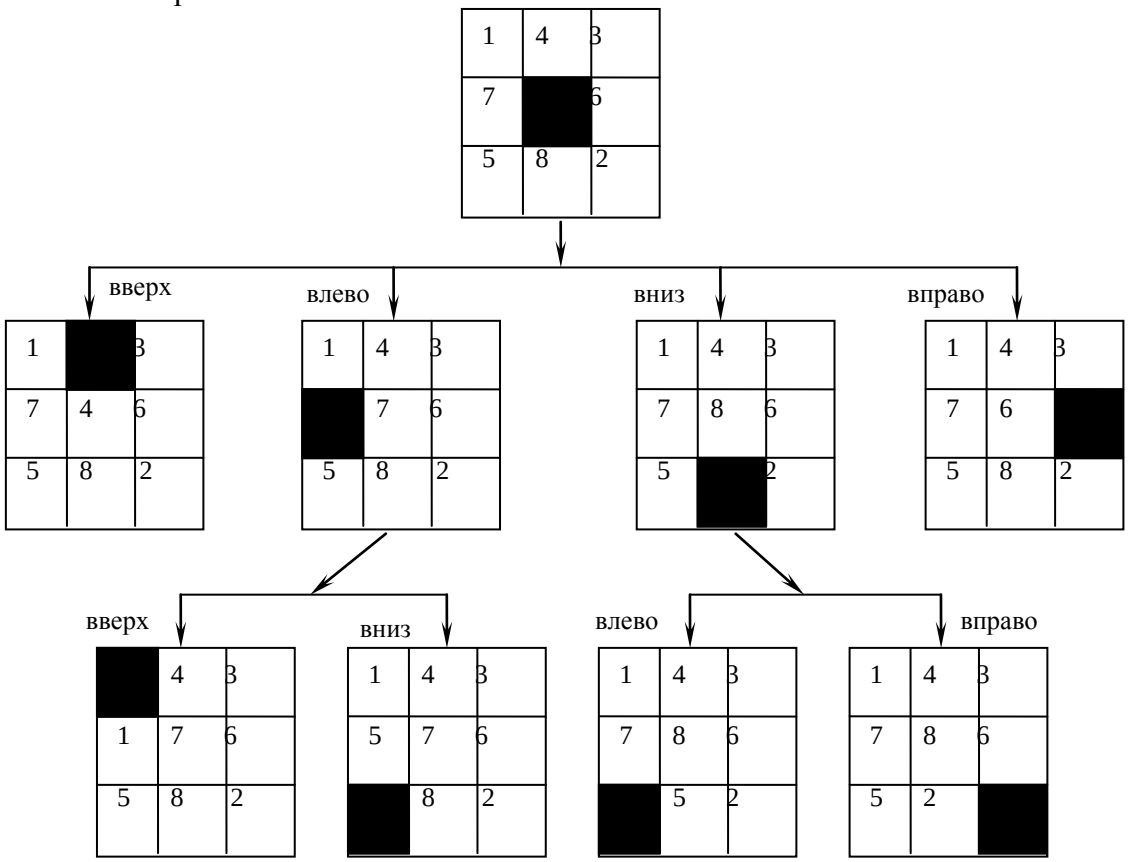


Рис.3 – Фрагмент пространства состояний «игры в пятнашки»

Следует отметить, что полное пространство состояний этой игры распадается на два несвязанных подграфа одинакового размера, из чего следует, что половина состояний недостижимы из любой заданной начальной вершины.

Для мира блоков, описанного в предыдущем разделе, тоже может быть сформирован граф пространства состояний, фрагмент которого приведен на рис. 4. Вершины графа соответствуют возможным состояниям мира блоков. Дуги графа помечены операциями, которые могут быть выполнены в состоянии, соответствующем родительской вершине. Планирование операций, которые необходимо выполнить для получения требуемого расположения блоков, можно представить как поиск пути на графе возможных состояний от текущего расположения блоков к целевому расположению.

