КОНСПЕКТ ЛЕКЦИЙ
по дисциплине
ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМЫ И ТЕХНОЛОГИИ
для студентов 4 курса по направлению 230400 – Информационные системы и технологии

План:

1. Введение вискусственный интеллект

История развития искусственного интеллекта как науки; Определение искусственного интеллекта; История развития искусственного интеллекта; Задачи искусственного интеллекта; Направления и подходы к исследованиям в области искусственного интеллекта; Классификация интеллектуальных информационных систем.

2. Основы теории искусственного интеллекта

Представление знаний; Данные и знания; Классификация моделей представления знаний; Нейронные сети; Классификация искусственных нейронных сетей; Однослойные искусственные нейронные сети; Многослойные нейронные сети; Задачи, решаемые нейронными сетями; Эволюционное моделирование; Генетические алгоритмы; Виды генетических алгоритмов; Нечеткие множества и нечеткая логика; Теория нечетких множеств; Нечеткая логика.

3. Интеллектуальные информационные системы

Экспертные системы; Модель экспертных систем; Классификация экспертных систем и оболочек экспертных систем; Средства разработки экспертных систем; Системы поддержки принятия решений; Структура систем поддержки принятия решений; Классификация систем поддержки принятия решений.

СОДЕРЖАНИЕ

Часть 1. ВВЕДЕНИЕ В ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ

§1. История развития искусственного интеллекта как науки

Определение искусственного интеллекта

Термин «искусственный интеллект» (artificialintelligence) был предложен в 1956 году. Слово intelligence означает «умение рассуждать разумно», а вовсе не «интеллект», для которого есть термин intellect.

Искусственный интеллект (ИИ) занимается изучением разумного поведения (у людей, животных и машин) и пытается найти способы моделирования подобного поведения в любом типе искусственно созданного механизма. Несмотря на то что термину больше полувека, единого определения его не существует. Разные исследователи по-разному определяют эту науку, в зависимости от своего взгляда на нее, и работают над созданием систем, которые:

- думают подобно людям;
- думают рационально;
- действуют подобно людям;
- действуют рационально.

Синтезируя десятки определений искусственного интеллекта из различных источников, в качестве рабочего определения можно предложить следующее.

Искусственный интеллект – это одно из направлений информатики, целью которого является разработка аппаратно-программных средств, позволяющих пользователю-непрограммисту ставить и решать свои, традиционно считающиеся интеллектуальными, задачи, общаясь с ЭВМ на ограниченном подмножестве естественного языка.

При воссоздании разумных рассуждений и действий возникают определенные трудности. Во-первых, в большинстве случаев, выполняя какие-то действия, человек не осознает, как это делает, не известен точный способ, метод или алгоритм понимания текста, распознавания лиц,

доказательства теорем, решения задач, сочинения стихов и т.д. Во-вторых, на современном уровне развития компьютер слишком далек от человеческого уровня компетентности и работает по другим принципам.

Искусственный интеллект всегда был междисциплинарной наукой, являясь одновременно и наукой и искусством, и техникой и психологией. Методы искусственного интеллекта разнообразны. Они активно заимствуются из других наук, адаптируются и изменяются под решаемую задачу. Для создания интеллектуальной системы необходимо привлекать специалистов из прикладной области, поэтому в рамках искусственного интеллекта сотрудничают лингвисты, нейрофизиологи, психологи, экономисты, информатики, программисты и т.д.

История развития искусственного интеллекта

Идея создания искусственного подобия человека для решения сложных задач и моделирования человеческого разума витала в воздухе еще в древнейшие времена. Так, в Древнем Египте была создана «оживающая» механическая статуя бога Амона. У Гомера в «Илиаде» бог Гефест ковал человекоподобных существ.

Искусственный интеллект является в некотором смысле наукой будущего, в которой нет жесткого разделения по областям и ясно видна связь между отдельными дисциплинами, которые лишь отражают определенную грань познания.

Точный свод законов, руководящих рациональной частью мышления, был сформулирован Аристотелем (384-322 гг. до н.э.). Однако родоначальником искусственного интеллекта считается средневековый испанский философ, математик и поэт Раймонд Луллий, который еще в XIII веке попытался создать механическую машину для решения различных задач на основе разработанной им всеобщей классификации понятий. В XVIII веке Лейбниц и Декарт независимо друг от друга продолжили эту идею, предложив универсальные языки классификации всех наук. Труды этих ученых можно считать первыми теоретическими работами в области искусственного интеллекта.Теория игр и теория принятия решений, данные о строении мозга, когнитивная психология – все это стало строительным материалом для искусственного интеллекта.

Но окончательное рождение искусственного интеллекта как научного направления произошло только после создания ЭВМ в 40-х годах XX века и выпуска Норбертом Винером основополагающих работ по новой науке – кибернетике.

Формирование искусственногоинтеллекта как науки произошло в **1956 году.** Д. Маккарти, М. Минский, К. Шеннон и Н. Рочестер организовали двухмесячный семинар в Дартмуте для американских исследователей, занимающихся теорией автоматов, нейронными сетями, интеллектом. Хотя исследования в этой области уже активно велись, но именно на этом семинаре появились термин и отдельная наука – искусственный интеллект.

Одним из основателей теории искусственного интеллекта считается известный английский ученый Алан Тьюринг, который в 1950 году опубликовал статью «Вычислительные машины и разум» (переведенную на русский язык под названием «Может ли машина мыслить?»). Именно в ней описывался ставший классическим «тест Тьюринга», позволяющий оценить «интеллектуальность» компьютера по его способности к осмысленному диалогу с человеком.

Первые десятилетия развития искусственного интеллекта (1952-1969 гг.)были полны успехов и энтузиазма. А. Ньюэлл, Дж. Шоу и Г. Саймон создали программу для игры в шахматы на основе метода, предложенного в 1950 году К. Шенноном, формализованного А. Тьюрингом и промоделированного им же вручную. К работе была привлечена группа голландских психологов под руководством А. де Гроота, изучавших стили игры выдающихся шахматистов. В 1956 году этим коллективом был создан язык программирования ИПЛ1 – практически первый символьный язык обработки списков и написана первая программа «Логик-Теоретик», предназначенная для автоматического доказательства теорем в исчислении высказываний. Эту программу можно отнести к первым достижениям в области искусственного интеллекта.

В 1960 году этой же группой была написана программа GPS (GeneralProblemSolver) – универсальный решатель задач. Она могла решать ряд головоломок, вычислять неопределенные интегралы, решать некоторые другие задачи. Результаты привлекли внимание специалистов в

области вычислений, и появились программы автоматического доказательства теорем из планиметрии и решения алгебраических задач.

С 1952 года А. Самюэл написал ряд программ для игры в шашки, которые играли на уровне хорошо подготовленного любителя, причем одна из них научилась играть лучше, чем ее создатель.

В 1958 году Д. Маккарти определил новый язык высокого уровня Lisp, который стал доминирующим для искусственного интеллекта.

Первые нейросети появились в конце 50-х годов. В 1957 году Ф. Розенблаттом была предпринята попытка создать систему, моделирующую человеческий глаз и его взаимодействие с мозгом, – персептрон.

Первая международная конференция по искусственному интеллекту (IJCAI) состоялась в 1969 году в Вашингтоне.

В 1963 году Д. Робинсон реализовал метод автоматического доказательства теорем, получивший название «принцип резолюции», и на основе этого метода в 1973 году был создан язык логического программирования Prolog.

В США появились первые коммерческие системы, основанные на знаниях, – *экспертные системы*. Происходит коммерциализация искусственного интеллекта. Растут ежегодные капиталовложения и интерес к *самообучающимся системам*, создаются промышленные экспертные системы. Разрабатываются методы представления знаний.

Первая экспертная система была создана Э. Фейгенбаумом в 1965 году. Но до коммерческой прибыли было еще далеко. Лишь в 1986 году первая коммерческая система R1 компании DEC позволила сэкономить примерно 40 миллионов долларов за год. К 1988 году компанией DEC было развернуто 40 экспертных систем. В компании DuPont применялось 100 систем, и экономия составляла примерно 10 миллионов в год.

В 1981 году Япония, в рамках 10-летнего плана по разработке интеллектуальных компьютеров на базе Prolog, приступила к разработке компьютера 5-го поколения, основанного на знаниях. 1986 год стал годом возрождения интереса к нейронным сетям.

В 1991 году Япония прекращает финансирование проекта компьютера 5-го поколения и начинает проект создания компьютера 6-го поколения – нейрокомпьютера.

В 1997 году компьютер «ДипБлю» победил в игре в шахматы чемпиона мира Г. Каспарова, доказав возможность того, что

искусственный интеллект может сравняться с человеком или превзойти его в ряде интеллектуальных задач (пусть и в ограниченных условиях).

Огромную роль в борьбе за признание искусственного интеллекта в нашей стране сыграли академики А. И. Берг и Г. С. Поспелов.

В 1954-1964 гг. создаются отдельные программы и проводятся исследования в области поиска решения логических задач. Создается программа АЛПЕВ ЛОМИ, автоматически доказывающая теоремы. Она основана на оригинальном обратном выводе Маслова, аналогичном методу резолюций Робинсона. Среди наиболее значимых результатов, полученных отечественными учеными в 60-е годы, следует отметить алгоритм «Кора» М. М. Бонгарда, моделирующий деятельность человеческого мозга при распознавании образов. Большой вклад в становление российской школы искусственного интеллекта внесли выдающиеся ученые М. Л. Цетлин, В. Н. Пушкин, М. А. Гаврилов, чьи ученики и явились пионерами этой науки в России.

В 1964 году предлагался метод автоматического поиска доказательства теорем в исчислении предикатов, получивший название «обратный метод Маслова».

В 1965-1980 гг. произошло рождение нового направления – ситуационного управления (в западной терминологии соответствует представлению знаний). Основателем этой научной школы стал профессорД. А. Поспелов.

В Московском государственном университете в 1968 году В.Ф. Турчиным был создан язык символьной обработки данных РЕФАЛ.

Задачи искусственного интеллекта

Искусственный интеллект преследует множество целей. Одной из основных задач искусственного интеллекта является создание полного научного описания интеллекта человека, животного и машины и вычисление принципов, общих для всех троих. Моделирование разума необходимо для решения задач. К интеллектуальным задачам можно отнести все задачи, алгоритм нахождения которых неизвестен. Но, например, перебор всех возможных комбинаций также является алгоритмом. Применить его на практике, к сожалению, на современном

уровне развития техники к большинству задач невозможно (современная ЭВМ не может сгенерировать все простые перестановки более чем 12-ти разных предметов, так как этих перестановок более 479 млн).

Комбинаторный взрыв, с которым столкнулись исследователи уже в ранних исследованиях, – пример этого. В таких случаях, когда незначительное увеличение входных данных задачи ведет к возрастанию количества повторяющихся действий в степенной зависимости, говорят о неполиномиальных алгоритмах, которые характеризуются тем, что количество операций в них возрастает в зависимости от числа входов по закону, близкому к экспоненте. Подобные алгоритмы решения имеет чрезвычайно большой круг задач, особенно комбинаторных проблем, связанных с нахожденим сочетаний, перестановок, размещений каких-либо объектов.Поэтому труднорешаемой (нерешаемой) задачей можно называть такую задачу, для которой не существует эффективного алгоритма решения. Экспоненциальные алгоритмы решений, в том числе и исчерпывающие, абсолютно неэффективны для случаев, когда входные данные меняются в достаточно широком диапазоне значений, следовательно, в общем случае считать их эффективными нельзя.

Эффективный алгоритм имеет не настолько резко возрастающую зависимость количества вычислений от входных данных, например ограниченно полиномиальную, то есть x находится в основании, а не в показателе степени. Такие алгоритмы называются полиномиальными, и, как правило, если задача имеет полиномиальный алгоритм решения, то она может быть решена на ЭВМ с большой эффективностью. К таким можно отнести задачи сортировки данных, многие задачи математического программирования и т.п.

Следовательно, современный компьютер не может выполнить решение полностью аналитически. Возможна замена аналитического решения численным алгоритмом, который итеративно (то есть циклически повторяя операции) или рекурсивно (вызывая процедуру расчета из самой себя) выполняет операции, шаг за шагом приближаясь к решению. Если число этих операций возрастает, время выполнения, а возможно, и расход других ресурсов (например, ограниченной машинной памяти), также возрастает, стремясь к бесконечности. Задачи, своими алгоритмами решения создающие предпосылки для резкого возрастания использования

ресурсов, в общем виде не могут быть решены на цифровых вычислительных машинах, так как ресурсы всегда ограничены.

Решением подобных задач и занимается искусственный интеллект. Исследователи изучают процессы мышления, разумное поведение для того, чтобы найти методы решения подобных задач, так как человек в своей деятельности сталкивается с ними достаточно часто и успешно решает.

Хотя до сих пор многое задачи не решены, определенные достижения в этой области есть. Исследователи использовали различные подходы и методы, чтобы получить результат. В конце 50-х годов родилась модель лабиринтного поиска и появилась теория распознавания образов как следствие начала использования ЭВМ для решения невычислительных задач. Начало 60-х годов называют эпохой эвристического программирования, когда использовались стратегии действий на основе известных, заранее заданных эвристик. Эвристики позволяют сокращать количество рассматриваемых вариантов. В середине 60-х годов к решению задач стали активно подключать методы математической логики. С середины 70-х годов исследователи стали уделять внимание системам, основным на экспертных знаниях.

Такие системы применимы к слабоформализуемым задачам. Неформализованные задачи обычно обладают следующими особенностями:

– ошибочностью, неоднозначностью, неполнотой и противоречивостью исходных данных;

– ошибочностью, неоднозначностью, неполнотой и противоречивостью знаний о проблемной области и решаемой задаче;

– большой размерностью пространства решения (то есть перебор при поиске решения весьма велик);

– динамически изменяющимися данными и знаниями.

§2. Направления и подходы к исследованиям в области искусственного интеллекта

Основные подходы к исследованию искусственного интеллекта

Вскоре после признания искусственного интеллекта отдельной областью науки произошло разделение его на два направления: **нейрокибернетика** и **кибернетика черного ящика**. Эти направления развиваются практически независимо друг от друга, существенно различаясь как в методологии, так и в технологии.

Нейрокибернетики взяли за основу структуру и принципы функционирования единственного созданного природой устройства, способного рассуждать, – мозга. Клетки мозга называются нейронами, отсюда и название направления. Ученые считают, что, смоделировав мозг, смогут воссоздать и его работу.

Исследователи направления «кибернетика черного ящика» придерживались мнения, что не важно, по каким принципам работает устройство, какие средства и методы лежат в его основе, главное – имитировать функции мозга, даже если кроме результата это не будет иметь ничего общего с естественным разумом.

В настоящее время стали заметны тенденции к объединению этих направлений вновь в единое целое. Стало появляться множество гибридных методов и систем, например экспертная система на базе нейронной сети или нейронная сеть, обучаемая генетическим алгоритмом.

Исследователи, моделирующие только отдельные функции интеллекта, например распознавание образов, синтез речи, принятие решений, работают в рамках направления «**слабый** искусственный интеллект». Попытки воссоздать работу интеллекта в полном объеме относятся к направлению «**сильный** искусственный интеллект». Все основные достижения в области искусственного интеллекта относятся к слабому искусственному интеллекту.

Кроме этого выделяют **нисходящий (семиотический)** и **восходящий(биологический)** подходы.

Нисходящий подход предусматривает моделирование высокоуровневых психических процессов, таких как мышление, речь, эмоции и т.д.

Восходящий подход исследует интеллектуальное поведение систем на базе более мелких «неинтеллектуальных» элементов. Нейронные сети и эволюционное моделирование относятся к этому подходу.

Интеллектуальные системы разрабатываются с привлечением различных средств и методов. Существует четыре основных подхода к их построению: **логический, структурный, эволюционный** и **имитационный**.

Основой для логического подхода служит булева алгебра. Такая интеллектуальная система представляет собой машину доказательства теорем. При этом исходные данные хранятся в базе данных в виде аксиом, правила логического вывода – как отношения между ними. Кроме того, каждая такая машина имеет блок генерации цели, и система вывода пытается доказать данную цель как теорему. Если цель доказана, то трассировка примененных правил позволяет получить цепочку действий, необходимых для реализации поставленной цели. Мощность такой системы определяется возможностями генератора целей и машиной доказательства теорем. Для большинства логических методов характерна большая трудоемкость, поскольку во время поиска доказательства возможен полный перебор вариантов. Поэтому данный подход требует эффективной реализации вычислительного процесса, и хорошая работа обычно гарантируется при сравнительно небольшом размере базы данных.

Под структурным подходом подразумеваются попытки построения интеллектуальной системы путем моделирования структуры человеческого мозга, то есть рассматриваются системы, построенные в рамках направления «нейрокибернетика».

При построении интеллектуальной системы с помощью эволюционного подхода основное внимание уделяется построению начальной модели и правилам, по которым она может изменяться (эволюционировать). Причем модель может быть составлена по самым различным методам: это может быть и нейронная сеть, и набор логических правил, и любая другая модель. На основании проверки моделей отбираются самые лучшие из них, и на их базе по самым различным правилам генерируются новые модели, из которых опять выбираются самые лучшие и т.д.

Имитационный подход используется в рамках направления «кибернетика черного ящика». Интеллектуальные системы при таком подходе должны моделировать некую интеллектуальную функцию, то есть устанавливать необходимое соответствие между входами и выходами системы.

Основные направления исследований в области искусственного интеллекта

Тематика искусственного интеллекта охватывает огромный перечень научных направлений, начиная с таких задач общего характера, как обучение и восприятие, и заканчивая такими специальными задачами, как игра в шахматы, доказательство математических теорем, сочинение поэтических произведений и диагностика заболеваний. В искусственном интеллекте систематизируются и автоматизируются интеллектуальные задачи, и поэтому эта область касается любой сферы интеллектуальной деятельности человека.

Среди множества направлений искусственного интеллекта есть несколько ведущих, которые в настоящее время вызывают наибольший интерес у исследователей и практиков.

Представление знаний и разработка систем, основанных на знаниях

Это основное направление в области разработки систем искусственного интеллекта. Оно связано с разработкой моделей представления знаний, созданием баз знаний, образующих ядро экспертных систем.

Программное обеспечение систем искусственного интеллекта

В рамках этого направления разрабатываются специальные языки для решения интеллектуальных задач, в которых упор делается на преобладание логической и символьной обработки над вычислительными процедурами. Языки ориентированы на символьную обработку информации: LISP, PROLOG, РЕФАЛ и др. Помимо этого создаются пакеты прикладных программ, ориентированные на промышленную разработку интеллектуальных систем, или программные инструментарии искусственного интеллекта.

Разработка естественно-языковых интерфейсов и машинный перевод

Начиная с 50-х годов одной из популярных тем исследований в области искусственного интеллекта является компьютерная лингвистика, и в частности машинный перевод. Идея машинного перевода оказалась совсем не так проста, как казалось первым исследователям и разработчикам.

Интеллектуальные роботы

Роботы – это электротехнические устройства, предназначенные для автоматизации человеческого труда. Выделяют несколько поколений роботов.

I поколение. Роботы с жесткой схемой управления. Практически все современные промышленные роботы принадлежат к первому поколению. Фактически это программируемые манипуляторы.

II поколение. Адаптивные роботы с сенсорными устройствами. Есть образцы таких роботов, но в промышленности они пока используются мало.

III поколение. Самоорганизующиеся или интеллектуальные роботы. Это – конечная цель развития робототехники. Основные нерешенные проблемы при создании интеллектуальных роботов – проблема машинного зрения и проблема адекватного хранения и обработки трехмерной визуальной информации.

В настоящее время в мире изготавливается более 60 000 роботов в год. Фактически робототехника сегодня – это инженерная наука, не отвергающая технологий искусственного интеллекта, но не готовая пока к их внедрению в силу различных причин.

Обучение и самообучение

Активно развивающаяся область искусственного интеллекта. Включает модели, методы и алгоритмы, ориентированные на автоматическое накопление и формирование знаний на основе анализа и обобщения данных, обучение по примерам (или индуктивное), а также традиционные подходы из теории распознавания образов.

В последние годы к этому направлению тесно примыкают стремительно развивающиеся системы анализа данных и поиска закономерностей в базах данных.

Распознавание образов

Направление искусственного интеллекта, берущее начало у самых его истоков, но в настоящее время выделившееся в самостоятельную науку. Ее основной подход – описание классов объектов через определенные значения значимых признаков. Каждому объекту ставится в соответствие матрица признаков, по которой происходит его распознавание. Процедура распознавания использует чаще всего специальные математические процедуры и функции, разделяющие объекты на классы. Это направление близко к машинному обучению и тесно связано с нейрокибернетикой.

Новые архитектуры компьютеров

Самые современные процессоры сегодня основаны на традиционной последовательной архитектуре фон Неймана, используемой еще в компьютерах первых поколений. Эта архитектура крайне неэффективна для символьной обработки. Поэтому усилия многих научных коллективов и фирм уже десятки лет нацелены на разработку аппаратных архитектур, предназначенных для обработки символьных и логических данных. Создаются Пролог- и Лисп-машины, компьютеры V и VI поколений. Последние разработки посвящены компьютерам баз данных, параллельным и векторным компьютерам. И хотя удачные промышленные решения существуют, высокая стоимость, недостаточное программное оснащение и аппаратная несовместимость с традиционными компьютерами существенно тормозят широкое использование новых архитектур.

Игры

Это ставшее скорее историческим направление связано с тем, что на заре исследований искусственного интеллекта традиционно включало в себя игровые интеллектуальные задачи – шахматы, шашки, го. В основе первых программ лежал один из ранних подходов – лабиринтная модель мышления плюс эвристики.

Сейчас это скорее коммерческое направление, так как в научном плане эти идеи считаются тупиковыми. В настоящее время в компьютерных играх (например, UnrealTournament, ReturntoCastleWolfeStein, Black&White, Doom, Sim) стали применяться другие идеи искусственного интеллекта – нейронные сети, интеллектуальные агенты, генетические алгоритмы и т.д., которые

позволяют создавать персонажей (ботов) с различной степенью «интеллекта». Использование методов искусственного интеллекта в играх позволяет получать новые эффективные решения, создавать шаблоны проектирования, повышать развлекательность и достоверность игр.

Машинное творчество

Направление охватывает сочинение компьютером музыки (Айзексон, Хиллер, Зармпов), стихов (Д. Линк), живописи (Х. Фарид, Л. Моура) и даже сказок и афоризмов. Основным методом подобного «творчества» является метод пермутаций (перестановок) плюс использование некоторых баз знаний и данных, содержащих результаты исследований по структурам текстов, рифм, сценариям и т.п.

Нечеткие модели и мягкие вычисления.

Это направление представлено нечеткими схемами «вывода по аналогии», взглядом на теорию нечетких мер с вероятностных позиций, нечетким представлением аналитическими моделями для описания геометрических объектов, алгоритмами эволюционного моделирования с динамическими параметрами, такими как время жизни и размер популяции, методами решения оптимизационных задач с использованием технологий генетического поиска, гомеостатических и синергетических принципов и элементов самоорганизации.

Эвристическое программирование

В рамках направления исследуют последовательности мыслительных операций, выполнение которых приводит к успешному решению той или иной задачи, моделируют мыслительную деятельность человека для решения задач, не имеющих строгого формализованного алгоритма или связанных с неполнотой исходных данных.

Искусственная жизнь

Направление исследований, целью которого является создание искусственных существ, способных действовать не менее эффективно, чем живые существа. Мягкая искусственная жизнь создает вычислительные системы и модели, действующие на базе биологических и эволюционных принципов. Влажная искусственная жизнь синтезирует новые искусственные биологические формы. В рамках этого направления

используют генетические алгоритмы, клеточные автоматы, автономные агенты и т.д.

Когнитивное моделирование

Научное направление, являющееся плодотворным синтезом когнитивной графики и вычислительного моделирования, позволяющее существенно повысить познавательную эффективность современных ЭВМ. Методология когнитивного моделирования предназначена для анализа и принятия решений в плохо определенных ситуациях, основывается на моделировании субъективных представлений эксперта.

Эволюционное моделирование

При эволюционном моделировании процесс моделирования сложной социально-экономической системы сводится к созданию модели его эволюции или к поиску допустимых состояний системы, к процедуре (алгоритму) отслеживания множества допустимых состояний (траекторий).

Многоагентные системы

Направление искусственного интеллекта, которое рассматривает решение одной задачи несколькими интеллектуальными подсистемами – агентами. Агент – аппаратная или программная сущность, способная действовать в интересах достижения цели, поставленной перед всей системой.

Социальные системы дают еще одно модельное представление интеллекта с помощью глобального поведения, позволяющего им решать проблемы, которые бы не удалось решить отдельным их членам. Агенты в таких системах автономны или полуавтономны, у каждого агента есть определенный круг подзадач, причем он располагает малым знанием (или вовсе не располагает знанием) о том, что делают другие агенты или как они это делают. Каждый агент выполняет свою независимую часть решения проблемы и либо выдает собственно результат (что-то совершает), либо сообщает результат другим агентам.

Онтологии

В рамках этого направления исследуется возможность всеобъемлющей и детальной формализации некоторой области знаний с помощью концептуальной схемы – иерархической структуры данных, содержащей все релевантные классы объектов, их связи и правила

предметной области. Онтологии используются и людьми, и программными агентами, позволяют повторно использовать знания предметной области, отделять их от оперативных знаний и анализировать их. Разрабатываются языки описания онтологий (RDF, DAML, OWL, KIF).

Компьютерные вирусы

Последнее поколение компьютерных вирусов обладает всеми атрибутами систем искусственного интеллекта. Эти вирусы способны к размножению, мутации, эволюции, обучению. Современные проблемы по защите от них окажутся незначительными, когда они полностью проникнут в сферу искусственного интеллекта. Методы искусственного интеллекта необходимы как для их создания, так и для разработки эффективных средств защиты.

Интеллектуальное математическое моделирование

В данном направлении системы имитируют творческую деятельность математика-профессионала, занимающегося решением, например, краевых задач математической физики. Для этого используются базы знаний, содержащие теоремы, математические зависимости, эвристические правила. Такие системы способны к обучению и самообучению.

Это далеко не все направления искусственного интеллекта, существует множество направлений для решения множества задач.

§3. Классификация интеллектуальных информационных систем

Определение интеллектуальной информационной системы

Существует большое множество интеллектуальных информационных систем (ИИС). Однако общепринятого единого определения интеллектуальной информационной системы нет.

Интеллектуальной информационной системой называют автоматизированную информационную систему, основанную на знаниях, или комплекс программных, лингвистических и логико-математических средств для реализации основной задачи – осуществления поддержки деятельности человека и поиска информации в режиме продвинутого диалога на естественном языке.

Кроме того, информационно-вычислительными системами с интеллектуальной поддержкой для решения сложных задач называют те системы, в которых логическая обработка информации превалирует над вычислительной.

Таким образом, любая информационная система, решающая интеллектуальную задачу или использующая методы искусственного интеллекта, относится к интеллектуальным.

Для интеллектуальных информационных систем характерны следующие признаки:

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

Коммуникативные способности ИИС характеризуют способ взаимодействия (интерфейса) конечного пользователя с системой, в частности возможность формулирования произвольного запроса в диалоге с ИИС на языке, максимально приближенном к естественному.

Сложные плохо формализуемые задачи – это задачи, которые требуют построения оригинального алгоритма решения в зависимости от конкретной ситуации, для которой могут быть характерны неопределенность и динамичность исходных данных и знаний.

Способность к самообучению – это возможность автоматического извлечения знаний для решения задач из накопленного опыта конкретных ситуаций.

Адаптивность – способность к развитию системы в соответствии с объективными изменениями модели проблемной области.

Классификация интеллектуальных систем

В соответствии с перечисленными признаками ИИС делятся на(данная классификация – одна из возможных)(рис. 1):

- системы с коммутативными способностями (с интеллектуальным интерфейсом);
- экспертные системы (системы для решения сложных задач);
- самообучающиеся системы (системы, способные к самообучению);
- адаптивные системы (адаптивные информационные системы).

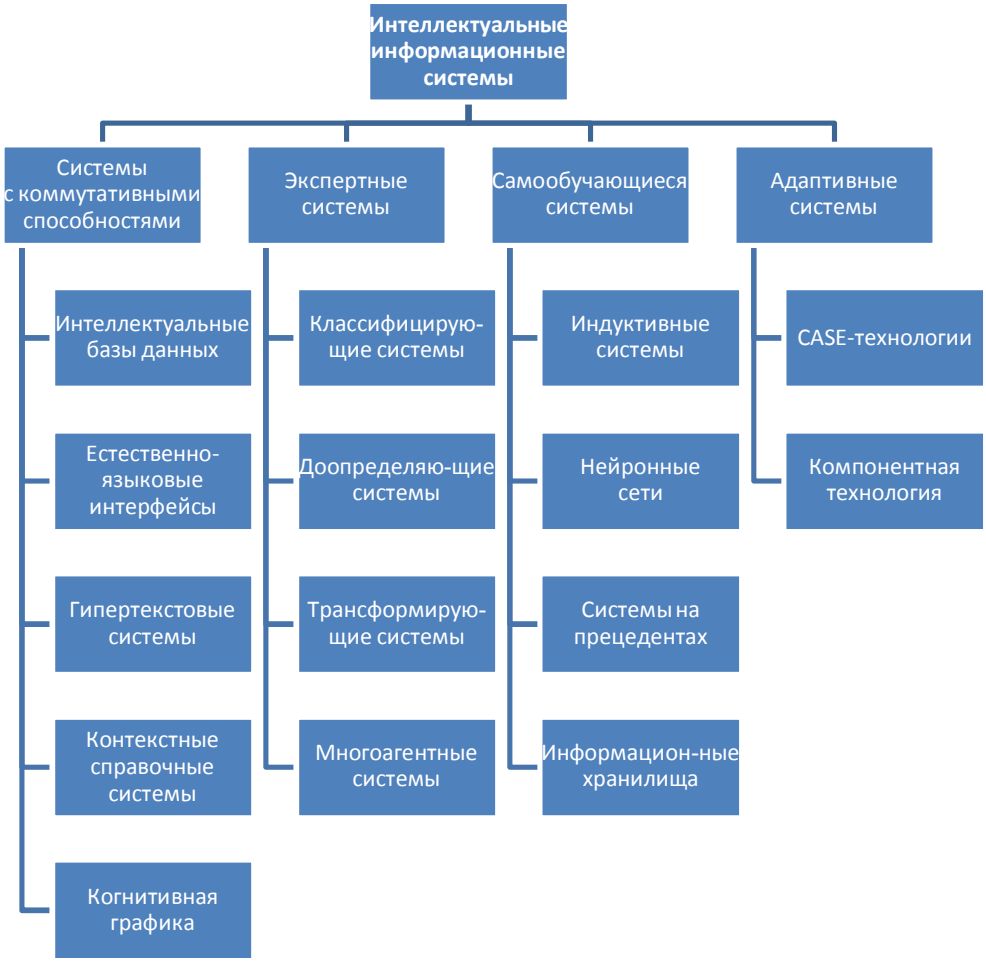

Рис. 1. **Классификация интеллектуальных информационных систем по типам систем**

Интеллектуальные базы данных отличаются от обычных баз данных возможностью выборки по запросу необходимой информации, которая может явно не храниться, а выводиться из имеющейся в базе данных.

Естественно-языковой интерфейс предполагает трансляцию естественно-языковых конструкций на внутримашинный уровень представления знаний. Для этого необходимо решать задачи морфологического, синтаксического и семантического анализа и синтеза высказываний на естественном языке. Так, морфологический анализ предполагает распознавание и проверку правильности написания слов по словарям, синтаксический контроль – разложение входных сообщений на отдельные компоненты (определение структуры) с проверкой соответствия грамматическим правилам внутреннего представления знаний и выявления недостающих частей и, наконец, семантический анализ – установление смысловой правильности синтаксических конструкций. Синтез высказываний решает обратную задачу преобразования внутреннего представления информации в естественно-языковое.

Естественно-языковой интерфейс используется для:

- доступа к интеллектуальным базам данных;
- контекстного поиска документальной текстовой информации;
- голосового ввода команд в системах управления;
- машинного перевода с иностранных языков.

Гипертекстовые системы предназначены для реализации поиска по ключевым словам в базах текстовой информации. Интеллектуальные гипертекстовые системы отличаются возможностью более сложной семантической организации ключевых слов, которая отражает различные смысловые отношения терминов. Таким образом, механизм поиска работает прежде всего с базой знаний ключевых слов, а уже затем непосредственно с текстом. В более широком плане сказанное распространяется и на поиск мультимедийной информации, включающей,помимо текстовой, и цифровую информацию.

Системы контекстной помощи можно рассматривать как частный случай интеллектуальных гипертекстовых и естественно-языковых систем. В отличие от обычных систем помощи, навязывающих пользователю схему поиска требуемой информации, в системах контекстной помощи пользователь описывает проблему (ситуацию), а система с помощью

дополнительного диалога ее конкретизирует и сама выполняет поиск относящихся к ситуации рекомендаций. Такие системы относятся к классу систем распространения знаний (KnowledgePublishing) и создаются как приложение к системам документации (например, технической документации по эксплуатации товаров).

Системы когнитивной графики позволяют осуществлять интерфейс пользователя с ИИС с помощью графических образов, которые генерируются в соответствии с происходящими событиями. Такие системы используются в мониторинге и управлении оперативными процессами. Графические образы в наглядном и интегрированном виде описывают множество параметров изучаемой ситуации. Например, состояние сложного управляемого объекта отображается в виде человеческого лица, на котором каждая черта отвечает за какой-либо параметр, а общее выражение лица дает интегрированную характеристику ситуации. Системы когнитивной графики широко используются также в обучающих и тренажерных системах на основе использования принципов виртуальной реальности, когда графические образы моделируют ситуации, в которых обучаемому необходимо принимать решения и выполнять определенные действия.

*Экспертные системы*предназначены для решения задач на основе накапливаемой базы знаний, отражающей опыт работы экспертов в рассматриваемой проблемной области.

*Многоагентные системы.*Для таких динамических систем характерна интеграция в базе знаний нескольких разнородных источников знаний, обменивающихся между собой получаемыми результатами на динамической основе.

Для *многоагентных систем* характерны следующие особенности:

1) проведение альтернативных рассуждений на основе использования различных источников знаний с механизмом устранения противоречий;

2) распределенное решение проблем, которые разбиваются на параллельно решаемые подпроблемы, соответствующие самостоятельным источникам знаний;

3) применение множества стратегий работы механизма вывода заключений в зависимости от типа решаемой проблемы;

4) обработка больших массивов данных, содержащихся в базе данных;

5) использование различных математических моделей и внешних процедур, хранимых в базе моделей;

6) способность прерывания решения задач в связи с необходимостью получения дополнительных данных и знаний от пользователей, моделей, параллельно решаемых подпроблем.

В основе *самообучающихся систем* лежат методы автоматической классификации примеров ситуаций реальной практики.

Характерными признаками самообучающихся систем являются:

- самообучающиеся системы «с учителем», когда для каждого примера задается в явном виде значение признака его принадлежности некоторому классу ситуаций (классообразующего признака);
- самообучающиеся системы «без учителя», когда по степени близости значений признаков классификации система сама выделяет классы ситуаций.

Индуктивные системы используют обобщение примеров по принципу от частного к общему. Процесс классификации примеров осуществляется следующим образом:

1. Выбирается признак классификации из множества заданных (либо последовательно, либо по какому-либо правилу, например в соответствии с максимальным числом получаемых подмножеств примеров).

2. По значению выбранного признака множество примеров разбивается на подмножества.

3. Выполняется проверка, принадлежит ли каждое образовавшееся подмножество примеров одному подклассу.

4. Если какое-то подмножество примеров принадлежит одному подклассу, то есть у всех примеров подмножества совпадает значение классообразующего признака, то процесс классификации заканчивается (при этом остальные признаки классификации не рассматриваются).

5. Для подмножеств примеров с несовпадающим значениемклассообразующего признака процесс классификации продолжается начиная с пункта 1 (каждое подмножество примеров становится классифицируемым множеством).

Нейронные сети представляют собой устройства параллельных вычислений, состоящие из множества взаимодействующих простых

процессоров. Каждый процессор такой сети имеет дело только с сигналами, которые он периодически получает, и сигналами, которые он периодически посылает другим процессорам.

В экспертных системах, *основанных на прецедентах* (аналогиях), база знаний содержит описания не обобщенных ситуаций, а собственно сами ситуации или прецеденты.

Поиск решения проблемы в экспертных системах, основанных на прецедентах, сводится к поиску по аналогии (то естьабдуктивный вывод от частного к частному).

В отличие от интеллектуальной базы данных,*информационное хранилище* представляет собой хранилище извлеченной значимой информации из оперативной базы данных, которое предназначено для оперативного ситуационного анализа данных (реализации OLAP-технологии).

Типичными задачами оперативного ситуационного анализа являются:

- определение профиля потребителей конкретных объектов хранения;
- предсказание изменений объектов хранения во времени;
- анализ зависимостей признаков ситуаций (корреляционный анализ).

Адаптивная информационная система– это информационная система, которая изменяет свою структуру в соответствии с изменением модели проблемной области.

При этом:

1) адаптивная информационная система должна в каждый момент времени адекватно поддерживать организацию бизнес-процессов;

2) адаптивная информационная система должна проводить адаптацию всякий раз, как возникает потребность в реорганизации бизнес-процессов;

3) реконструкция информационной системы должна проводиться быстро и с минимальными затратами.

Ядром адаптивной информационной системы является постоянно развиваемая модель проблемной области (предприятия), поддерживаемая в специальной базе знаний –репозитории. На основе ядра осуществляется генерация или конфигурация программного обеспечения. Таким образом, проектирование и адаптация ИС сводится, прежде всего, к построению модели проблемной области и ее своевременной корректировке.

Так как нет общепринятого определения, четкую единую классификацию интеллектуальных информационных систем дать затруднительно.

Если рассматривать интеллектуальные информационные системы с точки зрения решаемой задачи, то можно выделить системы управления и справочные системы, системы компьютерной лингвистики, системы распознавания, игровые системы и системы создания интеллектуальных информационных систем (рис.2).

При этом системы могут решать не одну, а несколько задач или в процессе решения одной задачи решать и ряд других. Например, при обучении иностранному языку система может решать задачи распознавания речи обучаемого, тестировать, отвечать на вопросы, переводить тексты с одного языка на другой и поддерживать естественно-языковой интерфейс работы.

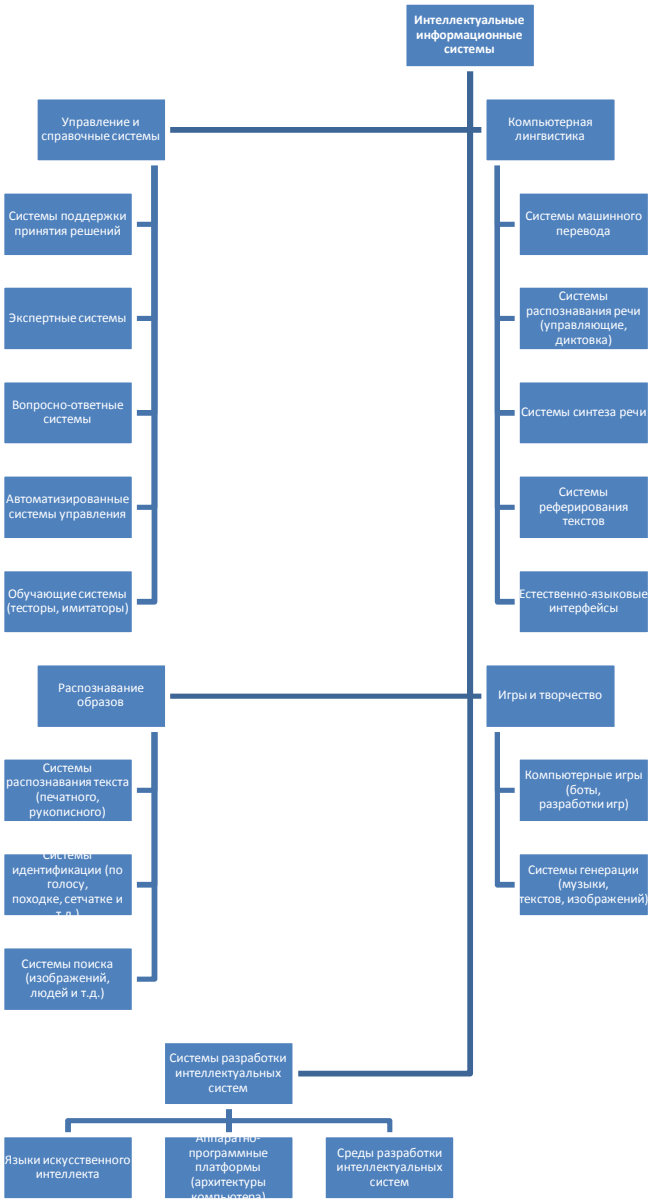

Рис. 2. **Классификация интеллектуальных информационных систем по решаемым задачам**

Если классифицировать интеллектуальные информационные системы по критерию «используемые методы», то они делятся на жесткие, мягкие и гибридные (рис.3).

Мягкие вычисления – это сложная компьютерная методология, основанная на нечеткой логике, генетических вычислениях, нейрокомпьютинге и вероятностных вычислениях. *Жесткие* вычисления – традиционные компьютерные вычисления (не мягкие). *Гибридные системы* – системы, использующие более чем одну компьютерную

технологию (в случае интеллектуальных систем – технологии искусственного интеллекта).

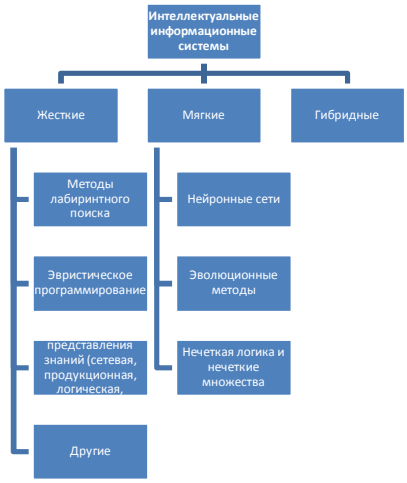

Рис. 3. **Классификация интеллектуальных информационных систем по методам**

Возможны и другие классификации, например выделяют системы общего назначения и специализированные системы (рис. 4).

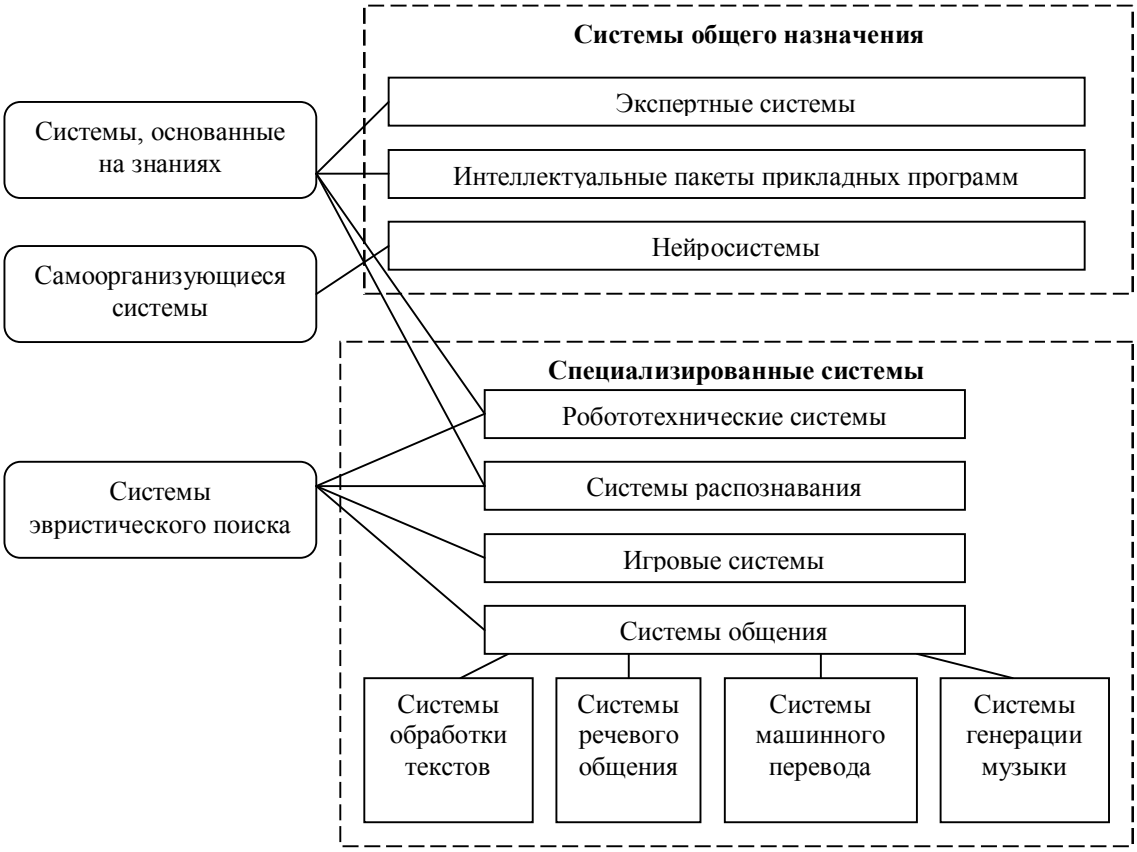

Рис. 4. **Классификация интеллектуальных систем по назначению**

Кроме того, эта схема отражает еще один вариант классификации по методам: системы, использующие методы представления знаний, самоорганизующиеся системы и системы, созданные с помощью эвристического программирования. Также в этой классификации системы генерации музыки отнесены к системам общения.