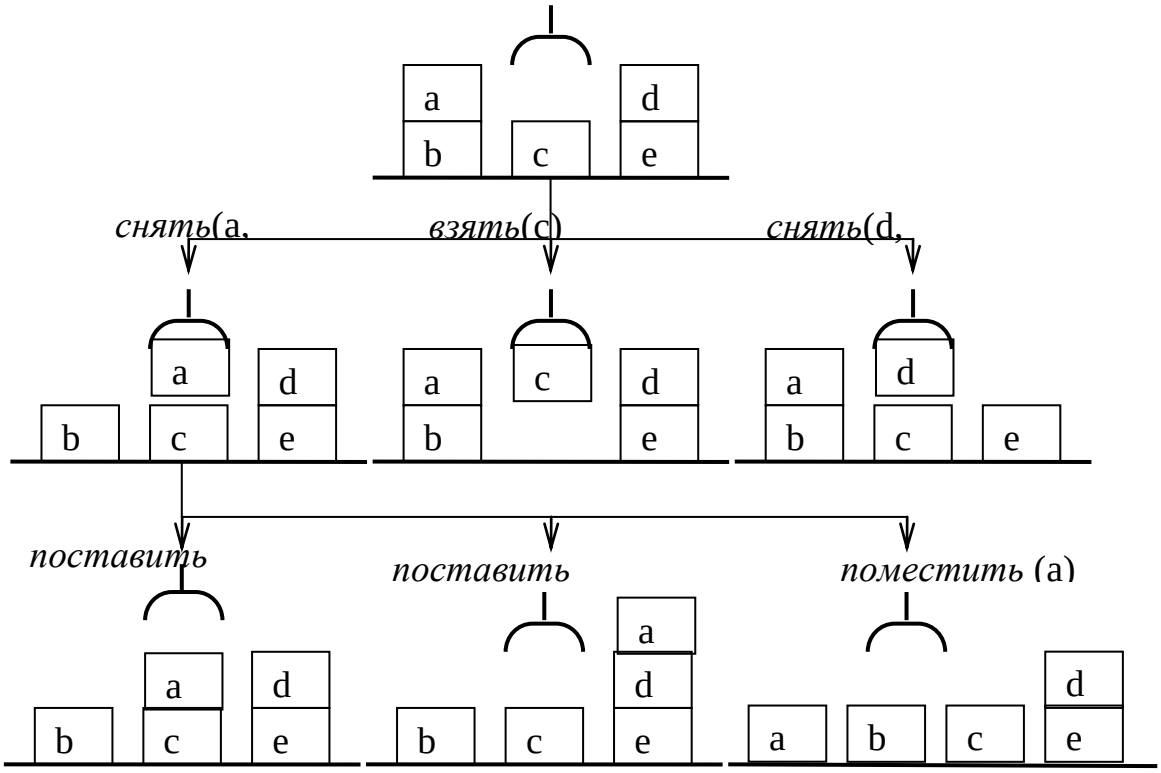


Рис.4 – Фрагмент графа пространства состояний мира блоков

Примером задачи, в которой цель задается свойством пути, является «задача коммивояжера», заключающаяся в том, что необходимо найти кратчайший путь соединяющий все вершины некоторого графа. Вершины этого графа – пункты, которые должен посетить коммивояжер, например, города. Помеченные дуги – возможность непосредственного перемещения между этими пунктами, например, длина пути между городами в километрах. На рис. 5 приведен такой граф для шести городов и фрагмент графа состояний задачи при условии, что начальный пункт – *A*.

Любой маршрут удобно представить последовательностью пунктов назначения, например, (*A, B, C, E, F, D*) и его длиной. Полное число вершин графа состояний составляет *N*!, где *N* – число пунктов назначения, (6!=720), при условии, что коммивояжер не должен возвращаться в исходный пункт, и каждый пункт имеет непосредственную связь со всеми другими пунктами. В данном случае целью является не некоторое состояние в пространстве поиска, а нахождение на графе состояний пути минимальной длины, что в принципе требует полного перебора всех состояний. Однако с ростом N число вершин графа состояний растет так быстро, что прямой перебор всех возможных вариантов становится неосуществимым.

Задача коммивояжера используется на практике, в том числе обеспечивает решение проблемы разводки электронных схем, задачи рентгеновской кристаллографии и маршрутизации при производстве СБИС.

Поиск в пространстве состояний можно вести как в прямом направлении, т.е. от исходных данных к цели, так и в обратном направлении. При прямом поиске к исходному состоянию задачи применяются возможные операции, приводящие к другим состояниям, причем процесс продолжается до получения требуемого состояния. При обратном поиске анализируются операции, которые могут привести к заданному состоянию, в результате

чего получается множество промежуточных целей. Процесс повторяется, пока не буден получено состояние, соответствующие исходному состоянию задачи.

Например, необходимо определить: является ли человек А прямым потомком человека В, при условии, что полные генеалогические деревья для А и В известны. Прямой поиск на таком графе требует просматривать всех потомков В, до тех пор пока не будет найден А. Если считать, что в генеалогическом дереве существует 10 поколений и каждая семья имеет в среднем трех детей, то при прямом поиске потребуется просмотреть до 3^{10} вершин. При обратном поиске потребуется просмотреть не более 2^{10} вершин, т.к. каждый человек имеет только прямых двух предков.

Процесс поиска от цели может быть более эффективен, чем прямой поиск в тех случаях когда, цель поиска явно сформулирована, имеется большое число правил, позволяющих эффективно ограничить множество возможных ветвей на каждом шаге, исходные данные приведены не полностью и могут добавляться в процессе решения задачи. Например, при диагностике болезни, как правило, в начальный момент имеются результаты далеко не всех возможных анализов и тестов. Рассматривая возможные диагнозы в качестве целей, можно проводить поиск в обратном направлении, получая необходимые анализы в зависимости от выбранной гипотезы о диагнозе для его подтверждения или отмены.