К интеллектуальным системам *общего назначения* относятся системы, которые не только исполняют заданные процедуры, но на основе метапроцедур поиска генерируют и исполняют процедуры решения новых конкретных задач.

Специализированные интеллектуальные системы выполняют решение фиксированного набора задач, предопределенного при проектировании системы.

Отсутствие четкой классификации также объясняется многообразием интеллектуальных задач и интеллектуальных методов, кроме того, искусственный интеллект – активно развивающаяся наука, в которой новые прикладные области осваиваются ежедневно.

Часть 2. ОСНОВЫ ТЕОРИИ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

§1. Представление знаний

Данные и знания

Данные – это информация, полученная в результате наблюдений или измерений отдельных свойств (атрибутов), характеризующих объекты, процессы и явления предметной области.

Знание– форма существования и систематизации результатов познавательной деятельности человека. Знание помогает людям рационально организовывать свою деятельность и решать различные проблемы, возникающие в ее процессе; субъективный образ объективной реальности, то есть адекватное отражение внешнего и внутреннего мира в сознании человека в форме представлений, понятий, суждений, теорий.

Знание *(в широком смысле)* – совокупность понятий, теоретических построений и представлений.

Знание *(в узком смысле)*– признак определенного объема информации, определяющий ее статус и отделяющий от всей прочей информации по критерию способности к решению поставленной задачи.

Знание *(с точки зрения представления знаний в интеллектуальных системах)*– это связи и закономерности предметной области (принципы, модели, законы), полученные в результате практической деятельности и профессионального опыта, позволяющего специалистам ставить и решать задачи в данной области.

Знания от данных отличаются рядом **свойств:**

- внутренняя интерпретируемость;
- структурированность;
- связность;
- семантическая метрика;
- активность.

Внутренняя интерпретируемость. Данные, хранящиеся в памяти или на внешних носителях, лишены имен, таким образом, отсутствует возможность их однозначной идентификации системой. Данные может идентифицировать лишь программа, извлекающая их по определенному алгоритму. При переходе к знаниям в память вводится дополнительная информация (атрибуты: фамилия, год рождения, специальность, стаж). Атрибуты могут играть роль имен. По ним можно осуществлять поиск нужной информации.

Структурированность. Информационные единицы должны обладать гибкой структурой. Иначе говоря, должна существовать возможность произвольного установления между отдельными информационными единицами отношений типа «часть–целое», «род–вид» или «элемент–класс».

Связность. Между информационными единицами должна быть предусмотрена возможность установления связей различного типа. Семантика отношений может носить декларативный или процедурный характер. Например, две и более информационные единицы могут быть связаны отношением «одновременно», две информационные единицы – отношением «причина–следствие» или «быть рядом».

Семантическая метрика. На множестве информационных единиц в некоторых случаях полезно задавать отношение, характеризующее их ситуационную близость, то есть силу ассоциативной связи. Его можно было бы назвать отношением релевантности для информационных единиц. Оно дает возможность выделять в информационной базе некоторые типовые ситуации (например, «покупка», «регулирование движения на перекрестке»). Отношение релевантности при работе с информационными единицами позволяет находить знания, близкие к уже найденным.

Активность. Все вычислительные процессы инициируются командами, а данные используются этими командами лишь в случае необходимости. Иначе говоря, данные пассивны, а команды активны.

Знания позволяют адаптироваться и действовать в реальной действительности. Существует огромное множество различных знаний, начиная от рецепта приготовления омлета до квантовой физики. Знания можно классифицировать по нескольким критериям (рис. 5).

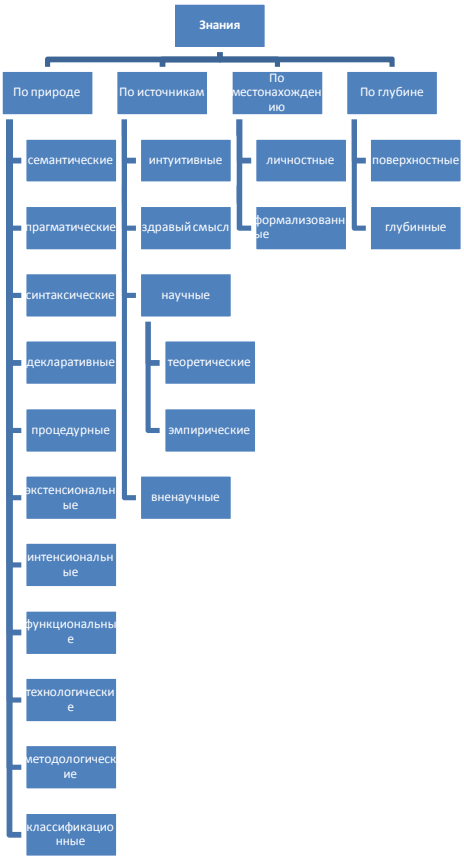

Рис. 5. **Классификация знаний**

Знание *синтаксического* типа характеризует синтаксическую структуру потока информации, которая не зависит от смысла и содержания используемых при этом понятий, то есть интеллектуальную систему не образует.

Семантическое знание рассматривается как структура, образующая текущий контекст. Оно содержит информацию, непосредственно связанную с текущими значениями и смыслом описываемых понятий, и предопределяет состояние связей данных в информационной базе.

Прагматическое знание предопределяет наиболее вероятные связи, описывающие данные с точки зрения решаемой задачи (обобщенный или «объективный» контекст), например с учетом действующих в данной задаче специфических критериев и соглашений.

Декларативные знания содержат в себе представление о структуре понятий. Эти знания приближены к данным, фактам. Например, высшее учебное заведение есть совокупность факультетов, а каждый факультет в свою очередь есть совокупность кафедр.

Процедурные знания имеют активную природу. Они определяют представления о средствах и путях получения новых знаний, проверке знаний. Это алгоритмы разного рода. С развитием информатики все большая часть знаний сосредотачивалась в структурах данных (таблицы, списки, абстрактные типы данных), то есть увеличивалась роль декларативных.

Существенными для понимания природы знаний являются способы определения понятий. Один из широко применяемых способов основан на идее интенсионала и экстенсионала.

Интенсионал понятия – это определение его через соотнесение с понятием более высокого уровня абстракции с указанием специфических свойств.

Экстенсионал понятия – это определение понятия через перечисление его конкретных примеров, то есть понятий более низкого уровня абстракции. Интенсионалы формируют знания об объектах, в то время как экстенсионалы объединяют данные.

Отсюда *интенсиональные* знания – это знания о предметной области, которые отражают факты, закономерности, свойства и характеристики, справедливые для любых ситуаций, которые могут возникнуть в этой предметной области.

Экстенсиональные знания – это знания о предметной области, отражающие факты, закономерности, свойства и характеристики, типичные для конкретных ситуаций или классов однотипных ситуаций, которые могут возникнуть в этой области.

Функциональные знания – это знания о выполняемых функциях отдельных предметов и о применении их в реальной действительности.

Технологические знания – специализированные знания, обеспечивающие поддержание технологических параметров производства; производственный опыт и навыки, используемые при решении повседневных производственных вопросов. Это может быть знание последовательности операций или знание технологической цепочки, позволяющие достигать поставленные цели в соответствии с принятой технологией.

Методологические знания – знания о методах преобразования действительности, научные знания о построении эффективной

деятельности. К методологическим знаниям относят знание целей, форм и направлений развития теории, методов и способов эффективного преобразования практики.

Классификационные знания применяются главным образом в науке, являются обобщенными, системными знаниями. Пример – система элементов Д.И. Менделеева.

Интуиция– это вид знания, специфика которого обусловлена способом его приобретения. Это знание, не нуждающееся в доказательстве и воспринимаемое как достоверное. По способу получения интуиция – это прямое усмотрение объективной связи вещей, не опирающееся на доказательство (интуицияесть усмотрение внутренним зрением; от лат. intueri– созерцать).

Под *здравым смыслом* понимают знания, позволяющие принимать правильные решения и делать правильные предположения, основываясь на логическом мышлении и накопленном опыте. В этом значении термин зачастую акцентирует внимание на способности человеческого разума противостоять предрассудкам, заблуждениям, мистификациям.

Научные знания в любом случае должны быть основанными на эмпирической или теоретической доказательной основе.

Теоретические знания– абстракции, аналогии, схемы, отображающие структуру и природу процессов, протекающих в предметной области. Эти знания объясняют явления и могут использоваться для прогнозирования поведения объектов. Теоретический уровень научного знания предполагает установление законов, дающих возможность идеализированного восприятия, описания и объяснения эмпирических ситуаций, то есть познания сущности явлений. Теоретические законы имеют более строгий, формальный характер по сравнению с эмпирическими. Термины описания теоретического знания относятся к идеализированным, абстрактным объектам. Подобные объекты невозможно подвергнуть непосредственной экспериментальной проверке.

Эмпирические знания получают в результате применения эмпирических методов познания: наблюдения, измерения, эксперимента. Это знания о видимых взаимосвязях между отдельными событиями и фактами в предметной области. Эмпирические знания, как правило, констатируют качественные и количественные характеристики объектов и

явлений. Эмпирические законы часто носят вероятностный характер и не являются строгими.

Вненаучные знания могут быть различными. *Паранормальные*знания – знания, несовместимые с имеющимся гносеологическим стандартом. Широкий класс *паранаучного* (пара от греч.около, при) знания включает в себя учения или размышления о феноменах, объяснение которых не является убедительным с точки зрения критериев научности. *Лженаучные* знания– сознательно эксплуатирующие домыслы и предрассудки. В качестве симптомов лженауки выделяют малограмотный пафос, принципиальную нетерпимость к опровергающим доводам, а также претенциозность. Лженаучные знания сосуществуют с научными знаниями.

Личностные (неявные, скрытые) знания – это знания людей, полученные из практики и опыта.

Формализованные (явные) знания – знания, содержащиеся в документах, на компакт-дисках, в персональных компьютерах, в Интернете, в базах знаний, в экспертных системах. Формализованные знания объективизируются знаковыми средствами языка, охватывают те знания, о которых мы знаем, их можно записать, сообщить другим.

Классификация моделей представления знаний

Для хранения данных используются базы данных (для них характерны большой объем и относительно небольшая удельная стоимость информации), для хранения знаний – базы знаний (небольшого объема, но исключительно дорогие информационные массивы).

База знаний – основа любой интеллектуальной системы, где знания описаны на некотором языке представления знаний, приближенном к естественному. Сегодня знания приобрели чисто декларативную форму, то есть знаниями считаются предложения, записанные на языках представления знаний, приближенных к естественному языку и понятных неспециалистам.

Совокупность знаний, нужных для принятия решений, принято называть предметной областью или **знаниями о предметной области**. В любой предметной области есть свои понятия и связи между ними, своя

терминология, свои законы, связывающие между собой объекты данных предметной области, свои процессы и события. Кроме того, каждая предметная область имеет свои методы решения задач. Решая задачи такого вида на ЭВМ, используют информационные системы, ядром которых является база знаний, содержащая основные характеристики предметных областей.

Базы знаний базируются на **моделях представления знаний**, подобно базам данных, которые основаны на моделях представления данных (иерархической, сетевой, реляционной, постреляционной и т.д.).

При представлении знаний в памяти интеллектуальной системы традиционные языки, основанные на численном представлении данных, являются неэффективными. Для этого используются специальные языки представления знаний, основанные на символьном представлении данных. Они делятся на типы по формальным моделям представления знаний. Наиболее часто используется на практике классификация моделей представления знаний, приведенная на рис. 6, где модели представления знании делятся на детерминированные (жесткие) и мягкие.

Рис. 6. **Модели представления знаний**

Детерминированные модели включают в себя фреймы, логико-алгебраические модели, семантические сети и продукционные модели. Мягкие модели включают в себя нечеткие системы, нейронные сети, эволюционные модели, гибридные системы.

С моделированием знаний непосредственно связана проблема выбора языка представления. В целях классификации моделей представления знаний выделяется девять ключевых требований к моделям знаний:

1) общность (универсальность);
2) наглядность представления знаний;
3) однородность;
4) реализация в модели свойства активности знаний;
5) открытость;
6) возможность отражения структурных отношений объектов предметной области;
7) наличие механизма «проецирования» знаний на систему семантических шкал;
8) возможность оперирования нечеткими знаниями;
9) использование многоуровневых представлений (данные, модели, метамодели, метаметамодели и т.д.).

Модели представления знаний не удовлетворяют полностью этим требованиям, чем и объясняется их многообразие и активное развитие данного направления.

Логико-алгебраические модели представления знаний

В логических моделях знания представляются в виде совокупности правильно построенных формул какой-либо формальной системы (ФС), которая задается четверкой

$$S=<T,P,A,R>,$$

где T– множество базовых (терминальных) элементов, из которых формируются все выражения; P– множество синтаксических правил, определяющих синтаксически правильные выражения из терминальных элементов ФС; A – множество аксиом ФС, соответствующих синтаксически правильным выражениям, которые в рамках данной ФС априорно считаются истинными; R –конечное множество отношений $\{r_1, r_2, ..., r_n\}$ между формулами, называемыми правилами вывода, позволяющих получать из одних синтаксически правильных выражений другие.

Для любого r_i существует целое положительное число j, такое что для каждого множества, состоящего из j формул, и для каждой формулы F эффективно решается вопрос о том, находятся ли эти j-формулы в отношении r_i с формулой F. Если r_i выполняется, то F называют непосредственным следствием F-формул по правилу r_i.

Следствием (выводом) формулы в теории S называется такая последовательность правил, что для любого из них представленная формула является либо аксиомой теории S, либо непосредственным следствием.

Простейшей логической моделью является исчисление высказываний, которое представляет собой один из начальных разделов математической логики, служащий основой для построения более сложных формализмов. В практическом плане исчисление высказываний применяется в ряде предметных областей (в частности, при проектировании цифровых электронных схем). Развитие логики высказываний нашло отражение в исчислении предикатов первого порядка.

Под *исчислением предикатов* понимается формальный язык для представления отношений в некоторой предметной области. Основное преимущество исчисления предикатов– хорошо понятный механизм математического вывода, который может быть непосредственно запрограммирован. Предикатом называют предложение, принимающее только два значения: «истина» или «ложь». Для обозначения предикатов применяются логические связки между высказываниями: $\neg$– не, $\vee$– или, $\wedge$– и, $\supset$– если, а также квантор существования $\exists$и квантор всеобщности $\forall$.

Таким образом, логика предикатов оперирует логическими связками между высказываниями, например она решает вопросы: можно ли на основе высказывания A получить высказывание B и т.д.

Допустимые выражения в исчислении предикатов называются правильно построенными формулами, состоящими из атомных формул. Атомные формулы состоят из предикатов и термов, разделяемых круглыми, квадратными и фигурными скобками.

Предикатные символы – это в основном глагольная форма (например: ПИСАТЬ, УЧИТЬ, ПЕРЕДАТЬ), но не только глагольная

форма, а форма прилагательных, наречий (например: КРАСНЫЙ, ЗНАЧЕНИЕ, ЖЕЛТЫЙ).

Предикатные символы и константы, как правило, обозначаются заглавными символами, функциональные символы и переменные – строчными.

В абстрактных примерах они обозначаются латинскими буквами f, g, h. В предложениях предикатной формы важны отношения и элементы. Определяя отношения, мы определяем значимость элементов выражения. Элементы могут быть предикатами и термами.

Если существует некоторая предметная область, то предикаты определяют отношения в этой предметной области, константы – элементы этой предметной области, функциональный символ – функцию.

Рассмотрим некоторые примеры. Высказывание «у каждого человека есть отец» можно записать:

$\forall x \exists y$ (ЧЕЛОВЕК(x) $\supset$ ОТЕЦ(y,x))

Выражение «Антон владеет красной машиной» записывается, например, так:

$\exists x$ (ВЛАДЕЕТ(АНТОН, x) МАШИНА(x) $\wedge$ КРАСНЫЙ(x))

Представление знаний в рамках логики предикатов служит основой направления ИИ, называемого логическим программированием. Методы логического программирования в настоящее время широко используются на практике при создании ИИС в ряде предметных областей. Положительными чертами логических моделей знаний в целом являются:

- высокий уровень формализации, обеспечивающий возможность реализации системы формально точных определений и выводов;
- согласованность знаний как единого целого, облегчающая решение проблем верификации БЗ, оценки независимости и полноты системы аксиом и т.д.;
- единые средства описания как знаний о предметной области, так и способов решения задач в этой предметной области, что позволяет любую задачу свести к поиску логического вывода некоторой формулы в той или иной ФС.

Однако такое единообразие влечет за собой основной недостаток модели – сложность использования в процессе логического вывода

эвристик, отражающих специфику предметной области. К другим недостаткам логической модели относят:

- «монотонность»;
- «комбинаторный взрыв»;
- слабость структурированности описаний.

Продукционные модели представления знаний

Продукционная модель или модель, основанная на правилах, позволяет представить знания в виде предложений типа «если (условие), то (действие)».

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

Продукционная модель в чистом виде не имеет механизма выхода из тупиковых состояний в процессе поиска. Она продолжает работать, пока не будут исчерпаны все допустимые продукции. Практические реализации продукционных систем содержат механизмы возврата в предыдущее состояние для управления алгоритмом поиска.

Рассмотрим пример использования продукционных правил.

П1: Если «отдых – летом» и «человек – активный», то «ехать в горы».

П2: Если «любит солнце», то «отдых летом».

Предположим, в систему поступили данные:«человек активный» и «любит солнце».

Прямой вывод: исходя из данных, получить ответ.

1-й проход

Шаг 1. Пробуем П1; не работает (не хватает данных «отдых – летом»).

Шаг 2. Пробуем П2; работает, в базу поступает факт «отдых – летом».

2-й проход

Шаг 3. Пробуем П1; работает, активируется цель «ехать в горы», которая и выступает как совет, который дает ЭС.

Обратный вывод: подтвердить выбранную цель при помощи имеющихся правил и данных.

1-й проход

Шаг 1. Цель –«ехать в горы»: пробуем П1– данных «отдых – летом» нет, они становятся новой целью, и ищется правило, где она в правой части.

Шаг 2. Цель «отдых – летом»: правило П2 подтверждает цель и активирует ее.

2-й проход

Шаг 3. Пробуем П1; подтверждается искомая цель.

Основные преимущества продукционных систем:

- простота и гибкость выделения знаний;
- отделение знаний от программы поиска;
- модульность продукционных правил (правила не могут «вызывать» другие правила);
- возможность эвристического управления поиском;
- возможность трассировки «цепочки рассуждений»;
- независимость от выбора языка программирования;
- продукционные правила являются правдоподобной моделью решения задачи человеком.

Имеется большое число программных средств, реализующих продукционный подход (например, языки высокого уровня CLIPS и OPS 5; «оболочки» или «пустые» ЭС – EXSYS Professional и Kappa, инструментальные –KEE, ARTS, PIES , а также промышленных ЭС на его основе).

Семантические сети

Семантическая сеть – это ориентированный граф, вершины которого – понятия, а дуги – отношения между ними. Термин *«семантическая»*означает«смысловая», а сама семантика – это наука, устанавливающая отношения между символами и объектами, которые они обозначают, то есть наука, определяющая смысл знаков. Модель на основе семантических сетей была предложена американским психологом Куиллианом.

В качестве понятий обычно выступают абстрактные или конкретные объекты, а *отношения* – это связи типа: «это» (АКО – A-Kind-Of, is или «элемент класса»), «имеет частью» (haspart), «принадлежит», «любит» и т.д.

Можно предложить несколько классификаций семантических сетей, связанных с типами отношений между понятиями.

По количеству типов отношений:

- однородные (с единственным типом отношений);
- неоднородные (с различными типами отношений).

По типам отношений:

- бинарные (в которых отношения связывают два объекта);
- N-арные (в которых есть специальные отношения, связывающие более двух понятий).

Наиболее часто в семантических сетях используются следующие отношения:

- элемент класса (роза –*это* цветок);
- атрибутивные связи /иметь свойство (память *имеет свойство* – объем);
- *значение* свойства (цвет *имеет значение*желтый);
- пример элемента класса (роза, *например*,чайная);
- связи типа *«часть-целое»*(велосипед *включает* руль);
- функциональные связи (определяемые обычно глаголами «производит», «влияет» и др.);
- количественные *(больше, меньше, равно*и др.*);*
- пространственные *(далеко от, близко от, за, под, над*и др.*);*
- временные*(раньше, позже, в течение*и др.*);*
- логические связи *(и, или, не)* и др.

Минимальный состав отношений в семантической сети таков:

- элемент класса или АКО;
- атрибутивные связи /иметь свойство;
- значение свойства.

Недостатком этой модели является сложность организации процедуры вывода.

Пример семантической сети для предложения типа «Поставщик осуществил поставку изделий для клиента до 1 июня 2010 года в количестве 10 000 штук» представлен на рис. 7.

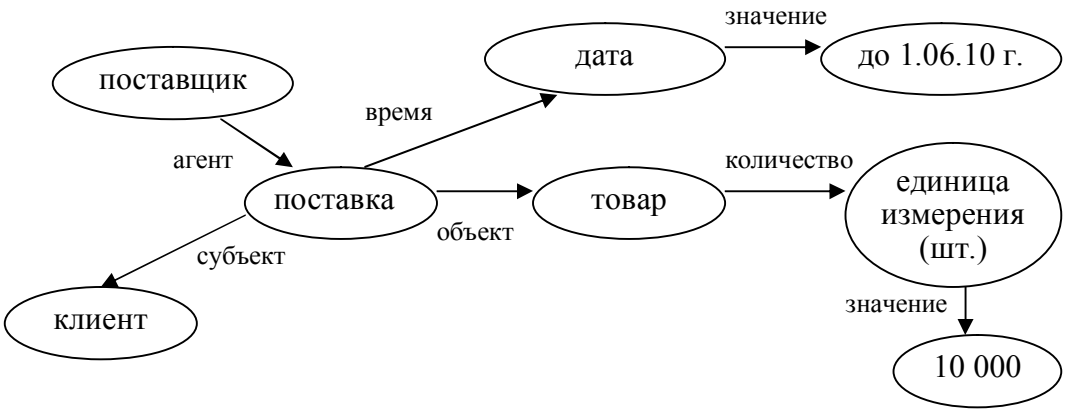

Рис. 7. **Пример семантической сети**

Фреймы

Фрейм – это минимально возможное описание сущности какого-либо события, ситуации, процесса или объекта.В историческом плане развитие фреймовой модели связано с теорией фреймов М. Минского, определяющей способ формализации знаний, используемый при решении задач распознавания образов (сцен) и понимания речи.«Отправным моментом для данной теории служит тот факт, что человек, пытаясь познать новую для себя ситуацию или по-новому взглянуть на уже привычные вещи, выбирает из своей памяти некоторую структуру данных (образ), называемую нами фреймом, с таким расчетом, чтобы путем изменения в ней отдельных деталей сделать ее пригодной для понимания более широкого класса явлений или процессов». Другими словами, фрейм – это форма описания знаний, очерчивающая рамки рассматриваемого (в текущей ситуации при решении данной задачи) фрагмента предметной области.

Модель фрейма является достаточно универсальной, поскольку позволяет отобразить все многообразие знаний о мире через:

- фреймы-структуры, для обозначения объектов и понятий (заем, залог, вексель);
- фреймы-роли (менеджер, кассир, клиент);
- фреймы-сценарии (банкротство, собрание акционеров, празднование именин);
- фреймы-ситуации (тревога, авария, рабочий режим устройства) и др.

Различают *фреймы-образцы,*или *прототипы,*и *фреймы-экземпляры,* которые создаются для отображения реальных фактических ситуаций на основе поступающих данных.

Фрейм имеет имя (название) и состоит из *слотов.*

Традиционно структура фрейма может быть представлена как список свойств:

(ИМЯ ФРЕЙМА:
(имя 1-го слота: значение 1-го слота),
(имя 2-го слота: значение 2-го слота),
……..
(имя N-го слота: значение N-го слота)).

Ту же запись можно представить в виде таблицы (см. табл. 1), дополнив ее двумя столбцами.

Таблица 1.Структура фрейма

Имя фрейма			
Имя слота	**Значение слота**	**Способ получения значения**	**Присоединенная процедура**

В таблице дополнительные столбцы (3-й и 4-й) предназначены для описания способа получения слотом его значения и возможного присоединения к тому или иному слоту специальных процедур, что допускается в теории фреймов. В качестве значения слота может выступать имя другого фрейма – так образуются сети фреймов.

Широко известны такие фреймо-ориентированные экспертные системы, как ANALYST, МОДИС.

§2. Нейронные сети

Так как основная задача искусственного интеллекта – моделирование рассуждений, а природное «устройство, способное думать» – это мозг, то очевидной задачей является создание «искусственного мозга» «по образу и подобию» человеческого. Исследованием этого вопроса стали заниматься в рамках направления искусственного интеллекта «нейрокибернетика».

Устройства мозга изучают такие науки, как психология, нейрофизиология, нейробиология, обладающие достаточным объемом знаний. Основной идеей нейрокибернетики стало воссоздание «в железе» клетки мозга – нейрона.

Нейроны – специализированные клетки, способные принимать, обрабатывать, кодировать, передавать и хранить информацию, организовывать реакции на раздражения, устанавливать контакты с другими нейронами, клетками органов. Уникальной особенностью нейрона является способность генерировать электрические разряды и передавать информацию с помощью специализированных окончаний – синапсов.

Число нейронов мозга человека приближается к 10^{11}. На одном нейроне может быть до 10 000 синапсов. Если только эти элементы считать ячейками хранения информации, то можно прийти к выводу, что нервная система может хранить 10^{19} ед. информации, то есть способна вместить практически все знания, накопленные человечеством.

На рис. 8 приведена схема строения «типичного» нейрона.

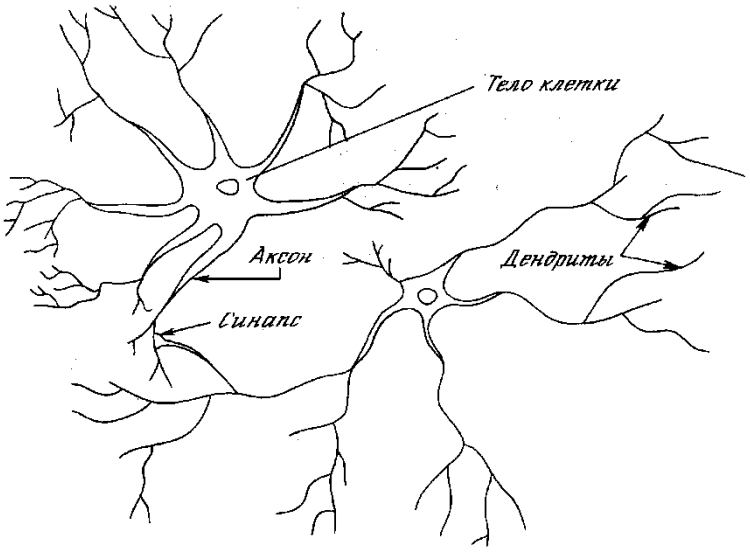

***Рис. 8.*Общая схема строения биологического нейрона**

Тело клетки содержит множество ветвящихся отростков двух типов. Отростки первого типа, называемые дендритами за их сходство с кроной раскидистого дерева, служат в качестве входных каналов для нервных импульсов от других нейронов. Эти импульсы поступают в сому, или тело, клетки размером от 3 до 100 микрон, вызывая ее специфическое возбуждение, которое затем распространяется по выводному отростку

второго
типа – аксону. Длина аксонов обычно заметно превосходит размеры дендритов, в отдельных случаях достигая десятков сантиметров и даже метров. Нейрон может находиться в двух состояниях: возбужденном или невозбужденном.

1943 год стал годом рождения теории искусственных нейронных сетей. Дж. Маккалок и У. Питт предложили модель формального нейрона (рис. 9) и описали основные принципы построения нейронных сетей.

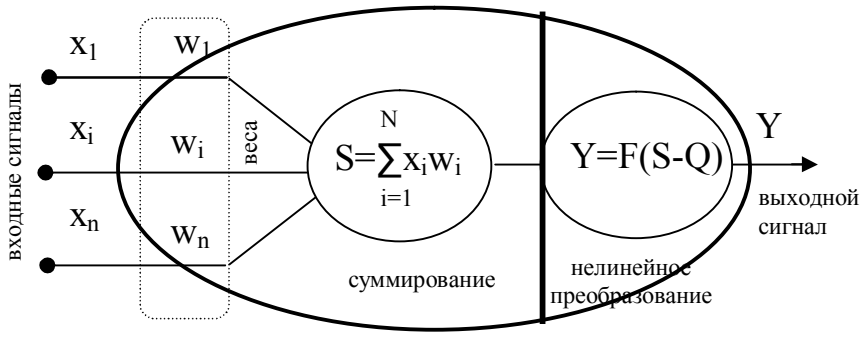

Рис. 9. **Модель формального нейрона**

Синапсы нейрона – места контактов нервных волокон – согласно этой модели, передают сигналы, определяя силу воздействия сигнала с этого входа на выходной сигнал, поступающий на аксон. Для этого каждому входу ставится в соответствие весовой коэффициент w_i. Дендриты получают входной сигнал, представленный вектором x_i. Затем нейрон обрабатывает поступивший сигнал, производя взвешенное суммирование и нелинейное преобразование, используя для этого активационную функцию (функции активации могут быть разными, наиболее распространенные – линейная, пороговая и сигмойда), аргументом которой будет результат суммирования минус пороговое значение. Это значение определяет уровень сигнала, на который нейрон будет реагировать.

Главными свойствами биологического нейрона являются его способность к обучению, универсальность, способность решать различные задачи. Описанная выше модель неспособна к этому. Обучаемой она стала лишь в 1949 году благодаря Д.Хеббу (D. Hebb), который, опираясь на физиологические и психологические исследования, выдвинул гипотезу об

обучаемости биологических нейронов. Его метод обучения стал отправной точкой для алгоритмов обучения нейронных сетей без учителя.

В 1957 году в мировом научном мире произошло второе по значимости в истории нейронных сетей событие: американский физиолог Ф. Розенблатт разработал модель зрительного восприятия и распознавания –персептрон (perceptron), а затем и построил первый нейрокомпьютер «Марк-1».

Нейронные сети были просты, понятны и многообещающи. Сложные процессы мышления, казалось, были готовы раскрыться перед человеком, для их описания требовались лишь элементарные математические операции: сложение, умножение и линейная функция. На персептрон, а затем и многослойный персептрон возлагались большие надежды, поэтому бурный энтузиазм, с которым он был принят, достаточно быстро сменился жесткой критикой.

В 1969 году вышла в свет книга «Персептроны» М. Минского и С. Паперта, которая ознаменовала окончание первого этапа в истории нейронных сетей. В этой книге был проведен анализ возможностей однослойных нейронных сетей и их ограничений.

Ограничения, обнаруженные Минским, были характерны однослойным нейронным сетям, но не многослойным нейронным сетям, которым была присуща другая проблема – их обучение.

Предложенный в 1986 году Д. Румельхардоми другими учеными алгоритм обратного распространения ошибки стал одним из ведущих факторов, породивших современный нейросетевой бум, так как являлся эффективным способом обучения нейронной сети достаточно произвольной структуры.

Нейронные сети могут реализовываться как программно, так и аппаратно. Постепенно направление нейрокибернетики преобразовалось в нейрокомпьютинг – шестое поколение компьютеров, базирующееся на нейронных сетях, имеющее свои достоинства и недостатки. Сравнительный анализ машины фон Неймана, идеи которой воплощены в первых 4 поколениях, и биологической нейронной системы, являющейся прообразом нейрокомпьютера, представлен в таблице 2.

Таблица 2. Сравнение машины фон Неймана с биологической нейронной системой

	Машина фон Неймана	**Биологическая нейронная система**
Процессор	Сложный	Простой
	Высокоскоростной	Низкоскоростной
	Один или несколько	Большое количество
Память	Отделена от процессора	Интегрирована в процессор
	Локализована	Распределенная
	Адресация не по содержанию	Адресация по содержанию
Вычисления	Централизованные	Распределенные
	Последовательные	Параллельные
	Хранимые программы	Самообучение
Надежность	Высокая уязвимость	Живучесть
Специализация	Численные и символьные операции	Проблемы восприятия
Среда функционирования	Строго определенная	Плохо определенная
	Строго ограниченная	Без ограничений

Классификация искусственных нейронных сетей

По топологии (рис. 10, 11):

- полносвязные (каждый нейрон связан со всеми остальными нейронами, в том числе и сам с собой);
- многослойные (нейроны располагаются слоями, и каждый нейрон последующего слоя связан со всеми нейронами текущего слоя).

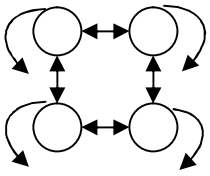

Рис. 10. **Полносвязная нейронная сеть**

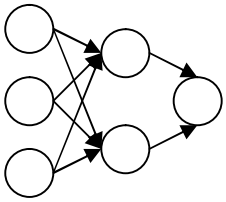

Рис. 11. **Слоистая нейронная сеть**

По организации обучения:

- с учителем (нейронную сеть обучают, подавая на вход значения обучающей выборки и предоставляя требуемые выходные значения);

• без учителя (на входы нейронной сети подают множество объектов, и нейронная сеть сама делит их на кластеры или классы).

По типам структур (рис. 12):

• нейроны с одним типом функции активации (все нейроны сети имеют одну функцию активации f(x), например линейную);

• нейроны с несколькими типами функций активации (нейроны сети имеют различные функции активации).

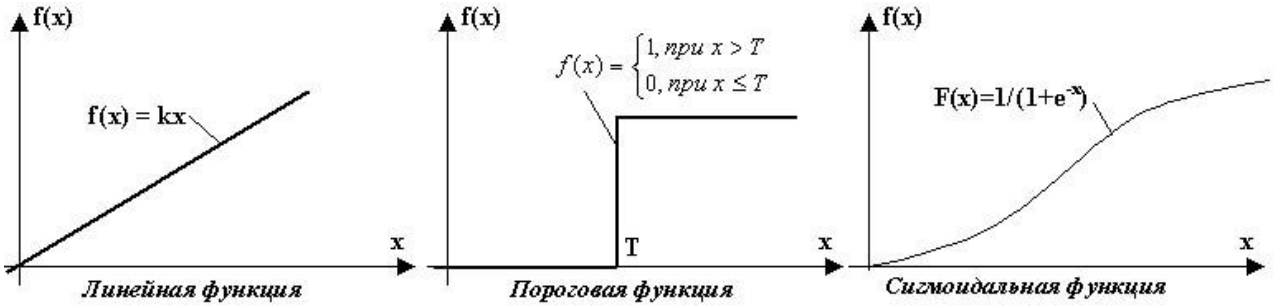

***Рис. 12.*Активационные функции**

По типу связей:

• прямого распространения (без обратных связей между нейронами; к таким сетям относятся однослойный и многослойный персептроны, сеть радиальных базисных функций);

• рекуррентные (с обратной связью, от выходов нейронов к входам; к таким сетям относятся соревновательные сети и сеть Хопфилда).

По типу сигнала:

• бинарные (на входы подаются только нули и единицы);

• аналоговые (на входы нейронов подаются значения непрерывных функций).

Однослойные искусственные нейронные сети

Один нейрон способен выполнять простейшие процедуры распознавания, но только соединение нескольких нейронов способно решить практически полезную задачу. Простейшая сеть (рис. 13) состоит из группы нейронов, образующих слой.

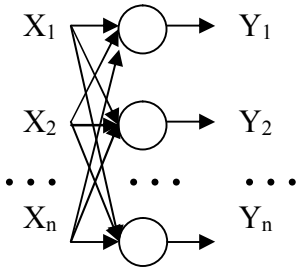

Рис. 13. **Однослойная нейронная сеть**

Обучение по дельта-правилу

Дельта-правило является обобщением алгоритма обучения персептрона. Дельта-правило работает только с непрерывными, дифференцируемыми функциями в режиме обучения с учителем (supervisedlearning). Ошибка, вычисляемая в процессе обучения сети, – это функция, характеризующая качество обучения данной сети, поэтому процесс обучения нейронной сети можно представить как процесс минимизации функции ошибки. Направление изменения значения функции можно установить, вычислив производную для функции одного переменного или градиент для функции многих переменных. При минимизации значения функции многих переменных меньшее значение необходимо искать в направлении антиградиента.

В данном алгоритме обучения начальные веса могут быть любыми.

Процесс обучения можно считать завершенным, если достигнута некая заранее установленная минимальная ошибка или алгоритм проработал условленное количество раз.

Алгоритм обучения по дельта-правилу:

1 шаг: инициализация матрицы весов (и порогов, в случае использования пороговой функции активации) случайным образом.

2 шаг:предъявление нейронной сети образа (на вход подаются значения из обучающей выборки – вектор X, берется соответствующий выход – вектор D).

3 шаг: вычисление выходных значений нейронной сети (вектор Y).

4 шаг: вычисление для каждого нейрона величины расхождения реального результата с желаемым:

$$\varepsilon_i = (d_i - y_i),$$

где d_i– желаемое выходное значение на i-нейроне, y_i– реальное значение на i-нейроне.

5 шаг:изменение весов (и порогов при использовании пороговой функции) по формулам:

$$w_{ij}(t+1) = w_{ij}(t) - \eta \cdot \varepsilon_i \cdot x_j,$$

$$\theta_i(t+1) = \theta_i(t) - \eta \cdot \varepsilon_i,$$

где t – номер текущей итерации цикла обучения, w_{ij} – вес связи j-входа с i-нейроном, η – коэффициент обучения (задается от 0 до 1), x_j – входное значение, θ_i – пороговое значение i-нейрона.

6 шаг:проверка условия продолжения обучения (вычисление значения ошибки и/или проверка заданного количества итераций). Если обучение не завершено, то 2 шаг, иначе заканчиваем обучение.

Многослойные нейронные сети

Однослойные нейронные сети имеют свои ограничения, на которые указал Минский, многослойные сети свободны от них. Они обладают большими вычислительными возможностями. Хотя созданы сети всех конфигураций, какие только можно себе представить, послойная организация нейронов копирует слоистые структуры определенных отделов мозга. Многослойные сети образовываются каскадами слоев. Выход одного слоя является входом для последующего слоя, такая организация сети образует многослойную нейронную сеть с прямым распространением (рис. 14).

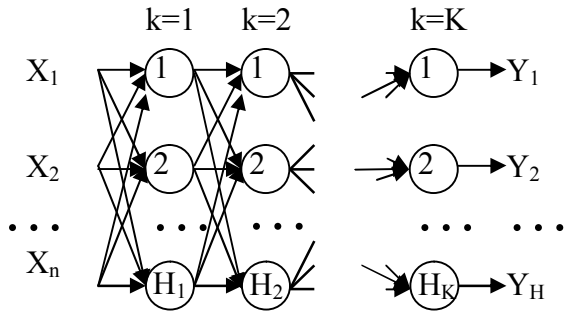

***Рис. 14.*Многослойная нейронная сеть прямого распространения**

Многослойная сеть может содержать произвольное количество слоев (K), каждый слой состоит из нескольких нейронов, число которых также

может быть произвольно (H_k – количество нейронов в слое), количество входов n, количество выходов $H=H_K$– числу нейронов в выходном (последнем) слое.

Слои между первым и последним называются промежуточными или скрытыми. Веса в такой сети имеют три индекса: i – номер нейрона текущего слоя, для которого связь входная; j – номер входа или нейрона слоя, для которого связь выходная; k – номер текущего слоя в нейронной сети (для входов, вектора X, k=0).

Обучение методом обратного распространения ошибки

Обучение алгоритмом обратного распространения ошибки предполагает два прохода по всем слоям сети: прямой и обратный.

При прямом проходе входной вектор подается на входной слой нейронной сети, после чего распространяется по сети от слоя к слою. В результате генерируется набор выходных сигналов, который и является фактической реакцией сети на данный входной образ. Во время прямого прохода все синаптические веса сети фиксированы.

Во время обратного прохода все синаптические веса настраиваются в соответствии с правилом коррекции ошибок, а именно: фактический выход сети вычитается из желаемого, в результате чего формируется сигнал ошибки. Этот сигнал впоследствии распространяется по сети в направлении, обратном направлению синаптических связей. Отсюда и название – алгоритм обратного распространения ошибки. Синаптические веса настраиваются с целью максимального приближения выходного сигнала сети к желаемому.

Алгоритм обучения по дельта-правилу:

1 шаг: инициализация матриц весов случайным образом (в циклах).

2 шаг:предъявление нейронной сети образа (на вход подаются значения из обучающей выборки – вектор X и берется соответствующий выход – вектор D).

3 шаг (прямой проход): вычисление в циклах выходов всех слоев и получение выходных значений нейронной сети (вектор Y):

$$y_i^k = f(\sum_{j=0}^{H_{k-1}} w_{ij}^k \cdot y_j^{k-1}),$$

$$y_j^0 = x_j,$$

$$y_0^{k-1} = 1,$$

$$x_0 = 1,$$

где y_i^k – выход i-нейрона k-слоя, f – функция активации, w_{ij}^k – синаптическая связь между j-нейроном слоя k-1 и i-нейроном слоя k, x_j – входное значение.

4 шаг (обратный проход): изменение весов в циклах по формулам:

$$w_{ij}^k(t+1) = w_{ij}^k(t) + \eta \cdot \delta_i^k \cdot y_j^{k-1},$$

$$\delta_i^k = (d_i - y_i) \cdot y_i \cdot (1 - y_i)$$

– для последнего (выходного) слоя,

$$\delta_i^k = y_i \cdot (1 - y_i) \cdot \sum_{l=1}^{H_{k+1}} \delta_l^{k+1} \cdot w_l^{k+1}$$

– для промежуточных слоев, где t – номер текущей итерации цикла обучения (номер эпохи), η – коэффициент обучения (задается от 0 до 1), y_i^k – выход i-нейрона k-слоя, w_{ij}^k –синаптическая связь между j-нейроном слоя k-1 и i-нейроном слоя k, d_i – желаемое выходное значение на i-нейроне, y_i – реальное значение на i-нейроне выходного слоя.

5 шаг:проверка условия продолжения обучения (вычисление значения ошибки и/или проверка заданного количества итераций). Если обучение не завершено, то 2 шаг, иначе заканчиваем обучение. Среднеквадратичная ошибка вычисляется следующим образом:

$$\varepsilon = \frac{1}{Q} \cdot \sum_{q=1}^{Q} \sum_{i=1}^{H} (d_i - y_i)^2,$$

где Q – общее число примеров, H – количество нейронов в выходном слое, d_i – желаемое выходное значение на i-нейроне, y_i – реальное значение на i-нейроне выходного слоя.

Задачи, решаемые нейронными сетями

1. ***Классификация образов.*** Задача состоит в определении принадлежности входного образа (например, языкового сигнала или рукописного символа), представленного вектором признаков, к одному или нескольким предварительно определенным классам. К известным приложениям относятся распознавание букв, распознавание языка, классификация сигнала электрокардиограммы, классификация клеток крови.

2. ***Кластеризация/категоризация.*** При решении задачи кластеризации обучающее множество не имеет меток классов. Алгоритм кластеризации основан на подобии образов и помещает похожие образы в один кластер. Известны случаи применения кластеризации для добычи знаний, сжатия данных и исследования свойств данных.

3. ***Аппроксимация функций.*** Предположим, что есть обучающая выборка $((x_1,y_1), (x_2,y_2), ..., (x_n,y_n))$ (пары данных вход-выход), которая генерируется неизвестной функцией F, искаженной шумом. Задача аппроксимации состоит в нахождении неизвестной функции F. Аппроксимация функций необходима при решении многочисленных инженерных и научных задач моделирования.