Любой поиск в пространстве состояний основан на алгоритме поиска с возвратами, который заключается в том, что в процессе поиска последовательно генерируются новые возможные состояния и формируются три списка состояний:

- список еще нерассмотренных состояний, для того чтобы можно было вернуться к любому из них;
- список просмотренных состояний, для того чтобы исключить их повторный просмотр;
- список узлов текущего просматриваемого пути, который возвращается в качестве результата поиска при достижении цели.

Каждое новое состояние, сгенерированное в процессе поиска, проверяется на вхождение в эти списки для предотвращения зацикливания.

Алгоритм поиска с возвратами запускается из начального состояния и следует по некоторому пути до тех пор, пока не будет достигнуто целевое состояние или концевая вершина графа состояний. В последнем случае алгоритм возвращается в какую-либо из пройденных вершин и просматривает ее вершины-братья, спускаясь по одной из ветвей.

Существует две возможные стратегии определения порядка просмотра вершин-потомков: поиск в глубину и поиск в ширину, а также основанный на сочетании этих стратегий метод итерационного заглубления.

При поиске в глубину необходимо сформировать всех потомков текущего состояния и затем рассматривать одного из них как следующее состояние. Если дальнейшие потомки не найдены, то рассматриваются вершины-братья. Например, на графе состояний, приведенном на рис.6, последовательность просмотра вершин, при условии, что цель – состояние U и очередность просмотра вершин-братьев соответствует порядку их генерации, т.е. по алфавиту, будет следующая: A,B,E,K,S,L,T, F,M,C,G,N,Y,J,P,U.

Как видно из алгоритма, поиск в глубину не гарантирует нахождение кратчайшего пути от начального состояния к цели. Его следует использовать, если известно, что путь к цели будет длинным, т.к. поиск в глубину не будет тратить время на просмотр большого количества близких к начальному состояний. Однако он может затеряться в глубинах графа, пропуская более короткие пути к цели или застрять на не ведущем к цели длинном пути. Поиск в глубину эффективен для графов с высокой степенью связности, т.к. он не требует запоминания всех узлов каждого уровня. В этом

случае степень использования пространства состояний является линейной функцией длины пути.

Поиск в ширину формирует всех потомков для всех имеющихся на текущем уровне вершин и просматривает их. Если цель не достигнута, то формируется следующий уровень вершин. Например, для графа на рис.6 последовательность просмотра вершин при поиске в ширину будет следующая: A,B,C,D,E,F,G,H,I,J,K, L,M,N,O,P,Q,R,S,T,U.

Из алгоритма видно, что поиск в ширину гарантирует нахождение кратчайшего пути от начального состояния к цели. Его следует использовать при низком коэффициенте ветвления, т.к. если состояния имеют высокое средне число потомков, то экспоненциальный характер зависимости числа узлов графа от числа уровней приводит к недопустимому увеличению пространства состояний.

Метод итерационного заглубления заключается в последовательном увеличении глубины поиска на единицу на каждой итерации. На первом шаге глубина поиска – 1 уровень, т.е. просматриваются все возможные состояния, являющиеся непосредственными потомками начального состояния. Если цель не найдена, то выполняется следующий шаг с предельной глубиной поиска – 2 и т.д. На каждой итерации алгоритм выполняет поиск в глубину с учетом текущего предельного числа уровней. Например, для графа на рис.6 последовательность просмотра вершин при поиске с итерационным заглублением будет следующая: A,B,C,D,E, K,L,F,M,C,G,N,H,O,P,I,Q,J,R,S,T,U. Как видно, последовательность просмотра вершин отличается от последовательности их просмотра при поиске в ширину, хотя и в том и в другом подходе цель достигается после просмотра всех узлов.

Поиск с итерационным заглублением гарантирует нахождение кратчайшего пути от начального состояния к цели, т.к. в процессе поиска узлы просматриваются по уровням.

Все описанные методы поиска представляют собой итерационные, т.е. пошаговые процессы, причем на каждом шаге необходимо проверять принадлежность нового состояния одному из списков для исключения зацикливания. Наиболее эффективным и компактным алгоритмом, решающим эту задачу, является рекурсивный алгоритм.

2.7 Лабораторная работа №7 (2 часа).

Тема: «Персептрон. Обучение персептрона**»**

2.7.1 Цель работы: Изучить алгоритм обучения персептрона.

2.7.2 Задачи работы:

1. Алгоритм обучения персептрона.

2. Правило корректировки весов.