4. ***Предвидение/прогноз.*** Пусть заданы n дискретных отсчетов $\{y(t_1), y(t_2), ..., y(t_n)\}$ в последовательные моменты времени $t_1, t_2,..., t_n$. Задача состоит в предвидении значения $y(t_n+1)$ в следующий момент времени t_n+1. Предвидение/прогноз имеют большое значение для принятия решений в бизнесе, науке и технике (предвидение цен на фондовой бирже, прогноз погоды).

5. ***Оптимизация.*** Многочисленные проблемы в математике, статистике, технике, науке, медицине и экономике могут рассматриваться как проблемы оптимизации. Задачей алгоритма оптимизации является нахождение такого решения, которое удовлетворяет системе ограничений и максимизирует или минимизирует целевую функцию.

6. ***Память, адресуемая по содержанию.*** В традиционных компьютерах обращение к памяти доступно только с помощью адреса, не зависящего от содержания памяти. Более того, если допущена ошибка в вычислении адреса, то может быть найдена совсем другая информация. Ассоциативная память или память, адресуемая по смыслу, доступна по

указанию заданного содержания. Содержимое памяти может быть вызвано даже по частичному входу или при поврежденном содержании. Ассоциативная память может быть использована в мультимедийних информационных базах данных.

7. ***Управление.*** Рассмотрим динамическую систему, заданную совокупностью {u(t), y(t)}, где u(t) – входное управляющее воздействие, а y(t) –выход системы в момент времени t. В системах управления с эталонной моделью целью управления является расчет такого входного воздействия u(t), при котором система действует по желательной траектории, заданной эталонной моделью. Примером является оптимальное управление двигателем.

§3. Эволюционное моделирование

Одной из главных характеристик искусственного интеллекта как науки является его междисциплинарность, позволяющая привлекать интересные идеи, теории из других областей знаний, адаптируя и используя готовые разработки для своих задач. Так было с нейронными сетями, с моделированием рассуждений, с компьютерной лингвистикой и т.д. Многие значимые теории науки были так или иначе рассмотрены через призму искусственного интеллекта. Теория Ч. Дарвина (1859 г.) стала отправной точкой для еще одного направления исследований – эволюционного моделирования.

Основной тезис эволюционного моделирования – заменить процесс моделирования сложного объекта моделированием его эволюции. Он направлен на применение механизмов естественной эволюции при синтезе сложных систем обработки информации. Дарвин сформулировал основной закон развития органического мира, охарактеризовав его взаимодействием трех следующих факторов:

- наследственность (потомки сохраняют свойства родителей);
- изменчивость (потомки почти всегда не идентичны);
- естественный отбор (выживают наиболее приспособленные).

Теория Дарвина, дополненная генетическими знаниями, называется синтетической теорией эволюции. Случайное появление новых признаков она объяснила мутациями – изменениями, возникающими в ДНК организмов.

Понятие «эволюционное моделирование» сформировалось в работах Л. Фогеля, А. Оуэне, М. Уолша. В 1966 году вышла их совместная книга «Искусственный интеллект и эволюционное моделирование». История эволюционных вычислений началась с разработки ряда различных независимых моделей. Основными из них были генетические алгоритмы (ГА) и классификационные системы Д. Холланда (Holland), опубликованные в начале 60-х годов и получившие всеобщее признание после выхода в свет книги, ставшей классикой в этой области, «Адаптация в естественных и искусственных системах» (AdaptationinNaturalandArtificalSystems, 1975). В 70-х годах в рамках теории случайного поиска Л.А. Растригиным был предложен ряд алгоритмов, использующих идеи бионического поведения особей. Развитие этих идей нашло отражение в цикле работ И.Л. Букатовой по эволюционному моделированию. Развивая идеи М.Л. Цетлина о целесообразном и оптимальном поведении стохастических автоматов, Ю.И. Неймарк предложил осуществлять поиск глобального экстремума на основе коллектива независимых автоматов, моделирующих процессы развития и элиминации особей. Несмотря на разницу в подходах, все они базировались на принципах эволюции.

В рамках эволюционного моделирования создавались и исследовались модели происхождения молекулярно-генетических систем обработки информации, модели, характеризующие общие закономерности эволюционных процессов, и производился анализ моделей искусственной «эволюции» с целью применения метода эволюционного поиска к практическим задачам оптимизации.

В начале 70-х годов лауреат Нобелевской премии М. Эйген предпринял впечатляющую попытку построения моделей возникновения в ранней биосфере Земли молекулярно-генетических систем обработки информации (в модели квазивидов рассматривается поэтапная эволюция популяции информационных последовательностей (векторов), компоненты которых принимают небольшое число дискретных значений).

Вслед за Эйгеном в 1980 году новосибирскими учеными В. Ратнером и В.Шаминым была предложена модель сайзеров (модель сайзеров в простейшем случае рассматривает систему из трех типов макромолекул: полинуклеотидной матрицы и ферментов трансляции и репликации, кодируемых этой матрицей; полинуклеотидная матрица – это как бы запоминающее устройство, в котором хранится информация о функциональных единицах сайзера – ферментах). С. Кауфман с сотрудниками из Пенсильванского университета исследуют эволюцию автоматов, состоящих из соединенных между собой логических элементов.

Д. Коза разработал метод, позволяющий совершенствовать реальные технические системы методами генетического программирования. При этом эволюция затрагивает не отдельные численные параметры, а целые системы (с помощью этого метода удалось заново открыть 15 запатентованных схемотехнических решений: усилители, фильтры, контролеры и т.д.).

Достоинства эволюционных вычислений:

1) широкая область применения;

2) возможность проблемно-ориентированного кодирования решений, подбора начальной популяции, комбинирования эволюционных вычислений с неэволюционными алгоритмами, продолжения процесса эволюции до тех пор, пока имеются необходимые ресурсы;

3) пригодность для поиска в сложном пространстве решений большой размерности;

4) отсутствие ограничений на вид целевой функции;

5) ясность схемы и базовых принципов эволюционных вычислений;

6) интегрируемость эволюционных вычислений с другими неклассическими парадигмами искусственного интеллекта, такими как искусственные нейросети и нечеткая логика.

*Недостатки*эволюционных вычислений:

1) эвристический характер эволюционных вычислений не гарантирует оптимальности полученного решения;

2) относительно высокая вычислительная трудоемкость, которая преодолевается за счет распараллеливания на уровне организации эволюционных вычислений и на уровне их непосредственной реализации в вычислительной системе;

3) относительно невысокая эффективность на заключительных фазах моделирования эволюции (операторы поиска в эволюционных алгоритмах не ориентированы на быстрое попадание в локальный оптимум);

4) нерешенность вопросов самоадаптации.

К *методам эволюционного моделирования* относятся:

- метод группового учета аргументов:
  1) берется самый последний слой классификаторов;
  2) генерируется из них по определенным правилам новый слой классификаторов, которые теперь сами становятся последним слоем;
  3) отбирается из них F лучших, где F – ширина отбора (селекции);
  4) если не выполняется условие прекращения селекции (наступление вырождения), переход на п. 1, иначе лучший классификатор объявляется искомым решением задачи идентификации;
- эволюционное (генетическое) программирование (данные, которые закодированы в генотипе, могут представлять собой команды какой-либо виртуальной машины, в простейшем случае ничего не меняется в генетическом алгоритме, но тогда длина получаемой последовательности действий (программы) получается не отличающейся от начальных; современные алгоритмы генетического программирования распространяют генетические алгоритмы для систем с переменной длиной генотипа);
- генетические алгоритмы.

Генетические алгоритмы

«Отцом» генетических алгоритмов по праву считается Д. Холланд, метод вначале назывался репродуктивным планом Холланда. В дальнейшем генетические алгоритмы развивались в работах учеников Холланда: Д. Голдберга и К. Де Йонга– именно в них и закрепилось название метода.

Генетические алгоритмы – это раздел эволюционного моделирования, заимствующий методические приемы из теоретических положений генетики.

Генетические алгоритмы – адаптивные методы поиска, которые используются для решения задач функциональной оптимизации. Представляют собой своего рода модели машинного исследования поискового пространства, построенные на эволюционной метафоре. Характерные особенности: использование строк фиксированной длины для представления генетической информации, работа с популяцией строк, использование генетических операторов для формирования будущих поколений.

Генетические алгоритмы оперируют совокупностью особей (популяцией), которые представляют собой строки, кодирующие одно из решений задачи. Этим метод отличается от большинства других алгоритмов оптимизации, которые оперируют лишь с одним решением, улучшая его.

Генетические алгоритмы *применяются* для решения следующих задач:

- оптимизация функций;
- разнообразные задачи на графах (задача коммивояжера, раскраска и т.д.);
- настройка и обучение искусственной нейронной сети;
- задачи компоновки;
- составление расписаний;
- игровые стратегии;
- аппроксимация функций;
- искусственная жизнь;
- биоинформатика.

Преимущества генетических алгоритмов:

1) универсальность;
2) высокая обзорность поиска;
3) нет ограничений на целевую функцию;
4) любой способ задания функции.

Недостатки генетических алгоритмов:

1) относительно высокая вычислительная стоимость;
2) квазиоптимальность.

Когда надо использовать генетический алгоритм: много параметров, плохая целевая функция, комбинаторные задачи.

*Когда не надо использовать генетический алгоритм: з*адача хорошо решается традиционными методами, требуется высокая точность решения.

Схема функционирования генетического алгоритма

Из биологии известно, что любой организм может быть представлен своим фенотипом, который фактически определяет, чем является объект в реальном мире, и генотипом, который содержит всю информацию об объекте на уровне хромосомного набора. При этом каждый ген, то есть элемент информации генотипа, имеет свое отражение в фенотипе. Таким образом, для решения задач необходимо представить каждый признак объекта в форме, подходящей для использования в генетическом алгоритме. Все дальнейшее функционирование механизмов генетического алгоритма производится на уровне генотипа, позволяя обойтись без информации о внутренней структуре объекта, что и обуславливает его широкое применение в самых разных задачах.

В наиболее часто встречающейся разновидности генетического алгоритма для представления генотипа объекта применяются битовые строки. При этом каждому атрибуту объекта в фенотипе соответствует один ген в генотипе объекта. Набор генов или один ген представляют собой закодированный признак. Ген – это битовая строка, чаще всего фиксированной длины. Для кодирования гена в бинарной реализации генетического алгоритма часто используют код Грея (см. пример в табл.3).

Таблица 3. Пример фенотипа

Признак	**Двоичное значение признака**	**Десятичное значение признака**
Признак 1	0011	3
Признак 2	1100	12
Признак 3	1110	14
Признак 4	0111	7
Признак 5	1001	9

После определения фенотипа генетический алгоритм функционирует по следующей схеме действий (рис. 15):

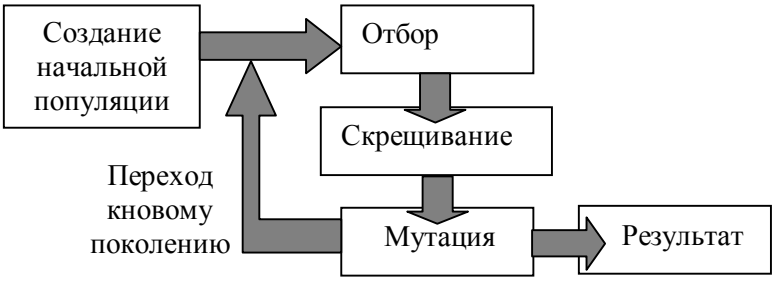

***Рис. 15.*Схема функционирования генетического алгоритма**

1. Формирование начальной популяции.
2. Оценка особей популяции.
3. Отбор (селекция).
4. Скрещивание.
5. Мутация.
6. Формирование новой популяции.
7. Если популяция не сошлась, то 2, иначе – останов (прекращение функционирования генетического алгоритма).

Формирование начальной популяции

Стандартный генетический алгоритм начинает свою работу с формирования начальной популяции – конечного набора допустимых решений задачи. Эти решения могут быть выбраны случайным образом или получены с помощью простых приближенных алгоритмов. Выбор начальной популяции не имеет значения для сходимости процесса, однако формирование «хорошей» начальной популяции (например, из множества локальных оптимумов) может заметно сократить время достижения глобального оптимума. Если отсутствуют предположения о местоположении глобального оптимума, то индивидов из начальной популяции желательно распределить равномерно по всему пространству поиска решения.

Популяция инициируется в начальный момент времени t=0 и состоит из k особей, каждая из которых представляет возможное решение рассматриваемой проблемы. Особь – это одна или несколько хромосом (обычно одна). Хромосома состоит из генов, то есть это битовая строка (хромосомы не ограничены бинарным представлением, есть реализации генетического алгоритма, построенные на векторах вещественных чисел).

Гены располагаются в различных позициях хромосомы и принимают значения, называемые аллелями.

Оценка особей популяции

Для решения задачи с помощью генетического алгоритма необходимо задать меру качества для каждого индивида в пространстве поиска. Для этой цели используется функция приспособленности (fitnessfunction). Функция приспособленности должна принимать неотрицательные значения на ограниченной области определения, при этом совершенно не требуются непрерывность и дифференцируемость. Значение этой функции определяет, насколько хорошо подходит особь для решения задачи.

В задачах максимизации целевая функция часто сама выступает в качестве функции приспособленности, для задач минимизации целевую функцию следует инвертировать. Если решаемая задача имеет ограничения, выполнение которых невозможно контролировать алгоритмически, то функция приспособленности, как правило, включает также штрафы за невыполнение ограничений (они уменьшают ее значение).

Отбор (селекция)

На каждом шаге эволюции с помощью вероятностного оператора селекции (отбора) выбираются два решения-родителя для их последующего скрещивания. Среди операторов селекции наиболее распространенными являются два вероятностных оператора пропорциональной и турнирной селекции. В некоторых случаях также применяется отбор усечением.

Пропорциональный отбор (proportionalselection)

Каждой особи назначается вероятность $P_s(i)$, равная отношению ее приспособленности к суммарной приспособленности популяции. Затем происходит отбор (с замещением) всех n особей для дальнейшей генетической обработки согласно величине $P_s(i)$.

Простейший пропорциональный отбор – рулетка – отбирает особей с помощью n «запусков» рулетки. Колесо рулетки содержит по одному сектору для каждого члена популяции. Размер i-го сектора пропорционален соответствующей величине $P_s(i)$. При таком отборе члены

популяции с более высокой приспособленностью с большей вероятностью будут чаще выбираться, чем особи с низкой приспособленностью.

Турнирный отбор

Турнирный отбор реализуется следующим образом: из популяции, содержащей m особей, выбирается случайным образом t особей, и наиболее приспособленная особь записывается в промежуточный массив (между выбранными особями проводится турнир). Эта операция повторяется m раз. Строки в полученном промежуточном массиве затем используются для скрещивания (случайным образом). Размер группы строк, отбираемых для турнира, часто равен 2. В этом случае говорят о двоичном/парном турнире.

Отбор усечением

Данная стратегия использует отсортированную по убыванию популяцию. Число особей для скрещивания выбирается в соответствии с порогом $T \in [0;1]$. Порог определяет, какая доля особей, начиная с самой первой (самой приспособленной), будет принимать участие в отборе. В принципе, порог можно задать и равным 1, тогда все особи текущей популяции будут допущены к отбору. Среди особей, допущенных к скрещиванию случайным образом m/2 раза, выбираются родительские пары, потомки которых образуют новую популяцию.

Ранговый отбор

Этот вид отбора подразумевает следующее: для каждой особи ее вероятность попасть в промежуточную популяцию пропорциональна ее порядковому номеру в отсортированной по возрастанию приспособленности популяции. Такой вид отбора не зависит от средней приспособленности популяции.

Элитный отбор

Элитные методы отбора гарантируют, что при отборе обязательно будут выживать лучший или лучшие члены популяции. Наиболее распространена процедура обязательного сохранения только одной лучшей особи, если она не прошла, как другие, через процесс отбора, кроссовера и мутации. Элитизм может быть внедрен практически в любой стандартный метод отбора. Использование элитизма позволяет не потерять хорошее промежуточное решение, но в то же время из-за этого алгоритм может «застрять» в локальном экстремуме. В большинстве случаев элитизм не

вредит поиску решения, и главное – это предоставить алгоритму возможность анализировать разные хромосомы из пространства поиска.

Скрещивание

Как известно, в теории эволюции важную роль играет то, каким образом признаки родителей передаются потомкам. В генетических алгоритмах за передачу признаков родителей потомкам отвечает оператор, который называется оператором **скрещивания** (его также называют **кроссовер** или **кроссинговер**). Этот оператор определяет передачу признаков родителей потомкам, к ним применяется вероятностный оператор скрещивания, который строит на их основе новые (1 или 2) решения-потомки. Отобранные особи подвергаются кроссоверу (иногда называемому рекомбинацией) с заданной вероятностью P_c. Если каждая пара родителей порождает двух потомков, для воспроизводства популяции необходимо скрестить m/2 пары. Для каждой пары с вероятностью P_c применяется кроссовер. Следовательно, с вероятностью 1-P_cкроссовер не происходит, и тогда неизмененные особи переходят на следующую стадию (мутации).

Существует большое количество разновидностей оператора скрещивания. Простейший одноточечный кроссовер работает следующим образом.

Сначала случайным образом выбирается одна из возможных точек разрыва. (Точка разрыва – участок между соседними битами в строке.) Обе родительские структуры разрываются на два сегмента по этой точке. Затем соответствующие сегменты различных родителей склеиваются и получаются два генотипа потомков:

Родитель 1: 1 0 0 1 0 1 1 | 0 1 0 0 1
Родитель 2: 0 1 0 0 0 1 1 | 0 0 1 1 1

Потомок 1: 1 0 0 1 0 1 1 | 0 0 1 1 1
Потомок 2: 0 1 0 0 0 1 1 | 0 1 0 0 1

В настоящее время исследователи ГА предлагают много других операторов скрещивания. Двухточечный кроссовер и равномерный кроссовер– вполне достойные альтернативы одноточечному оператору. В двухточечном кроссовере выбираются две точки разрыва, и родительские хромосомы обмениваются сегментом, который находится между двумя этими точками. В равномерном кроссовере каждый бит первого потомка

случайным образом наследуется от одного из родителей; второму потомку достается бит другого родителя.

Мутация

Следующий генетический оператор предназначен для того, чтобы поддерживать разнообразие особей в популяции, – это оператор **мутации.**После того как закончится стадия кроссовера, потомки могут подвергаться случайным модификациям. В простейшем случае в каждой хромосоме, которая подвергается мутации, каждый бит с вероятностью P_mизменяется на противоположный (это так называемая одноточечная мутация):

Особь до мутации: 1 0 0 1 0 1 1 0 0 1 1 1
Особь после мутации: 1 0 0 1 0 1 0 0 0 1 1 1

Более сложной разновидностью мутации являются операторы инверсии и транслокации. Инверсия – это перестановка генов в обратном порядке внутри произвольно выбранного участка хромосомы:

Особь до инверсии: 1 0 0 1 1 1 1 0 0 1 1 1
Особь после инверсии: 1 0 0 1 0 0 1 1 1 1 1 1

Транслокация– это перенос какого-либо участка хромосомы в другой сегмент этой же хромосомы:

Особь до транслокации: 1 0 0 1 1 1 1 0 0 1 1 1
Особь после транслокации: 1 1 1 0 0 0 1 1 0 1 1 1

Все перечисленные генетические операторы (одноточечный и многоточечный кроссовер, одноточечная мутация, инверсия, транслокация) имеют схожие биологические аналоги.

Формирование нового поколения

После скрещивания и мутации особей необходимо решить проблему: какие из новых особей войдут в следующее поколение, а какие – нет, и что делать с их предками. Есть два наиболее распространенных способа.

1. Новые особи (потомки) занимают места своих родителей. После этого наступает следующий этап, в котором потомки оцениваются, отбираются, дают потомство и уступают место своим «детям».

2. Следующая популяция включает в себя как родителей, так и их потомков.

Во втором случае необходимо дополнительно определить, какие из особей родителей и потомков попадут в новое поколение. В простейшем случае в него после каждого скрещивания включаются две лучшие особи из четверки родителей и их потомков. Более эффективным является механизм вытеснения, который реализуется таким образом, что стремится удалять «похожие» хромосомы из популяции и оставлять отличающиеся.

Критерии останова

Другой важный момент функционирования алгоритма – определение критериев останова. Вообще говоря, такой процесс эволюции может продолжаться до бесконечности. Обычно в качестве критериев останова применяются или ограничение на максимальное число эпох функционирования алгоритма, или определение его сходимости, обычно путем сравнивания приспособленности популяции на нескольких эпохах и остановки при стабилизации этого параметра.

Схождением называется такое состояние популяции, когда все строки популяции почти одинаковы и находятся в области некоторого экстремума (рис. 16).

В такой ситуации кроссовер практически никак не изменяет популяции. А вышедшие из этой области за счет мутации особи склонны вымирать, так как чаще имеют меньшую приспособленность, особенно если

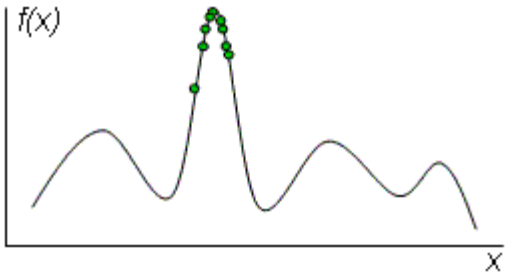

Рис. 16. **Схождение генетического алгоритма**

данный экстремум является глобальным максимумом. Таким образом, схождение популяции обычно означает, что найдено лучшее или близкое к нему решение.

Виды генетических алгоритмов

Существуют различные модели генетического алгоритма (классический, простой генетический алгоритм, гибридный, CHC генетический алгоритм и др.). Они различаются по стратегиям отбора и формирования нового поколения особей, операторами генетического алгоритма, кодированием генов и т.д.

CHC-алгоритм

CHC (Crossgenerationalelitistselection, Heterogenousrecombination, Cataclysmicmutation) был предложен Эсхелманом и характеризуется следующими параметрами:

1. Для нового поколения выбираются N лучших различных особей среди родителей и детей. Дублирование строк не допускается.

2. Для скрещивания выбирается случайная пара, но не допускается, чтобы между родителями было мало хэммингово расстояние или мало расстояние между крайними различающимися битами.

3. Для скрещивания используется разновидность однородного кроссовера HUX (HalfUniformCrossover): ребенку переходит ровно половина битов каждого родителя.

4. Размер популяции небольшой, около 50 особей. Этим оправдано использование однородного кроссовера.

CHC противопоставляет агрессивный отбор агрессивному кроссоверу, однако все равно малый размер популяции быстро приводит ее к состоянию, когда создаются только более или менее одинаковые строки. В таком случае CHC применяет cataclysmicmutation: все строки, кроме самой приспособленной, подвергаются сильной мутации (изменяется около трети битов). Таким образом, алгоритм перезапускается и далее продолжает работу, применяя только кроссовер.

Genitor

Этот алгоритм был создан Д. Уитли. Genitor-подобные алгоритмы отличаются от классического ГА следующими тремя свойствами:

1. На каждом шаге только одна пара случайных родителей создает только одного ребенка.

2. Этот ребенок заменяет не родителя, а одну из худших особей популяции (в первоначальном Genitor– самую худшую).

3. Отбор особи для замены производится по ее рейтингу, а не по приспособленности.

В Genitor поиск гиперплоскостей происходит лучше, а сходимость быстрее, чем у классического генетического алгоритма, предложенного Холландом.

Гибридные алгоритмы

Идея гибридных алгоритмов (hybridalgorithms) заключается в сочетании генетического алгоритма с некоторым другим методом поиска, подходящим в данной задаче. В каждом поколении каждый полученный потомок оптимизируется этим методом, после чего производятся обычные для генетического алгоритма действия.

Такой вид развития называется ламарковой эволюцией, при которой особь способна обучаться, а затем полученные навыки записывать в собственный генотип, чтобы потом передать их потомкам. И хотя такой метод ухудшает способность алгоритма искать решение с помощью отбора гиперплоскостей, однако на практике гибридные алгоритмы оказываются очень удачными. Это связано с тем, что обычно велика вероятность того, что одна из особей попадет в область глобального максимума и после оптимизации окажется решением задачи.

Параллельные генетические алгоритмы

Генетические алгоритмы можно организовать как несколько параллельно выполняющихся процессов, это увеличит их производительность.

Рассмотрим переход от классического генетического алгоритма к параллельному. Для этого будем использовать турнирный отбор. Заведем N / 2 процесса (здесь и далее процесс подразумевается как некоторая машина, процессор, который может работать независимо). Каждый из них будет выбирать случайно из популяции 4 особи, проводить 2 турнира и скрещивать победителей. Полученные дети будут записываться в новое поколение. Таким образом, за один цикл работы одного процесса будет сменяться целое поколение.

Островная модель

Островная модель (islandmodel, рис. 17) – это тоже модель параллельного генетического алгоритма. Она заключается в следующем:

пусть у нас есть 16 процессов и 1600 особей. Разобьем их на 16 подпопуляций по 100 особей. Каждая их них будет развиваться отдельно с помощью некого генетического алгоритма. Таким образом, можно сказать, что мы расселили особи по 16-ти изолированным островам.

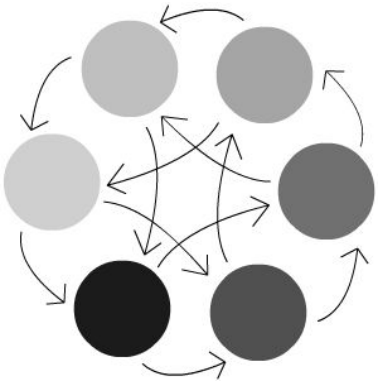

Рис. 17. **Островная модель генетического алгоритма**

Изредка (например, каждые 5 поколений) процессы (или острова) будут обмениваться несколькими хорошими особями. Этот процесс называется миграцией. Миграция позволяет островам обмениваться генетическим материалом.

Так как населенность островов обычно бывает невелика, подпопуляции будут склонны к преждевременной сходимости. Поэтому важно правильно установить частоту миграции. Чересчур частая миграция (или миграция слишком большого числа особей) приведет к смешению всех подпопуляций, и тогда островная модель будет несильно отличаться от обычного генетического алгоритма. Если же миграция будет слишком редкой, то она не сможет предотвратить преждевременного схождения подпопуляций.

Генетические алгоритмы стохастичны, поэтому при разных запусках популяция может сходиться к разным решениям (хотя все они в некоторой степени «хорошие»). Островная модель позволяет запустить алгоритм сразу несколько раз и пытаться совмещать «достижения» разных островов для получения в одной из подпопуляций наилучшего решения.

Ячеистые генетические алгоритмы

Ячеистые генетические алгоритмы (CellularGeneticAlgorithms) – модель параллельных генетических алгоритмов. Пусть дано 2500

процессов, расположенных на сетке размером 50×50 ячеек, замкнутой, как показано на рис. 18 (левая сторона замыкается с правой, верхняя – с нижней, получается тор).

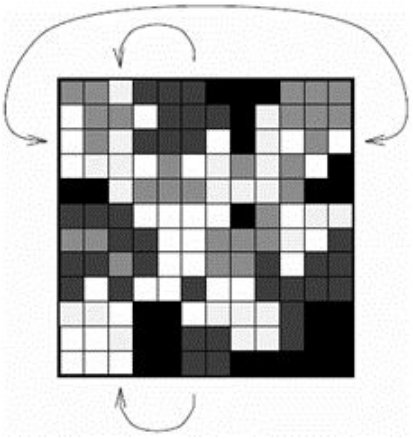

Рис. 18. **Ячеистый генетический алгоритм**

Каждый процесс может взаимодействовать только с четырьмя своими соседями (сверху, снизу, слева, справа). В каждой ячейке находится ровно одна особь. Каждый процесс будет выбирать лучшую особь среди своих соседей, скрещивать с ней особь из своей ячейки и одного полученного ребенка помещать в свою ячейку вместо родителя.

По мере работы такого алгоритма возникают эффекты, похожие на островную модель. Сначала все особи имеют случайную приспособленность (на рисунке она определяется по цвету). Спустя несколько поколений образуются небольшие области похожих особей с близкой приспособленностью. По мере работы алгоритма эти области растут и конкурируют между собой.

§4. Нечеткие множества и нечеткая логика

В основе нечеткой логики лежит теория нечетких множеств, изложенная в серии работ Л. Заде в 1965-1973 годах. Математическая теория нечетких множеств (fuzzysets) и нечеткая логика (fuzzylogic) являются обобщениями классической теории множеств и классической формальной логики. Основной причиной появления новой теории стало

наличие нечетких и приближенных рассуждений при описании человеком процессов, систем, объектов.

Л. Заде, формулируя это главное свойство нечетких множеств, базировался на трудах предшественников. В начале 1920-х годов польский математик Лукашевич трудился над принципами многозначной математической логики, в которой значениями предикатов могли быть не только «истина» или «ложь». В 1937 году еще один американский ученый М.Блэк впервые применил многозначную логику Лукашевича к спискам как множествам объектов и назвал такие множества неопределенными.

Прежде чем нечеткий подход к моделированию сложных систем получил признание во всем мире, с момента зарождения теории нечетких множествпрошло не одно десятилетие.

Нечеткая логика как научное направление развивалась непросто, не избежала она и обвинений в лженаучности. Даже в 1989 году, когда примеры успешного применения нечеткой логики в обороне, промышленности и бизнесе исчислялись десятками, Национальное научное общество США обсуждало вопрос об исключении материалов по нечетким множествам из институтских учебников.

Первый период развития нечетких систем (конец 60-х – начало 70-х гг.) характеризуется развитием теоретического аппарата нечетких множеств. В 1970 году Беллман совместно с Заде разработали теорию принятия решений в нечетких условиях.

В 70-80 годы (второй период) появляются первые практические результаты в области нечеткого управления сложными техническими системами (парогенератор с нечетким управлением). И. Мамдани в 1975 году спроектировал первый функционирующий на основе алгебры Заде контроллер, управляющий паровой турбиной. Одновременно стало уделяться внимание вопросам создания экспертных систем, построенных на нечеткой логике, разработке нечетких контроллеров. Нечеткие экспертные системы для поддержки принятия решений нашли широкое применение в медицине и экономике.

Наконец, в третьем периоде, который длится с конца 80-х годов и продолжается в настоящее время, появляются пакеты программ для построения нечетких экспертных систем, а области применения нечеткой логики заметно расширяются. Она применяется в автомобильной,

аэрокосмической и транспортной промышленности, в области изделий бытовой техники, в сфере финансов, анализа и принятия управленческих решений и многих других. Кроме того, немалую роль в развитии нечеткой логики сыграло доказательство знаменитой теоремы FAT (FuzzyApproximationTheorem)
Б. Коско, в которой утверждалось, что любую математическую систему можно аппроксимировать системой на основе нечеткой логики.

Одним из самых впечатляющих результатов стало создание управляющего микропроцессора на основе нечеткой логики, способного автоматически решать известную задачу «о собаке, догоняющей кота». В 1990 году Комитет по контролю экспорта США внес нечеткую логику в список критически важных оборонных технологий, не подлежащих экспорту потенциальному противнику.

В бизнесе и финансах нечеткая логика получила признание после того, как в 1988 году экспертная система на основе нечетких правил для прогнозирования финансовых индикаторов единственная предсказала биржевой крах. И количество успешных фаззи-применений в настоящее время исчисляется тысячами.

В Японии это направление переживает настоящий бум. Здесь функционирует специально созданная организация LaboratoryforInternationalFuzzyEngineeringResearch. Программой этой организации является создание более близких человеку вычислительных устройств.

Информационные системы, базирующиеся на нечетких множествах и нечеткой логике, называют нечеткими системами.

Достоинства нечетких систем:

- функционирование в условиях неопределенности;
- оперирование качественными и количественными данными;
- использование экспертных знаний в управлении;
- построение моделей приближенных рассуждений человека;
- устойчивость при действии на систему всевозможных возмущений.

Недостатками нечетких систем являются:

- отсутствие стандартной методики конструирования нечетких систем;
- невозможность математического анализа нечетких систем существующими методами;

- применение нечеткого подхода по сравнению с вероятностным не приводит к повышению точности вычислений.

Теория нечетких множеств

Главное отличие теории нечетких множеств от классической теории четких множеств состоит в том, что если для четких множеств результатом вычисления характеристической функции могут быть только два значения – 0 или 1, то для нечетких множеств это количество бесконечно, но ограничено диапазоном от нуля до единицы.

Нечеткое множество

Пусть U– так называемое универсальное множество, из элементов которого образованы все остальные множества, рассматриваемые в данном классе задач, например множество всех целых чисел, множество всех гладких функций и т.д. Характеристическая функция множества $A \subseteq U$ – это функция μ_A, значения которой указывают, является ли $x \in U$ элементом множества A:

$$\mu_A(x) = \begin{cases} 1, & x \in A; \\ 0, & x \notin A. \end{cases}$$

В теории *нечетких множеств* характеристическая функция называется **функцией принадлежности**, а ее значение $\mu_A(x)$ –**степенью принадлежности** элемента x нечеткому множеству A.

Более строго:***нечетким множеством*** A называется совокупность пар

$$A = \left\{ \left\langle x, \mu_A(x) \right\rangle \middle| x \in U \right\},$$

где μ_A – функция принадлежности, то есть $\mu_A : U \to [0,1]$.

Пусть, например, U ={a, b, c, d, e}, $A = \left\{ \langle a,0 \rangle, \langle b,0.1 \rangle, \langle c,0.5 \rangle, \langle d,0.9 \rangle, \langle e,1 \rangle \right\}$. Тогда элемент a не принадлежит множеству A, элемент b принадлежит ему в малой степени, элемент c более или менее принадлежит, элемент d принадлежит в значительной степени, еявляется элементом множества A.

Пример. Пусть универсум U есть множество действительных чисел. Нечеткое множество A, обозначающее множество чисел, близких к 10, можно задать следующей функцией принадлежности (рис. 19):

$$\mu_A(x) = (1 + |x - 10|^m)^{-1},$$

где $m \in N$.

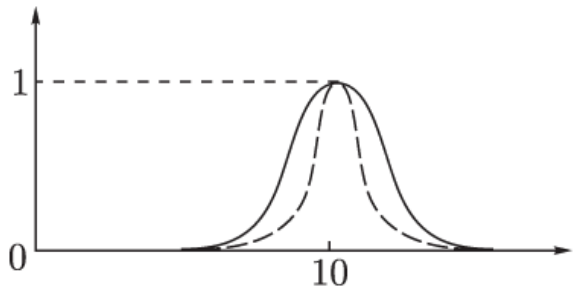

Рис. 19. **Функция принадлежности** μ_A

Показатель степени m выбирается в зависимости от степени близости к 10. Например, для описания множества чисел, очень близких к 10, можно положить m= 4, для множества чисел, не очень далеких от 10, m = 1.

Носителем нечеткого множества A называется четкое множество $\widetilde{A}$ таких точек в U, для которых величина $\mu_A(x)$ положительна, то есть $\widetilde{A} = \{x|\mu_A(x) > 0\}$.

Ядром нечеткого множества A называется четкое множество $\widetilde{A}$ таких точек в U, для которых величина $\mu_A(x) = 1$.

Множеством уровня α **(**α**-срезом)** нечеткого множества A называется четкое подмножество универсального множестваU, определяемое по формуле $A_\alpha = \{x|\mu_A(x) \geq \alpha\}$, где $\alpha \in [0,1]$.

Функцию принадлежности называют **нормальной**,если ядро нечеткого множества содержит хотя бы один элемент.

Операции над нечеткими множествами

Для нечетких множеств, как и для обычных, определены основные операции: объединение, пересечение и инверсия/дополнение.

Для определения пересечения и объединения нечетких множеств наибольшей популярностью пользуются следующие три группы операций:

Максиминные	$\mu_{A\cup B}(x) = \max\{\mu_A(x), \mu_B(x)\}$, $\mu_{A\cap B}(x) = \min\{\mu_A(x), \mu_B(x)\}$
Алгебраические	$\mu_{A\cup B}(x) = \mu_A(x) + \mu_B(x) - \mu_A(x)\cdot\mu_B(x)$, $\mu_{A\cap B}(x) = \mu_A(x)\cdot\mu_B(x)$

Ограниченные	$\mu_{A\cup B}(x) = \min\{1, \mu_A(x) + \mu_B(x)\}$, $\mu_{A\cap B}(x) = \max\{0, \mu_A(x) + \mu_B(x) - 1\}$

Дополнение нечеткого множества во всех трех случаях определяется одинаково:

$$\mu_{\overline{A}}(x) = 1 - \mu_A(x).$$

Пример. Пусть A – нечеткое множество «от 5 до 8»и B – нечеткое множество «около 4», заданные своими функциями принадлежности (рис. 20).

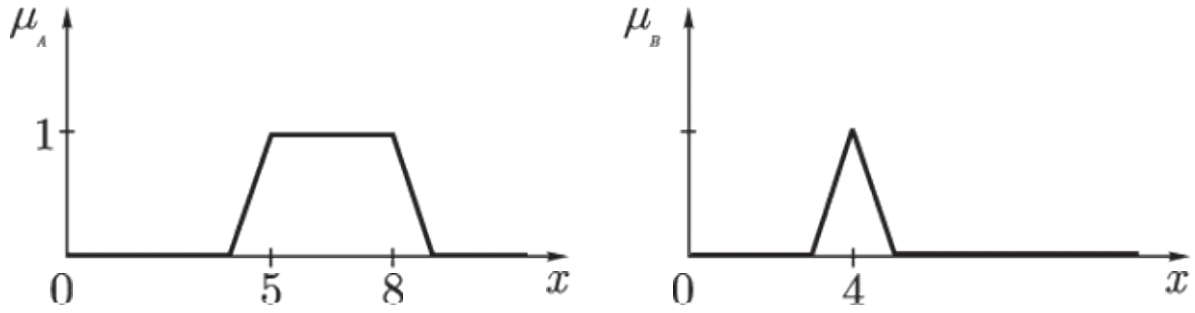

Рис. 20. **Функции принадлежности нечетких множеств A и B**

Тогда, используя максиминные операции, мы получим следующие множества, изображенные на рис. 21.

Рис. 21. **Функции принадлежности нечетких множеств, полученных из A и B**

При максиминном и алгебраическом определении операций не будут выполняться законы противоречия и исключения третьего:

$$A \cap \overline{A} \neq 0,\ A \cup \overline{A} \neq U,$$

а в случае ограниченных операций не будут выполняться свойства идемпотентности и дистрибутивности:

$$A \cup A \neq A,\ A \cap A \neq A,$$

$$A \cup (B \cap C) \neq (A \cap B) \cup (A \cap C),\ A \cap (B \cup C) \neq (A \cup B) \cap (A \cup C).$$

Можно показать, что при любом построении операций объединения и пересечения в теории нечетких множеств приходится отбрасывать либо

законы противоречия и исключения третьего, либо законы идемпотентности и дистрибутивности.

Нечеткая логика

Понятие нечеткой и лингвистической переменных используется при описании объектов и явлений с помощью нечетких множеств.

Нечеткая переменная характеризуется тройкой <α, X, A>, где α– наименование переменной, X – универсальное множество (область определения α), A – нечеткое множество на X, описывающее ограничения (то есть$\mu_A(x)$) на значения нечеткой переменной α.

Лингвистической переменной называется набор <β,T,X,G,M>, где β– наименование лингвистической переменной, T – множество ее значений (терм-множество), представляющих собой наименования нечетких переменных, областью определения каждой из которых является множество X (множество T называется базовым терм-множеством лингвистической переменной), G – синтаксическая процедура, позволяющая оперировать элементами терм-множества T, в частности генерировать новые термы (значения), M – семантическая процедура, позволяющая превратить каждое новое значение лингвистической переменной, образуемое процедурой G, в нечеткую переменную, то есть сформировать соответствующее нечеткое множество.

Лингвистическую переменную можно определить как переменную, значениями которой являются не числа, а слова или предложения естественного (или формального) языка. Например, лингвистическая переменная «возраст» может принимать следующие значения: «очень молодой», «молодой», «среднего возраста», «старый», «очень старый» и др. Ясно, что переменная «возраст» будет обычной переменной, если ее значения – точные числа; лингвистической она становится, будучи использованной в нечетких рассуждениях человека.

Каждому значению лингвистической переменной соответствует определенное нечеткое множество со своей функцией принадлежности. Так, лингвистическому значению «молодой» может соответствовать функция принадлежности, изображенная на рис.22.

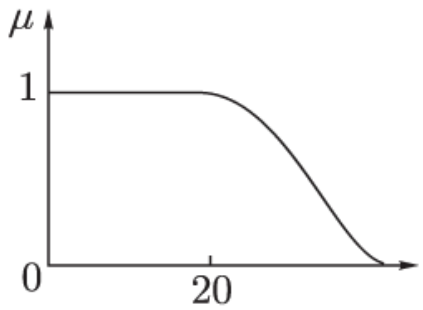

Рис. 22. **Функция принадлежности значения «молодой» лингвистической переменной «возраст»**

Пример.Пусть эксперт определяет толщину выпускаемого изделия с помощью понятий «*малая толщина*», «*средняя толщина*» и «*большая толщина*», при этом минимальная толщина равна 10 мм, а максимальная – 80 мм (рис. 23).

Формализация такого описания может быть проведена с помощью следующей лингвистической переменной <β, T, X, G, M>, где

β– толщина изделия;

T – {«*малая толщина*», «*средняя толщина*», «*большая толщина*»};

X = [10, 80];

G – процедура образования новых термов с помощью связок *и*, *или* и модификаторов типа *очень*, *не*, *слегка* и др. Например: «*малая или средняя толщина*» (рис. 24), *«очень малая толщина*» и др.;

M – процедура задания на X = [10, 80] нечетких подмножеств A_1= «*малая толщина*», A_2 = «*средняя толщина*», A_3= «*большая толщина*», а также нечетких множеств для термов из G(T) в соответствии с правилами трансляции нечетких связок и модификаторов *и*, *или*, *не*, *очень*, *слегка* и др.

Наряду с рассмотренными выше базовыми значениями лингвистической переменной «*толщина*» (T={«*малая толщина*», «*средняя толщина*», «*большая толщина*»}) возможны значения, зависящие от области определения X. В данном случае значения лингвистической переменной «толщина изделия» могут быть определены как «*около 20 мм*», «*около 50 мм*», «*около 70 мм*», то есть в виде *нечетких чисел*.

Рис. 23. **Функции принадлежности нечетких множеств:**
«*малая толщина*» = A_1, «*средняя толщина*» = A_2, «*большая толщина*» = A_3

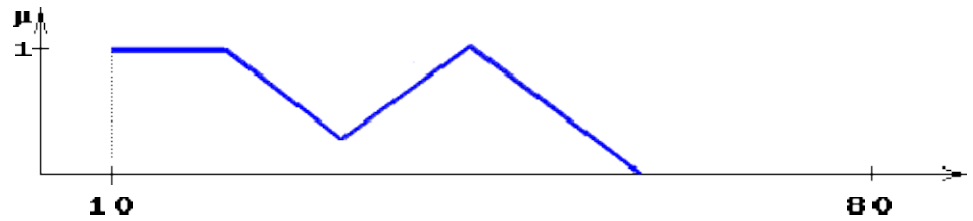

Рис. 24. **Функция принадлежности**
нечеткого множества «*малая или средняя толщина*» = $A_1 \cup A_1$

Нечеткими высказываниями будем называть высказывания следующего вида:

1. Высказывание <β есть β'>, где β– наименование лингвистической переменной, β' – ее значение, которому соответствует нечеткое множество на универсальном множестве X.

Например, высказывание <*давление большое*> предполагает, что лингвистической переменной «*давление*» придается значение «*большое*», для которого на универсальном множестве X переменной «*давление*» определено соответствующее данному значению «*большое*» нечеткое множество.

2. Высказывание <β есть mβ'>, где m – модификатор, которому соответствуют слова «очень», «более или менее», «много больше» и др.

Например:<*давление очень большое*>, <*скорость много больше средней*> и др.

3. Составные высказывания, образованные из высказываний видов 1 и 2 и союзов *и;или;если…то...; если… то… иначе...*

Основным правилом вывода в традиционной логике является правило modusponens, согласно которому мы судим об истинности высказывания B по истинности высказываний A и $A \rightarrow B$.

Например, если A– высказывание <Иван в больнице>, B– высказывание <Иван болен>, то если истинны высказывания <Иван в

больнице> и <Если Иван в больнице, то он болен>, то истинно и высказывание <Иван болен>.

Во многих привычных рассуждениях, однако, правило modusponens используется не в точной, а в приближенной форме. Так, обычно мы знаем, что A истинно и что $A^* \to B$, где A^* есть в некотором смысле приближение A. Тогда из $A^* \to B$ мы можем сделать вывод о том, что B приближенно истинно.