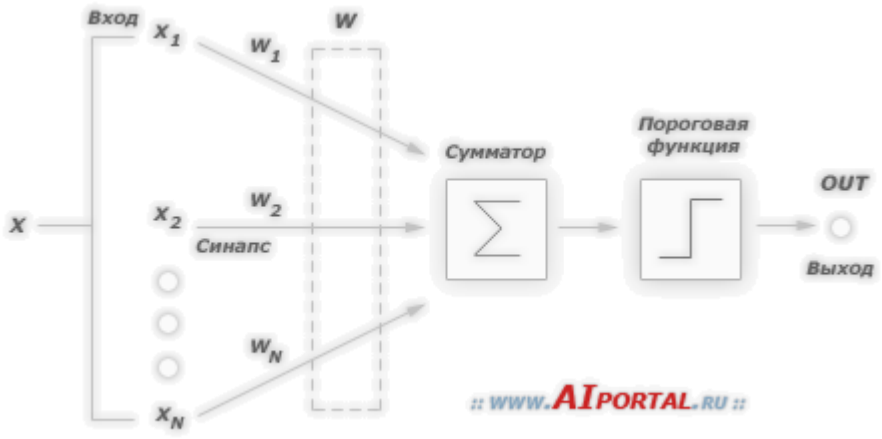
3. Дельта правило.

2.7.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. ПЭВМ

2.7.4 Описание (ход) работы:

Допустим, есть следующая модель персептрона:



Алгоритм обучения персептрона следующий:

1. Присвоить синаптическим весам $w_1, w_2, ..., w_N$ некоторые начальные значения. Например, нулю.
2. Подать входной образ X и вычислить OUT. Если OUT правильный, то переходят к шагу 4. Иначе к шагу 3.
3. Применяя дельта-правило (см. ниже) вычислить новые значения синаптических весов.
4. Повторить шаги 2-4 данного алгоритма обучения персептрона пока сеть не станет выдавать ожидаемый выход на векторах из обучающей выборки или пока отклонение не станет ниже некоторого порога.

Т.о. образом логика обучения персептрона следующая: если сигнал персептрона при некотором образе верен, то ничего корректировать не надо, если нет – производится корректировка весов.

Правила корректировки весов следующие:

1. Если OUT неверен и равен нулю, то необходимо увеличить веса тех входов, на которые была подана единица.
2. Если OUT неверен и равен единице, то необходимо уменьшить веса тех входов, на которые была подана единица.

Поясним эти правила. Допустим, что на вход был подан некоторый обучающий двоичный вектор X. Этому вектору соответствует выход OUT равный единице. И этот выход неправильный. Тогда веса, присоединенные к единичным входам, должны быть уменьшены, так как они стремятся дать неверный результат. Аналогично, если некоторому другому обучающему вектору X соответствует неправильный выход OUT равный нулю, то веса, присоединенные к единичным входам, должны быть уже уменьшены.

Дельта-правило

Дельта-правило является математической моделью правил корректировки весов. Введем величину δ, которая равна разности между требуемым T и реальным OUT выходом:

$$\delta = T - OUT$$

Тогда, веса персептрона после коррекции будут равны:

$$w_N(i+1) = w_N(i) + \eta \delta x_N$$

где:

- i – номер текущей итерации обучения персептрона;
- η (греческая буква «эта») – коэффициент **скорости обучения**, позволяет управлять средней величиной изменения весов;

- x_N – величина входа соответствующая w_N синаптическому весу. Добавление величины x_N в произведение позволяет избежать изменение тех весов, которым на входе соответствовал ноль.

Существует доказательство сходимости этого алгоритма обучения персептрона за конечное число шагов. Модель персептрона в настоящий момент представляет больше историческую ценность, чем практическую. Но именно на его примере удается более наглядно показать некоторые важные принципы функционирования, в том числе и обучения нейронных сетей.

2.8 Лабораторная работа №8 (2 часа).

Тема: «Многослойный персептрон. Метод обратного распространения ошибки»

2.8.1 Цель работы: изучить метод обратного распространения ошибки

2.8.2 Задачи работы:

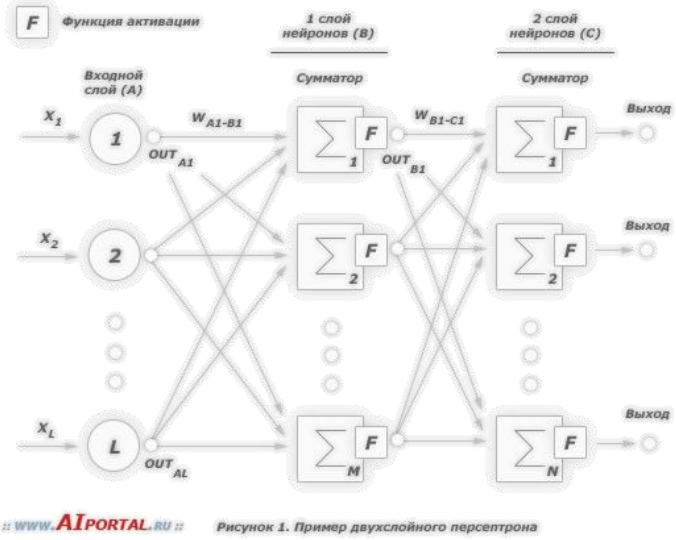
1. Обучить следующую нейронную сеть, применив алгоритм обратного распространения ошибки.