Нечеткая импликация выражается в следующем виде:

$$A \to B = \overline{A \cup B} \text{ и } \mu_{A \to B}(x) = \max\{1 - \mu_A(x), \mu_B(x)\}.$$

Основой для проведения операции нечеткого логического вывода является база правил, содержащая нечеткие высказывания в форме *если…то…*и функции принадлежности для соответствующих лингвистических термов. При этом должны соблюдаться следующие условия:

- существует хотя бы одно правило для каждого лингвистического терма выходной переменной;
- для любого терма входной переменной имеется хотя бы одно правило, в котором этот терм используется в качестве предпосылки (левая часть правила).

В противном случае имеет место неполная база нечетких правил.

Для реализации логического вывода необходимо выполнить следующее:

1. Сопоставить факты с каждым из правил и определить степень соответствия, назначив текущую силой правил.

2. Для каждого правила, сила которого больше заданного порога, вычислить достоверность левой части.

3. Для каждого правила с помощью оператора импликации вычислить достоверность правой части.

4. Для многих результатов, полученных по различным правилам, выбрать одно (усредненное).

Пример. Пусть есть некоторая система, например реактор, описываемая тремя параметрами: *температура, давление* и *расход рабочего вещества*. Все показатели измеримы, и множество возможных значений известно. Также из опыта работы с системой известны некоторые

правила, связывающие значения этих параметров. Предположим, что сломался датчик, измеряющий значение одного из параметров системы, но знать его показания необходимо хотя бы приблизительно. Тогда встает задача об отыскании этого неизвестного значения (пусть это будет *давление*) при известных показателях двух других параметров (*температуры* и *расхода*) и связи этих величин в виде следующих правил:

- если *Температура* низкая и *Расход* малый, то *Давление* низкое;
- если *Температура* средняя, то *Давление* среднее;
- если *Температура* высокая или *Расход* большой, то *Давление* высокое.

Температура, Давление и *Расход*– лингвистические переменные.

Температура. Универсум (множество возможных значений) – отрезок [0, 150]. Начальное множество термов {*Высокая, Средняя, Низкая*}. Функции принадлежности термов имеют следующий вид (рис. 25):

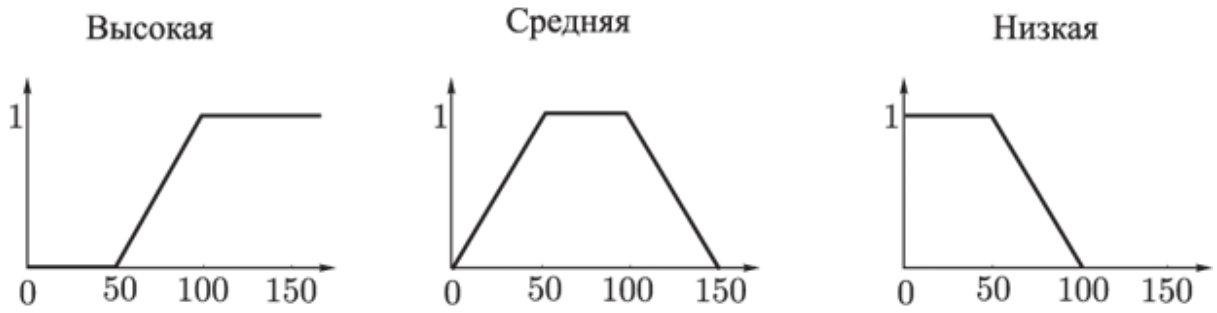

Рис. 25. **Функции принадлежности термов лингвистической переменной *Температура***

Давление. Универсум – отрезок [0, 100]. Начальное множество термов {*Высокое, Среднее, Низкое*}. Функции принадлежности термов имеют следующий вид (рис. 26):

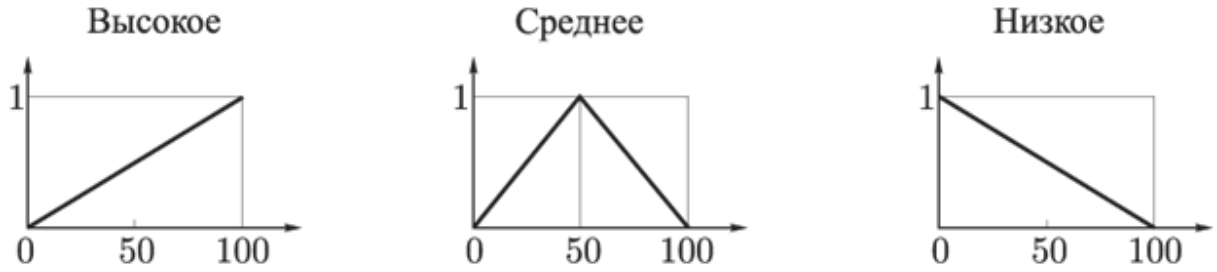

Рис. 26. **Функции принадлежности термов лингвистической переменной *Давление***

Расход. Универсум – отрезок [0, 8]. Начальное множество термов {*Большой, Средний, Малый*}. Функции принадлежности термов имеют следующий вид (рис. 27):

Рис. 27. **Функции принадлежности термов лингвистической переменной *Расход***

Пусть известны значения:*Температура* – 85 и *Расход* – 3.5. Произведем расчет значения давления.

Этап фаззификации (переход от заданных четких значений к степеням уверенности). По заданным значениям входных параметров найдем степени уверенности простейших утверждений:

- *Температура Высокая – 0.7;*
- *Температура Средняя – 1;*
- *Температура Низкая – 0.3;*
- *Расход Большой – 0;*
- *Расход Средний – 0.75;*
- *Расход Малый – 0.25.*

Затем вычислим степени уверенности посылок правил:

- *Температура Низкая и Расход Малый: min(Темп. Низкая, Расход Малый)= min(0.3; 0.25)=0.25;*
- *Температура Средняя: 1;*
- *Температура Высокая или Расход Большой: max(Темп. Высокая, Расход Большой)= max(0.7; 0)=0.7.*

Каждое из правил представляет собой нечеткую импликацию. Степень уверенности посылки мы вычислили, а степень уверенности заключения задается функцией принадлежности соответствующего терма. Поэтому, используя один из способов построения нечеткой импликации, мы получим новую нечеткую переменную, соответствующую степени уверенности в значении выходных данных при применении к заданным

входным соответствующего правила. Используя определение нечеткой импликации как минимума левой и правой частей (определение Мамдани), имеем (рис. 28):

Рис. 28. **Результат этапа фаззификации**

Этап аккумуляции (объединение результатов применения всех правил). Один из основных способов аккумуляции – построение максимума полученных функций принадлежности (объединение функций принадлежности, полученных на этапе фаззификации).

Полученную функцию принадлежности уже можно считать результатом. Это новый терм выходной переменной *Давление*. Его функция принадлежности говорит о степени уверенности в значении давления при заданных значениях входных параметров и использовании правил, определяющих соотношение входных и выходных переменных. Но обычно все-таки необходимо какое-то конкретное числовое значение (рис. 29).

Этап дефаззификации (получение конкретного значения из универса по заданной на нем функции принадлежности). Существует множество методов дефаззификации, но в этом случае достаточно метода первого максимума. Применяя его к полученной функции принадлежности, получаем, что значение давления – 50. Таким образом, если мы знаем, что температура равна 85, а расход рабочего вещества – 3.5, то можем сделать вывод, что давление в реакторе равно примерно 50.

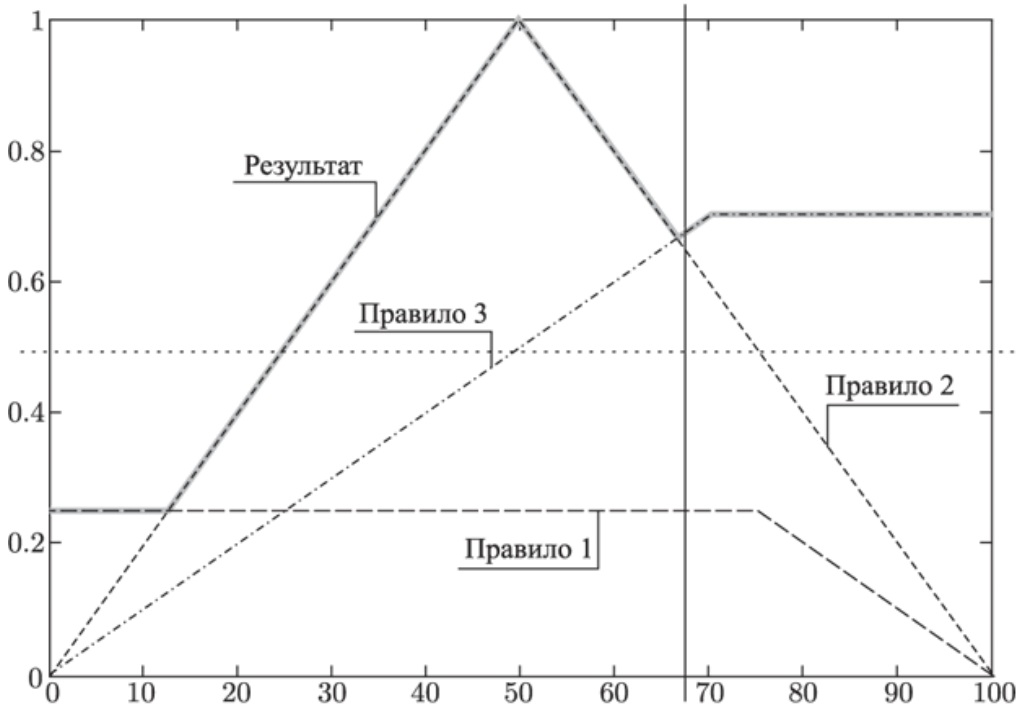

Рис. 29. **Результат этапа аккумуляции**

Часть 3. ИНТЕЛЛЕКТУАЛЬНЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ

§1. Экспертные системы

На начальных этапах развития искусственный интеллект подвергался жесткой критике, и одним из доводов был тот, что практической пользы от него нет, он занимается игрушками. Экспертные системы одними из первых доказали практическую пользу этого научного направления, принося в начале 80-х годов XX века коммерческую прибыль своим создателям.

Термин «системы, основанные на знаниях» (knowledge-basedsystems) появился в 1976 году одновременно с первыми системами, аккумулирующими опыт и знания экспертов.

Экспертные системы – это прикладные системы искусственного интеллекта, в которых база знаний представляет собой формализованные эмпирические знания высококвалифицированных специалистов (экспертов) в какой-либо узкой предметной области, они аккумулируют эти знания и тиражируют их для консультации менее квалифицированных специалистов.

В 1965 году Э. Фейгенбаум (бывший студент Герберта Саймона), Б. Бьюкенен (философ по образованию) и Д. Ледерберг (лауреат Нобелевской премии в области генетики) начали работы над первой экспертной системой DENDRAL. В 1969 году В. Мартином и Д. Мозесом была создана математическая экспертная система MACSYMA. Первая экспертная система для медицинской диагностики была создана в 1973 году Э. Шортлиффом и называлась MYCIN, она повлекла за собой разработку первого командного интерпретатора EMYCIN (В. Милле, Шортлифф и Бьюкенен). В 1976 году Дуда и Харт начали работу над экспертной системой PROSPECTOR, предназначенной для разведки полезных ископаемых, в 1984 году система точно предсказала существование месторождения молибдена, оцененного в многомиллионную сумму.

Эти экспертные системы, разработанные в 60-70-х годах стали классическими. По происхождению, предметным областям и по преемственности применяемых идей, методов и инструментальных программных средств их можно разделить на несколько семейств.

1. **META-DENDRAL.**Система DENDRAL позволяет определить наиболее вероятную структуру химического соединения по экспериментальным данным (масс-спектрографии, данным ядерного магнитного резонанса и др.).Meta-DENDRAL автоматизирует процесс приобретения знаний для DENDRAL. Она генерирует правила построения фрагментов химических структур.

2. **MYCIN-EMYCIN-TEIREIAS-PUFF-NEOMYCIN.**Это семейство медицинских ЭС и сервисных программных средств для их построения.

3. **PROSPECTOR-KAS.** СистемаPROSPECTOR предназначена для поиска (предсказания) месторождений на основе геологических анализов. KAS – система приобретения знаний для PROSPECTOR.

4. **CASNET-EXPERT.** Система CASNET – медицинская ЭС для диагностики выдачи рекомендаций по лечению глазных заболеваний. На ее основе разработан язык инженерии знаний EXPERT, с помощью которого создан ряд других медицинских диагностических систем.

5. **HEARSAY-HEARSAY-2-HEARSAY-3-AGE**. Первые две системы этого ряда являются развитием интеллектуальной системы распознавания слитной человеческой речи, слова которой берутся из заданного словаря. Эти системы отличаются оригинальной структурой, основанной на использовании доски объявлений – глобальной базы данных, содержащей текущие результаты работы системы. В дальнейшем на основе этих систем были созданы инструментальные системы HEARSAY-3 и AGE (AttempttoGeneralize – попытка общения) для построения экспертных систем.

6. Системы **AM** (ArtificalMathematician– искусственный математик) и **EURISCO** были разработаны в Станфордском университете доктором Дугласом Ленатом для исследовательских и учебных целей. В систему AM первоначально было заложено около 100 правил вывода и более 200 эвристических алгоритмов обучения, позволяющих строить произвольные математические теории и представления. EURISCO – это развитие системы AM, с ее помощью в военно-стратегической игре,

проводимой ВМФ США, была разработана стратегия, содержащая ряд оригинальных тактических ходов.

Кроме разработки самих экспертных систем исследователи занялись созданием инструментального средства для экспертных систем: в 1983 году компания IntelliCorp создала KEE, а в 1985 году агентство NASA выпустило первую версию CLIPS.

Экспертные системы быстро завоевали позиции на информационном рынке и получили широкое распространение. Уже в 1987 году опрос пользователей, проведенный журналом IntelligentTechnologies (США), показал, что примерно:

- 25% пользователей используют ЭС;
- 25% собираются приобрести ЭС в ближайшие 2-3 года;
- 50% предпочитают провести исследование об эффективности их использования.

В России в исследования и разработку ЭС большой вклад внесли работы Д. А. Поспелова (основателя Российской ассоциации искусственного интеллекта и его первого президента), Э. В. Попова, В. Ф. Хорошевского, В. Л. Стефанюка, Г. С. Осипова, В. К. Финна, В. Л. Вагина, В. И. Городецкого и многих других.

Экспертные системы 60-90-х годов являются первым поколением экспертных систем, для них характерно:

1) знаниями системы являются только знания эксперта, накопление знаний не предусматривается;

2) методы представления знаний позволяют описывать лишь статические предметные области;

3) модели представления знаний ориентированы на простые предметные области.

Развиваясь, экспертные системы вышли за эти рамки. Принципы представления знаний в экспертных системах второго поколенияизменились:

1) используются не поверхностные знания, а более глубинные;

2) для представления знаний привлекаются средства и методы других направлений искусственного интеллекта, например нейронных сетей;

3) системы имеют динамические базы знаний.

Появление Интернета не могло не повлиять на развитие экспертных систем. Возможность получать знания через сеть и извлекать знания из сети не могла не быть использована разработчиками. Поэтому сейчас развиваются распределенные и web-ориентированные экспертные системы.

Сейчас количество экспертных систем исчисляется тысячами и десятками тысяч. В развитых зарубежных странах сотни фирм занимаются их разработкой и внедрением в различные сферы жизни.

В качестве современных ЭС можно назвать быстродействующую систему OMEGAMON (фирма Candle, с 2004 года – IBM) для отслеживания состояния корпоративной информационной сети и G2 (фирма Gensym) – коммерческую экспертную систему для работы с динамическими объектами.

Экспертные системы используют в тех случаях, когда недостаточно экспертов, в опасных (вредных) для них условиях, в процессе обучения. Экспертные системы решают задачи, для которых отсутствуют четкие алгоритмы решения.

Модель экспертных систем

Экспертные системы работают в диалоговом режиме (отвечают на поставленные пользователем вопросы), при этом они должны уметь объяснять, откуда получено то или иное решение. Любая экспертная система содержит как минимум пять компонентов или подсистем (рис. 30).

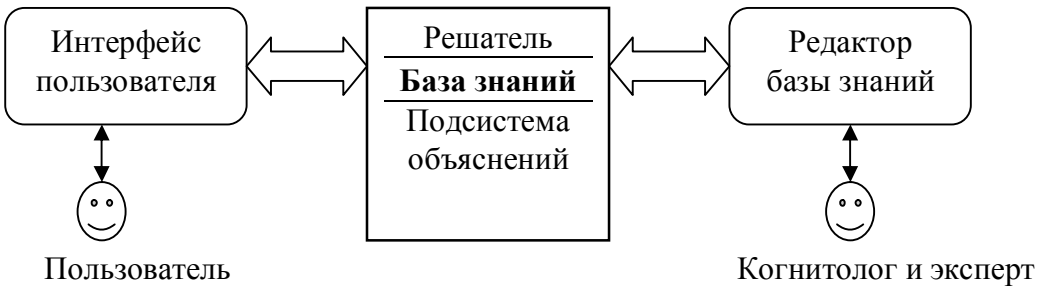

Рис. 30. **Базовая структура экспертной системы**

Пользователь экспертной системы –специалист предметной области, для которого предназначена система. Обычно его квалификация недостаточно высока, и поэтому он нуждается в помощи и информационной поддержке своей деятельности.

Инженер по знаниям – специалист в области искусственного интеллекта, работающий с экспертами и формирующий базу знаний. Синонимы: *когнитолог, инженер-интерпретатор, аналитик.*

Интерфейспользователя – комплекс программ, реализующих диалог пользователя с ЭС как на стадии ввода информации, так и при получении результатов.

База знаний – ядро экспертной системы, совокупность знаний предметной области, записанная на машинный носитель в форме, понятной эксперту и пользователю (обычно на некотором языке, приближенном к естественному).

Решатель –программа, моделирующая ход рассуждений эксперта на основании знаний, имеющихся в базе знаний. Синонимы: *дедуктивная машина, машина вывода, блок логического вывода.*

Подсистема объяснений –программа, позволяющая пользователю получить ответы на вопросы: «Как была получена та или иная рекомендация?» и «Почему система приняла такое решение?» Ответ на вопрос «как» –это трассировка всего процесса получения решения с указанием использованных фрагментов базы знаний, то есть всех шагов цепи умозаключений. Ответ на вопрос «почему» – ссылка на умозаключение, непосредственно предшествовавшее полученному решению, то есть отход на один шаг назад. Развитые подсистемы объяснений поддерживают и другие типы вопросов.

Интеллектуальный редактор базы знаний –программа, представляющая инженеру по знаниям возможность создавать базу знаний в диалоговом режиме. Включает в себя систему вложенных меню, шаблонов языка представления знаний, подсказок и других сервисных средств, облегчающих работу с базой.

Описанная структура является базовой и может расширяться (рис. 31).

Обозначенные штриховой линией подсистемы моделирования внешнего мира, интерфейс с внешним миром, система датчиков необходимы для экспертных систем реального времени для получения

данных и их интерпретации. Экспертные системы могут накапливать опыт в виде прецедентов (уже разрешенных ситуаций), которые сохраняются в базе знаний и используются в дальнейшем.

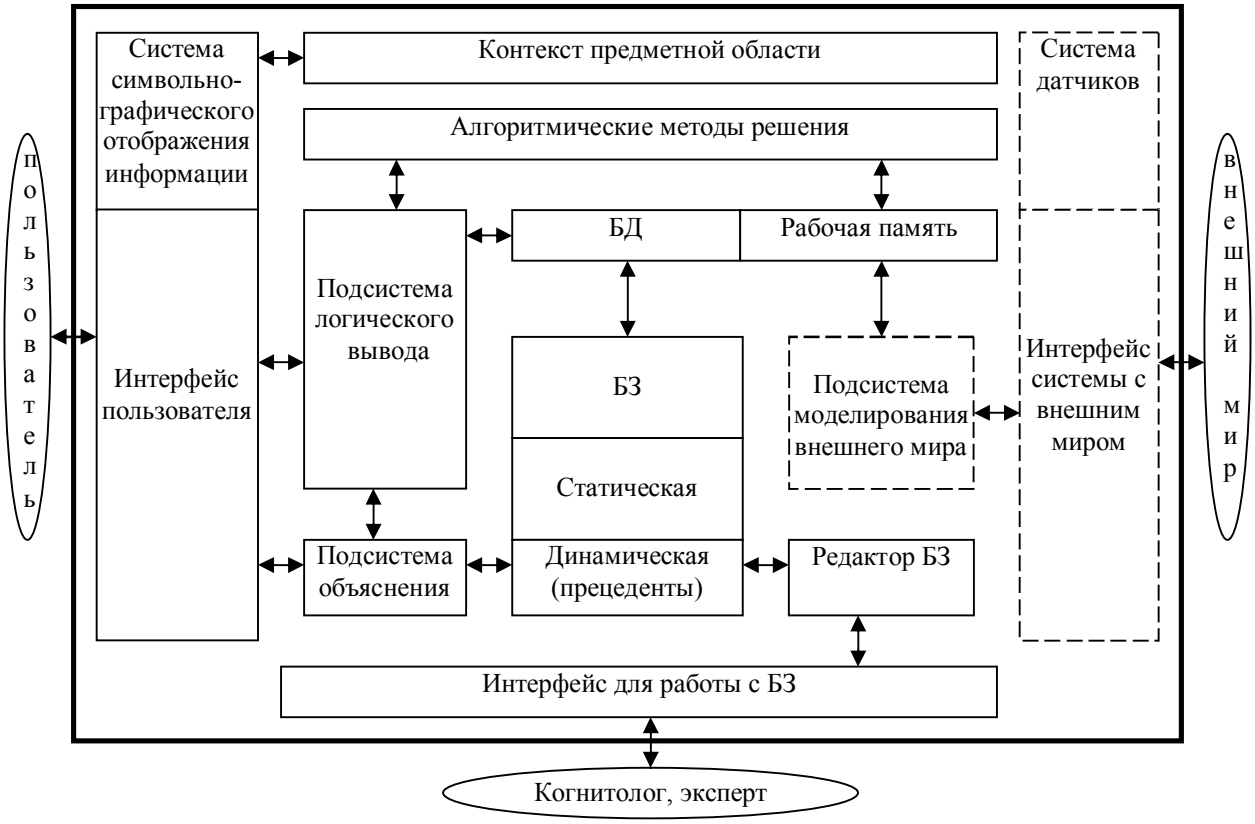

Рис. 31. **Структура экспертной системы**

Блок алгоритмических методов решения включает в себя все вычислительные операции и алгоритмы, реализуемые методами традиционного программирования, интегрированные в экспертную систему. Объединение в рамках экспертной системы методов традиционного программирования и искусственного интеллекта позволяет значительно повысить эффективность и качество принимаемых решений.