2.8.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. ПЭВМ

2.8.4 Описание (ход) работы:

Допустим необходимо обучить следующую нейронную сеть, применив алгоритм обратного распространения ошибки:



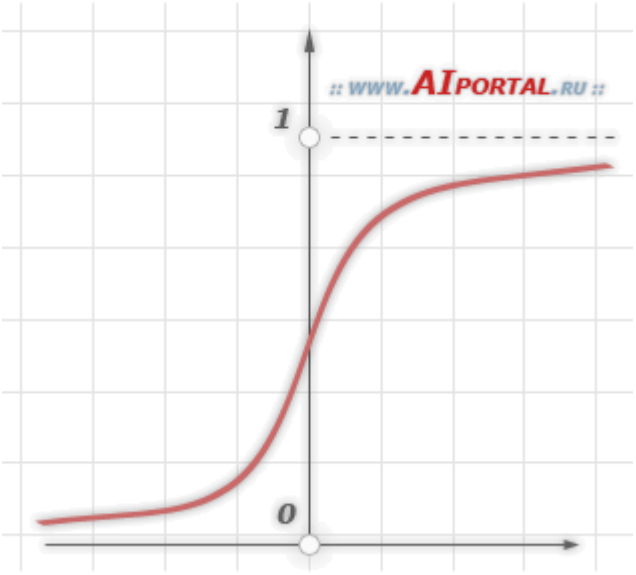
На приведенном рисунке использованы следующие условные обозначения:

- каждому слою нейронной сети соответствует своя буква, например: входному слою соответствует буква A, а выходному – C;
- все нейроны каждого слоя пронумерованы арабскими цифрами;
- $w_{A1\text{-}B1}$ – синаптический вес между нейронами $A1$ и $B1$;
- OUT_{A1} – выход нейрона $A1$.

В качестве активационной функции в многослойных персептронах, как правило, используется сигмоидальная активационная функция, в частности логистическая:

$$OUT = \frac{1}{1 + \exp\left(- \alpha Y\right)}$$

где α – параметр наклона сигмоидальной функции. Изменяя этот параметр, можно построить функции с различной крутизной. Оговоримся, что для всех последующих рассуждений будет использоваться именно логистическая функция активации, представленная только, что формулой выше.



Сигмоид сужает диапазон изменения так, что значение OUT лежит между нулем и единицей. Многослойные нейронные сети обладают большей представляющей мощностью, чем однослойные, только в случае присутствия нелинейности. Сжимающая функция обеспечивает требуемую нелинейность. В действительности имеется множество функций, которые могли бы быть использованы. Для алгоритма обратного распространения ошибки требуется лишь, чтобы функция была всюду дифференцируема. Сигмоид удовлетворяет этому требованию. Его дополнительное преимущество состоит в автоматическом контроле усиления. Для слабых сигналов (т.е. когда OUT близко к нулю) кривая вход-выход имеет сильный наклон, дающий большое усиление. Когда величина сигнала становится больше, усиление падает. Таким образом, большие сигналы воспринимаются сетью без насыщения, а слабые сигналы проходят по сети без чрезмерного ослабления.

Целью обучения сети алгоритмом обратного распространения ошибки является такая подстройка ее весов, чтобы приложение некоторого множества входов приводило к требуемому множеству выходов. Для краткости эти множества входов и выходов будут называться векторами. При обучении предполагается, что для каждого входного вектора существует парный ему целевой вектор, задающий требуемый выход. Вместе они называются обучающей парой. Сеть обучается на многих парах.

Алгоритм обратного распространения ошибки следующий:

1. Инициализировать синаптические веса маленькими случайными значениями.
2. Выбрать очередную обучающую пару из обучающего множества; подать входной вектор на вход сети.
3. Вычислить выход сети.
4. Вычислить разность между выходом сети и требуемым выходом (целевым вектором обучающей пары).
5. Подкорректировать веса сети для минимизации ошибки (как см. ниже).

6. Повторять шаги с 2 по 5 для каждого вектора обучающего множества до тех пор, пока ошибка на всем множестве не достигнет приемлемого уровня.

Операции, выполняемые шагами 2 и 3, сходны с теми, которые выполняются при функционировании уже обученной сети, т.е. подается входной вектор и вычисляется получающийся выход. Вычисления выполняются послойно. На рис. 1 сначала вычисляются выходы нейронов слоя B (слой A входной, а значит никаких вычислений в нем не происходит), затем они используются в качестве входов слоя C, вычисляются выходы OUT_{CN} нейронов слоя C, которые и образуют выходной вектор сети OUT.

Шаги 2 и 3 образуют так называемый «проход вперед», так как сигнал распространяется по сети от входа к выходу.

Шаги 4 и 5 составляют «обратный проход», здесь вычисляемый сигнал ошибки распространяется обратно по сети и используется для подстройки весов.

Рассмотрим подробней 5 шаг – корректировка весов сети. Здесь следует выделить два нижеописанных случая.

Случай 1. Корректировка синаптических весов выходного слоя

Например, для модели нейронной сети на рис. 1, это будут веса имеющие следующие обозначения: w_{B1-C1} и w_{B2-C1}. Определимся, что индексом p будем обозначать нейрон, из которого выходит синаптический вес, а q – нейрон в который входит:

Введем величину δ, которая равна разности между требуемым T_q и реальным OUT_q выходами, умноженной на производную логистической функции активации (формулу логистической функции активации см. выше):

$$\delta_q = OUT_q\left(1 - OUT_q\right)\left(T_q - OUT_q\right)$$

Тогда, веса выходного слоя после коррекции будут равны:

$$w_{p-q}\left(i+1\right) = w_{p-q}\left(i\right) + \eta\delta_q OUT_p$$

где:

- i – номер текущей итерации обучения;
- w_{p-q} – величина синаптического веса, соединяющего нейрон p с нейроном q;
- η (греческая буква «эта») – коэффициент «скорости обучения», позволяет управлять средней величиной изменения весов;
- OUT_p – выход нейрона p.

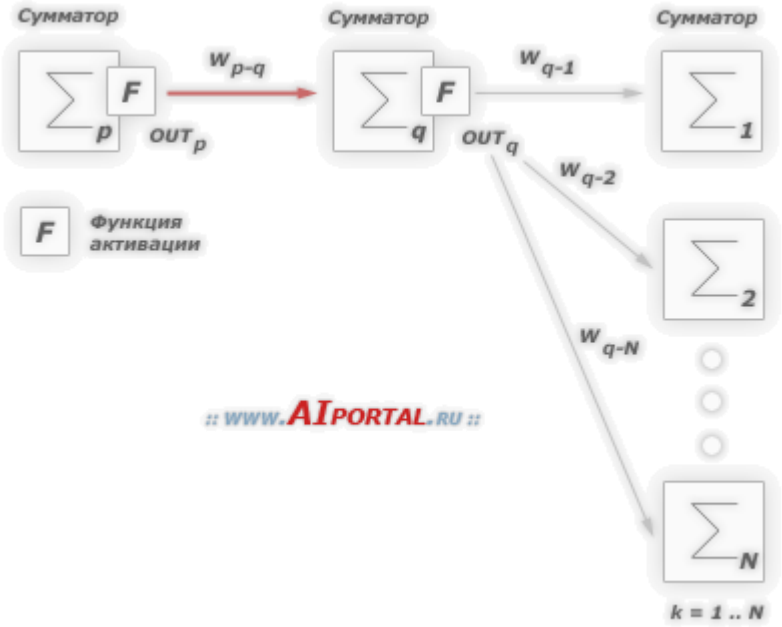
Приведем пример вычислений для синаптического веса w_{B1-C1} :

$$\delta_{C1} = OUT_{C1}\left(1 - OUT_{C1}\right)\left(T_1 - OUT_{C1}\right)$$

$$w_{B1\text{-}C1}\left(i+1\right) = w_{B1\text{-}C1}\left(i\right) + \eta\delta_{C1}OUT_{B1}$$

Случай 2. Корректировка синаптических весов скрытого слоя

Для модели нейронной сети на рис. 1, это будут веса соответствующие слоям A и B. Определимся, что индексом p будем обозначать нейрон из которого выходит синаптический вес, а q – нейрон в который входит (обратите внимание на появление новой переменной k):



Введем величину δ, которая равна:

$$\delta_q = OUT_q\left(1 - OUT_q\right)\sum_{k=1}^{M}\delta_k w_{q\text{-}k}$$

где:

- $\sum_{k=1}^{N}$ – сумма от 1 по N.

Тогда, веса скрытых слоев после коррекции будут равны:

$$w_{p\text{-}q}\left(i+1\right) = w_{p\text{-}q}\left(i\right) + \eta\delta_q OUT_p$$

Приведем пример вычислений для синаптического веса $w_{A1\text{-}B1}$:

$$\delta_{B1} = OUT_{B1}\left(1 - OUT_{B1}\right)$$

$$w_{A1\text{-}B1}\left(i+1\right) = w_{A1\text{-}B1}\left(i\right) + \eta\delta_{B1}OUT_{A1}$$

Для каждого нейрона в скрытом слое должно быть вычислено δ и подстроены все веса, ассоциированные с этим слоем. Этот процесс повторяется слой за слоем по направлению к входу, пока все веса не будут подкорректированы.

2.9 Лабораторная работа №9 (2 часа).

Тема: «Разработка генетического алгоритма. Тестирование генетического алгоритма**»**

2.9.1 Цель работы: изучение процесса разработки генетического алгоритма

2.9.2 Задачи работы:

1. Разработка генетического алгоритма настройки искусственной нейронной сети\

2. Анализ современных методов искусственного интеллекта, включающих генетические алгоритмы и искусственные нейронные сети.

2.9.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. ПЭВМ

2.9.4 Описание (ход) работы:

Постановка задачи оптимизации включает в себя множество допустимых решений $X = \{x_1, x_2, \ldots, x_n\}$ и числовую функцию, определенную на этом множестве, которая называется целевой функцией.

Нельзя отождествлять критерий (критерии) оптимальности и целевую функцию.

Целевая функция – это аналитическая зависимость между критерием (критериями) оптимальности и подлежащими оптимизации параметрами с указанием направления экстремума.

Отличие понятий «критерий» и «целевая функция» состоит в следующем:

1. Целевая функция может включать в себя более одного критерия.
2. Для целевой функции всегда и обязательно указывается вид экстремума:

Различают два вида задач оптимизации:

1. Задачу минимизации.
2. Задачу максимизации.

Чтобы решить задачу минимизации функции на множестве, необходимо найти такой вектор (а также соответствующее значение целевой функции), чтобы неравенство: выполнялось для всех. При этом называют оптимальным решением (точнее здесь – минимальным решением), а - оптимумом (минимумом).

Чтобы решить задачу максимизации функции на множестве, необходимо найти такой вектор (а также соответствующее значение целевой функции), чтобы неравенство: выполнялось для всех. При этом называют оптимальным (максимальным) решением, а – оптимумом (максимумом).

В общем виде находится именно вектор, т.к., например, при решении двухпараметрической задачи, он будет включать в себя два параметра, трехпараметрической – три параметра и т.д.

Рассмотрим Диофантово (только целые решения) уравнение: a+2b+3c+4d=30, где a, b, c и d - некоторые положительные целые. Применение ГА за очень короткое время находит искомое решение (a, b, c, d).

Конечно, Вы можете спросить: почему бы не использовать метод грубой силы: просто не подставить все возможные значения a, b, c, d (очевидно, 1 <= a,b,c,d <= 30) ?

Архитектура ГА-систем позволяет найти решение быстрее за счет более 'осмысленного' перебора. Мы не перебираем все подряд, но приближаемся от случайно выбранных решений к лучшим.

Для начала выберем 5 случайных решений: 1 =< a,b,c,d =< 30. Вообще говоря, мы можем использовать меньшее ограничение для b,c,d, но для упрощения пусть будет 30 (таблица 6).

Таблица 6 – Первое поколение хромосом

Хромосома	(a,b,c,d)
1	(1,28,15,3)
2	(14,9,2,4)
3	(13,5,7,3)
4	(23,8,16,19)
5	(9,13,5,2)

Чтобы вычислить коэффициенты выживаемости (fitness), подставим каждое решение в выражение a+2b+3c+4d. Расстояние от полученного значения до 30 и будет нужным значением (таблица 7).

Таблица 7 - Коэффициенты выживаемости первого поколения хромосом (набора решений)

Хромосома	Коэффициент выживаемости
1	|114-30|=84
2	|54-30|=24
3	|56-30|=26
4	|163-30|=133
5	|58-30|=28

Так как меньшие значения ближе к 30, то они более желательны. В нашем случае большие численные значения коэффициентов выживаемости подходят, увы, меньше. Чтобы создать систему, где хромосомы с более подходящими значениями имеют большие шансы оказаться родителями, мы должны вычислить, с какой вероятностью (в %) может быть выбрана каждая. Одно решение заключается в том, чтобы взять сумму обратных значений коэффициентов, и исходя из этого вычислять проценты (таблица 8). Все решения были сгенерированы Генератором Случайных Чисел – ГСЧ.

Таблица 8 - Вероятность оказаться родителем

Хромосома	Подходимость
1	(1/84)/0.135266 = 8.80%
2	(1/24)/0.135266 = 30.8%
3	(1/26)/0.135266 = 28.4%
4	(1/133)/0.135266 = 5.56%
5	(1/28)/0.135266 = 26.4%

Для выбора 5-и пар родителей (каждая из которых будет иметь 1 потомка, всего - 5 новых решений), представим, что у нас есть 10000-стонняя игральная кость, на 880 сторонах отмечена хромосома 1, на 3080 - хромосома 2, на 2640 сторонах - хромосома 3, на 556 - хромосома 4 и на 2640 сторонах отмечена хромосома 5. Чтобы выбрать первую пару, кидаем кость два раза и выбираем выпавшие хромосомы. Таким же образом выбирая остальных, получаем результат в таблице 9.

Таблица 9 - Симуляция выбора родителей

Хромосома отца	Хромосома матери
3	1
5	2
3	5
2	5
5	3

Каждый потомок содержит информацию о генах и отца и от матери. Вообще говоря, это можно обеспечить различными способами, однако в нашем случае можно использовать т.н. "кроссовер" (cross-over). Пусть мать содержит следующий набор решений: a1,b1,c1,d1, а отец - a2,b2,c2,d2, тогда возможно 6 различных кроссоверов (| = разделительная линия) (таблица 10):

Таблица 10 - Кроссоверы между родителями

Хромосома-отец	Хромосома-мать	Хромосома-потомок
a1 | b1,c1,d1	a2 | b2,c2,d2	a1,b2,c2,d2 or a2,b1,c1,d1
a1,b1 | c1,d1	a2,b2 | c2,d2	a1,b1,c2,d2 or a2,b2,c1,d1
a1,b1,c1 | d1	a2,b2,c2 | d2	a1,b1,c1,d2 or a2,b2,c2,d1

Есть достаточно много путей передачи информации потомку, и кроссовер - только один из них. Расположение разделителя может быть абсолютно произвольным, как и то, отец или мать будут слева от черты.

А теперь попробуем проделать это с нашими потомками и выведем результаты в таблицу 11.

Таблица 11 - Симуляция кроссоверов хромосом родителей

Хромосома-отец	Хромосома-мать	Хромосома-потомок
(13 | 5,7,3)	(1 | 28,15,3)	(13,28,15,3)
(9,13 | 5,2)	(14,9 | 2,4)	(9,13,2,4)
(13,5,7 | 3)	(9,13,5 | 2)	(13,5,7,2)
(14 | 9,2,4)	(9 | 13,5,2)	(14,13,5,2)
(13,5 | 7, 3)	(9,13 | 5, 2)	(13,5,5,2)

Теперь мы можем вычислить коэффициенты выживаемости (fitness) потомков (таблица 12).

Таблица 12 - Коэффициенты выживаемости потомков (fitness)

Хромосома-потомок	Коэффициент выживаемости
(13,28,15,3)	|126-30|=96
(9,13,2,4)	|57-30|=27
(13,5,7,2)	|57-30|=22
(14,13,5,2)	|63-30|=33
(13,5,5,2)	|46-30|=16

Средняя приспособленность (fitness) потомков оказалась 38.8, в то время как у родителей этот коэффициент равнялся 59.4. Следующее поколение может мутировать.

Например, мы можем заменить одно из значений какой-нибудь хромосомы на случайное целое от 1 до 30. Продолжая, таким образом, одна хромосома, в конце концов, достигнет коэффициента выживаемости 0, то есть станет решением. Системы с большей популяцией (например, 50 вместо 5-и) сходятся к желаемому уровню (0) более быстро и стабильно.

Работа ГА представляет собой итер-ый процесс, который продолжается до тех пор, пока поколения не перестанут существенно отличаться друг от друга, или не пройдет заданное кол-во поколений или заданное время. Для каждого поколения реализуются отбор, кроссовер (скрещивание) и мутация. Рас-м этот алгоритм.

Шаг 1: генерируется начальная популяция, состоящая из ***N*** особей со случайными наборами признаков.

Шаг 2 (борьба за существование): вычисляется *абсолютная приспособленность* каждой особи популяции к условиям среды *f(i)* и суммарная приспособленность особей популяции, характеризующая приспособленность всей популяции. Затем при ***пропорциональном отборе*** для каждой особи вычисляется ее *относительный вклад в суммарную приспособленность популяции* $P_s(i)$, т.е. относительные ее абсолютные приспособленности *f(i)* к суммарной приспособленности всех особей популяции (1):

$$P_s(i) = \frac{f(i)}{\sum_{i=1}^{N} f(i)} \qquad (1)$$

В выражении (1) сразу обращает на себя внимание возможность сравнения абсолютной приспособленности *i-й* особи *f(i)* не с суммарной приспособленностью всех особей популяции, а со средней абсолютной приспособленностью особ и популяции (2):

$$\bar{f} = \frac{1}{N}\sum_{i=1}^{N} f(i) \qquad (2)$$

Тогда получим (3):

$$P(i) = \frac{f(i)}{\bar{f}} = \frac{f(i)}{\frac{1}{N}\sum_{i=1}^{N} f(i)} \qquad (3)$$

Если взять логарифм по основанию 2 от выражения (3), то получим ***количество информации, содержащееся в признаках особи о том, что она выживет и даст потомство*** (4).

$$I(i) = Log_2 \frac{f(i)}{\bar{f}} \qquad (4)$$

Формально ***приспособленность особи представляет собой количество инф-и, содержащееся в ее фенотипе о продолжении ее генотипа в последующих поколениях.*** Т.к. кол-во потомства особи пропорционально ее приспособленности, то м. считать, что *если это кол-во инф-и:*

– *положительно,* то данная особь выживает и дает потомство, численность кот, пропорциональна этому количеству информации;

– *равно нулю,* то особь доживает до половозрелого возраста, но потомства не дает (его численность равна нулю);

– *меньше нуля,* то особь погибает до достижения половозрелого возраста.