Специфика предметной области, для которой строится система, отображается и описывается не только в базе знаний, но и в подсистеме «Контекст предметной области», которая позволяет более наглядно представить входную и выходную информацию в принятом для конкретной предметной области виде.

Современные информационные системы часто используют архитектуру «клиент-сервер», позволяющую строить распределенные и

сетевые приложения. При клиент-серверной архитектуре экспертной системы на стороне клиента находятся интерфейсы пользователя, эксперта, внешней среды и система датчиков. Остальные блоки располагаются на серверной стороне.

Классификация экспертных систем и оболочек экспертных систем

Существующее множество экспертных систем делится на несколько классов (рис.32) по различным критериям.

По назначению выделяют системы общего назначения, которые претендуют на универсальность в решении задач (CASNET), и специализированные, решающие конкретную задачу (1-st Clas, Элис) или ориентированные на определенную предметную область (MYCIN, MACSYMA, МОДИС, ДИАГЕН, INTERNIST-I).

По критерию *взаимодействия с внешней средой* различают статические системы, в которых есть только интерфейс пользователя, а механизм взаимодействия с внешним миром, например через датчики, отсутствует, и динамические, которые с помощью встроенных интерфейсов получают информацию с внешних датчиков или других устройств. Квазидинамические экспертные системы получают информацию об изменении ситуации во внешней среде через заданный промежуток времени (больше нескольких секунд).

Экспертные системы разрабатываются для различных ЭВМ и различаются по *аппаратно-программной платформе*. Они разрабатываются и эксплуатируются на персональных (PROSPECTOR), символьных (Picon), мини- (СПЭИС) и суперкомпьютерах (ЭКСПЕРТИЗА).

По *степени интеграции* экспертные системы могут быть автономными программными комплексами (ДАМП), которые работают самостоятельно, или частью более общей системы (интегрированные системы), а также звеном в цепочке программ, обрабатывающих информацию с общей целью, например управления предприятием.

В зависимости от *размера базы знаний* выделяют простые экспертные системы – до 1000 простых правил (GUIDON, Плотина), средние – от 1000 до 10000 структурированных правил (XCON, GOSSEYN, ДИАГЕН) и сложные – более 10000 структурированных правил.

Экспертные системы различаются *по стадии существования*, то есть по степени завершенности системы. Первая стадия существования экспертной системы – это исследовательский образец. Он разрабатывается 3-6 месяцев с минимальной базой знаний (SYSTEM-D, SYN). Вторая – демонстрационный образец – разрабатывается 6-12 месяцев (THYROID MODEL); третья – промышленный образец – разрабатывается 1-1,5 года с полной базой знаний (PUFF, FOSSIL) и последняя – коммерческий образец – разрабатывается 1,5-3 года с полной базой знаний (KNEECAP, MACSYMA).

Рис. 32. **Классификация экспертных систем**

Средства разработки экспертных систем

Существующие средства разработки экспертных систем можно разделить на 3 класса (рис. 33). *Традиционные языки программирования* (C++, Java, Delphi) позволяют построить экспертные системы «с нуля» для конкретной задачи или предметной области, обеспечив хорошие показатели качества и необходимую функциональность системы, но на разработку требуются значительные временные и финансовые ресурсы. Так создают экспертные системы любой стадии существования, в особенности коммерческие системы, продажа которых возместит затраты.

Рис. 33. **Классификация инструментальных средств разработки ЭС**

Языки искусственного интеллекта (LISP, PROLOG, Рефал) были разработаны специально для представления знаний. Построение с их помощью экспертных систем позволяет более легко оперировать экспертными знаниями, но ограничивает способ их представления структурой языка. С помощью языков искусственного интеллекта создаются исследовательские и демонстрационные образцы.

Следующий класс средств построения экспертных систем – специальный программный инструментарий – ориентирован только на создание интеллектуальных информационных систем и делится на два подкласса: оболочки и среды разработки интеллектуальных систем.

Среды разработки являются программными комплексами, позволяющими строить системы из отдельных готовых блоков. На их основе создаются демонстрационные и промышленные образцы экспертных систем.

Оболочка экспертных систем– инструментальное средство для проектирования и создания экспертных систем. В состав оболочки входят средства проектирования базы знаний с различными формами представления знаний и выбора режима работы решателя задач. Для конкретной предметной области инженер по знаниям определяет нужное представление знаний и

стратегии решения задач, а затем, вводя их в оболочку, создает конкретную экспертную систему.

Применение оболочки позволяет достаточно быстро и с минимальными затратами создать исследовательскую, демонстрационную или промышленную экспертную систему. Оболочки можно классифицировать следующим образом (рис.34). По степени отработанности выделяют экспериментальную (GPSI), исследовательскую (Expert) и коммерческую (EXSYS) оболочки.

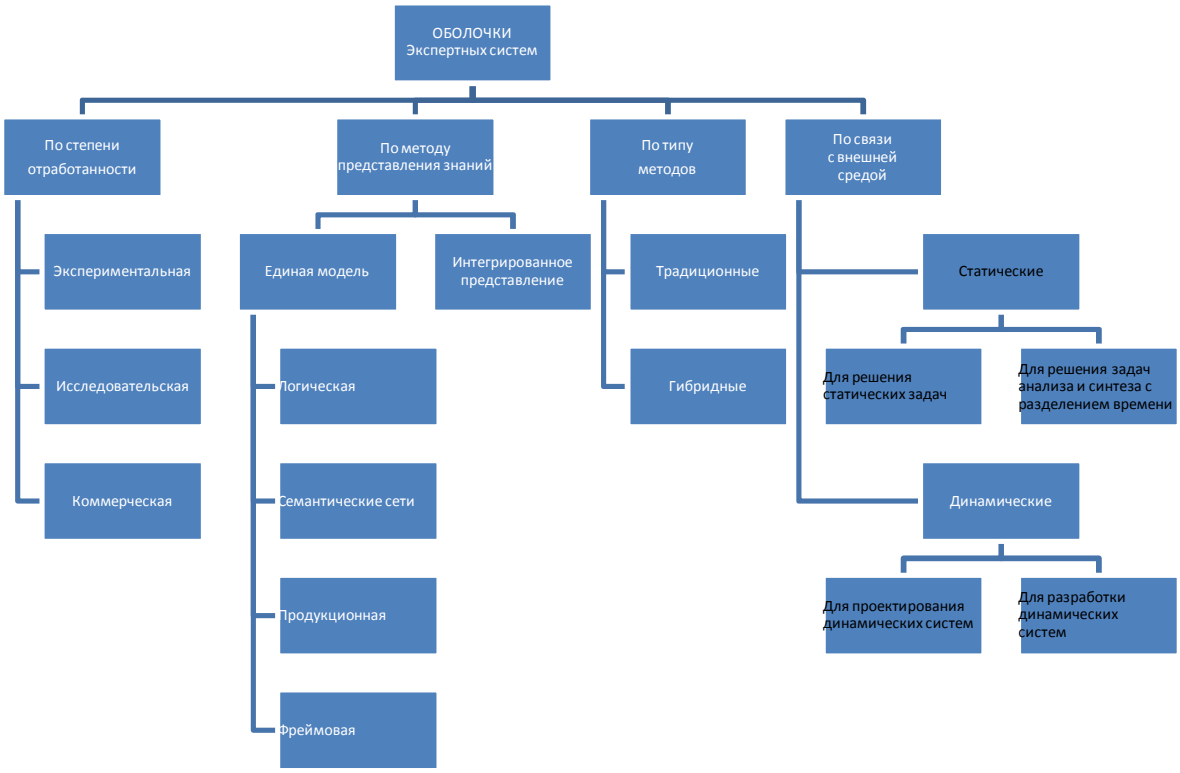

Рис. 34. **Классификация оболочек экспертных систем**

Знания в базе могут быть представлены одним способом (EMYCIN, CLIPS) – семантической сетью, продукциями, фреймами и т.д. или же несколькими (MINEVRA, EsWin) для создания более полной, гибкой и наглядной модели предметной области.

Используемые в оболочке методы могут быть традиционными (CubiCalc, NEXPERT, Алеф) – алгоритмы, деревья вывода и т.д. и гибридными (FuzzyCLIPS, MultiNeuron), где совместно с традиционными используются нейронные сети, нечеткая логика и т.д.

Существуют статические оболочки, предназначенные для решения статических задач (1-st Clas, Элис). Они характеризуются использованием

поверхностной технологии, общих правил и поиска решения от цели к данным, применяются для решения задач анализа.

Статические оболочки, предназначенные для решения задач анализа и синтеза с разделением времени (KAPPA, Clips), используют глубинный и структурный подходы, осуществляют поиск решений от цели к данным и от данных к цели.

Оболочки для проектирования динамических систем (Framework, NExpert) применяют поверхностный подход, принимают решения на основе правил общего вида.

Оболочки для разработки динамических систем (G2, Rethink, RkWorks) имеют подсистему моделирования, планировщика решений, используют смешанную технологию, правила общего и частного вида, решение задачи анализа и синтеза в реальном времени.

EMYCIN – первая оболочка, основанная на MYCIN. Принципы, разработанные для PROSPECTOR, были использованы при создании таких систем, как KAS, SAGE, SAVOIR.

Изменение принципов построения ведет к развитию инструментария. Поэтому оболочки прошли тот же эволюционный путь, что и ЭС. Современные оболочки предлагают следующие возможности (в каждой конкретной оболочке представлены частично):

- гибридное представление знаний (EsWin);
- выбор из нескольких стратегий вывода (G2, CLIPS);
- подключение библиотек и других систем (ACTIVATION FRAMEWORK);
- архитектура на основе «доски объявлений» (HEARSAY-III);
- архитектура «клиент-сервер» (JESS);
- интеграция в Интернет / Интранет (Egg2Lite, ExsysCorvid);
- графический интерфейс (WindExS,WxCLIPS);
- подсистема моделирования (G2);
- модульное построение системы (ReThink, G2);
- визуализация структуры БЗ (W.E.S.T.) и т.д.

§2. Системы поддержки принятия решений

Системы поддержки принятия решений – это программные средства и информационно-аналитические технологии, предназначенные специально для оказания помощи в решении задач поиска, анализа и выбора лучших из

возможных вариантов. При этом лицо, принимающее решение, должно обеспечиваться не только информационной, но в первую очередь технологической поддержкой процедуры, вплоть до выбора лучшего решения.

Понятие о поддержке в принятии решений сформулировали П. Кин и Ч. Стэйбел. Ранние определения систем поддержки принятия решений (в начале 70-х годов прошлого века) отражали следующие три свойства систем:

- возможность оперировать с неструктурированными или слабоструктурированными задачами, в отличие от задач, с которыми имеет дело исследование операций;
- интерактивные автоматизированные (то есть реализованные на базе компьютера) системы;
- разделение данных и моделей.

В современном представлении идеальная система поддержки принятия решений:

- оперирует со слабоструктурированными решениями;
- предназначена для лица, принимающего решения, различного уровня;
- может быть адаптирована для группового и индивидуального использования;
- поддерживает как взаимозависимые, так и последовательные решения;
- поддерживает 3 фазы процесса решения: интеллектуальную часть, проектирование и выбор;
- поддерживает разнообразные стили и методы решения, что может быть полезно при решении задачи группой лиц, принимающих решения;
- является гибкой и адаптируется к изменениям как организации, так и ее окружения;
- проста в использовании и модификации;
- улучшает эффективность процесса принятия решений;
- позволяет человеку управлять процессом принятия решений с помощью компьютера, а не наоборот;
- поддерживает эволюционное использование и легко адаптируется к изменяющимся требованиям;
- может быть легко построена, если возможно сформулировать логику конструкции системы поддержки принятия решений;
- поддерживает моделирование;

- позволяет использовать знания.

До середины 60-х годов XX века создание больших информационных систем было чрезвычайно дорогостоящим, поэтому первые информационные системы менеджмента (ManagementInformationSystems) были созданы в эти годы лишь в достаточно больших компаниях. Они предназначались для подготовки периодических структурированных отчетов для менеджеров. Но информационные системы способны набольшее. В конце 60-х годов появляется новый тип информационных систем – модель-ориентированные системы поддержки принятия решений (DecisionSupportSystems) или системы управленческих решений (ManagementDecisionSystems).

70-е годы XX века стали периодом возникновения ранних систем поддержки принятия решений и теоретических изысканий. К 1975 году Дж. Д. Литтл разработал систему Brandaid (Поддержка бренда), которая предназначалась для поддержки принятия решений в производстве, продвижении, ценообразовании и рекламе. Также Литтл в своей более ранней статье сформулировал критерии по формированию моделей и систем для поддержки принятия решений для менеджмента: *надежность, легкость контроля, простота и полнота набора необходимых деталей.*

В начале 80-х годов исследователи из академических институтов создали новую категорию программного обеспечения для поддержки группового принятия решений. Самыми ранними вариантами таких систем были Mindsight компании ExecucomSystems, GroupSystems, созданные в Аризонском университете, и система SAMM, созданная исследователями Университета Миннесоты. В 1984 году в Университете Аризоны была завершена разработка системы PLEXSYS и сформирована служба компьютеризованной поддержки групповых решений.

Новые технологии стали использовать совместно с системами поддержки принятия решений. Примерно с 1990-го года Б. Инмон и Р. Кимбел начали продвигать системы поддержки принятия решений, построенные с помощью технологий реляционных баз данных. Активно разрабатывались системы поддержки принятия решений, основанные на архитектуре «клиент-сервер», до этого в основном использовались большие компьютеры (mainframe). Системы поддержки принятия решений стали интегрироваться в долговременные хранилища. Системы поддержки принятия решений, использующие возможности Интернета, стали реальностью около 1995 года.

Структура систем поддержки принятия решений

Структура системы поддержки принятия решений зависит от решаемой задачи, предметной области, аппаратно-программной платформы и конкретной реализации. В самом общем виде систему поддержки принятия решения можно представить в виде двух подсистем:*системы поддержки генерации решений* и *системы поддержки выбора решений* (рис. 35).

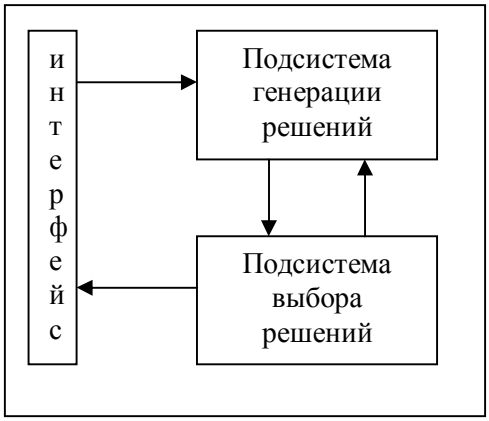

Рис. 35. **Обобщенная структура системы поддержки принятия решений**

Системы поддержки генерации решений можно разделить на *эвристические* и *оптимизационные*. Эвристические технологии стимулируют и дисциплинируют мышление (например, структурный и морфологический анализ), помогают находить варианты решений на базе известных правил, принципов и аналогов. Однако при формировании вариантов решений уникальных задач (например, при стратегическом планировании) их применимость часто ограничивают вспомогательными функциями. Оптимизационные системы поддержки принятия решений основаны на методах оптимального структурного синтеза и параметрической оптимизации.

Системы поддержки выбора решений предназначены для выбора эффективных вариантов решения, сгенерированных любым из вышеперечисленных методов либо поступивших извне (например, заявок на финансирование инвестиционных проектов). Эти системы базируются на методах многокритериального анализа и экспертных оценок.

Другой вариант обобщенной архитектуры системы поддержки принятия решений состоит из пяти частей (рис. 36): источники данных (часто используется база данных), система управления данными (если источников несколько, подсистема объединяет, проверяет и синхронизирует их), модели управления (включают в себя модели решаемой задачи и внешнего мира),

машина вывода (позволяет с помощью имеющихся данных и моделей получить и обосновать решение) и интерфейс пользователя.

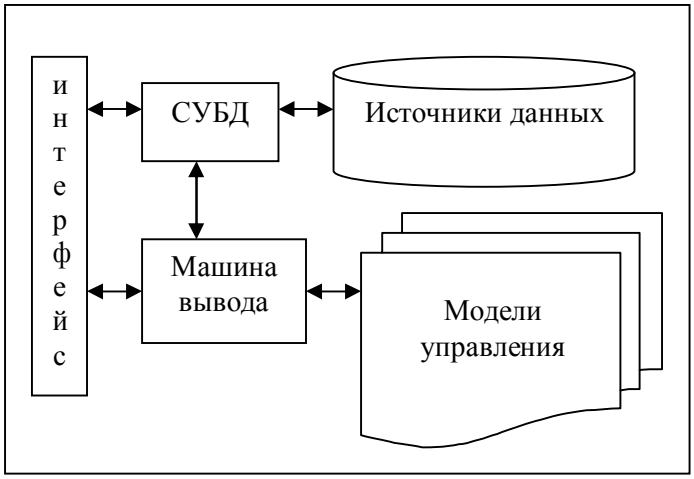

Рис. 36. **Компоненты системы поддержки принятия решений**

Систему поддержки принятия решений можно представить в виде процессов (рис. 37).

Система проводит сбор запрашиваемых у пользователя или внешних датчиков данных и вложенных в нее при создании данных и знаний. После этого определяет состояние, в котором находится система и решаемая задача, критерии и цели (может запрашивать и уточнять у пользователя). На основе полученных данных, которые содержатся в памяти, и имеющейся модели системы или задачи с учетом сформированных критериев и целей генерируется множество решений, которые проверяются на модели, и выбирается лучшее. После реализации решения производится оценка результатов: если она неудовлетворительная, то процессы генерации и выбора повторяются с учетом новых данных.

Рис. 37. **Процессы системы поддержки принятия решений**

С информационно-аналитической точки зрения, задачей системы поддержки принятия решений является агрегирование (сжатие) многокритериальной информации об анализируемых объектах до объема и формы представления, воспринимаемых лицом, принимающим решение.

С программно-технологической точки зрения, варианты решений являются для системы принятия решений просто анализируемыми объектами, которые характеризуются наборами количественных и качественных характеристик (показателей).

Чаще всего системы принятия решений используют при стратегическом планировании и выборе организации сложных систем. Несмотря на уникальность каждой из таких задач, при их решении используется типовая технология обработки информации. Поэтому в мире уже достаточно широко используются универсальные системы поддержки принятия решений, предназначенные для сравнения и выбора вариантов любых решений. Задача пользователя таких систем заключается в настройке универсальной программной оболочки на нужную предметную область путем ввода (импорта) информации об анализируемых объектах, а также иерархии требований и предпочтений. Для универсальных систем принятия решений анализируемыми объектами могут являться любые объекты, для которых требуется дать оценку их соответствия предъявляемым требованиям по многим критериям, принять решение альтернативного выбора, например

«выбрать лучшее из...» или «одобрить-отвергнуть», принять решение о распределении ресурсов среди группы объектов, исходя из их текущей приоритетности.

В зависимости от решаемой задачи в системах поддержки принятия решений могут использоваться различные методы принятия решений, привлекаться модели и методы, разработанные в рамках предметной области. Примерами методов принятия решения являются:

- декомпозиция главной цели до того уровня детализации, когда для нижнего уровня иерархии целей можно сформулировать критерии, позволяющие адекватно описать степень достижения целей при принятии той или иной альтернативы;
- метод аналитических иерархических процессов (лицо, принимающее решение, осуществляет вначале попарное сравнение значимости выбранных критериев, затем этот же метод используется для попарного сравнения альтернатив относительно каждого выбранного критерия; на основе этого система поддержки принятия решений рассчитывает коэффициенты значимости критериев, коэффициенты значимости альтернатив относительно каждого критерия, что позволяет рассчитать для каждой альтернативы значения линейной функции полезности);
- метод аналитических сетевых процессов, который позволяет учесть взаимосвязи между критериями;
- многоцелевое оценивание альтернатив (каждая альтернатива оценивается единым показателем эффективности – степенью влияния его выполнения на достижение главной цели).

Системы поддержки принятия решений начинают все шире применяться государственными организациями и крупными корпорациями (U.S. Navy, NASA, IBM, GeneralMotors, Xerox, 3M, RockwellInternational, ReiterConsultingGroupInternational и др.). Примеры задач, решаемых с привлечением таких систем:

- обоснование направлений развития систем высшего образования США на период 1985-2000 гг.;
- выбор методов завоевания рынка бытовой техники;
- оценка привлекательности регионов США для трудоустройства людей, окончивших колледж,в ближайшие 10 лет;
- распределение средств между мероприятиями, направленными на уменьшение бандитизма;

- оценка перспективности видов альтернативного горючего для автомобилей;
- распределение средств между проектами социальной программы гуманитарной направленности;
- отбор научно-технических проектов в рамках конкурса;
- выбор перспективных направлений информатизации страны и пр.

В последнее время системы поддержки принятия решений начинают применяться и в интересах малого и среднего бизнеса (например, выбор варианта размещения торговых точек, выбор кандидатуры на замещение вакантной должности, выбор варианта информатизации и т.д.).

Классификация систем поддержки принятия решений

Общепринятой исчерпывающей классификации систем поддержки принятия решений не существует, но системы поддержки принятия решений можно разделить по нескольким уровням.

На уровне пользователя системы поддержки принятия решений можно разделить на:

- пассивные;
- активные;
- кооперативные.

Пассивной системой поддержки принятия решений называется система, которая помогает процессу принятия решения, но не может вынести предложение, какое решение принять.

Активная система может сделать предложение, какое решение следует выбрать.

Кооперативная система позволяет лицу, принимающему решение, изменять, пополнять или улучшать решения, предлагаемые системой, посылая затем эти изменения в систему для проверки. Система изменяет, пополняет или улучшает эти решения и посылает их опять пользователю. Процесс продолжается до получения согласованного решения.

На концептуальном уровне выделяют следующие системы поддержки принятия решений:

- управляемые сообщениями;
- управляемые данными;
- управляемые документами;
- управляемые знаниями;

- управляемые моделями.

Система, управляемая сообщениями, поддерживает группу пользователей, работающих над выполнением общей задачи.

Системы, управляемые данными, ориентируются на доступ и манипуляции с данными.

Системы, управляемые документами, осуществляют поиск и манипулируют неструктурированной информацией, заданной в различных форматах.

Системы, управляемые знаниями, обеспечивают решение задач в виде фактов, правил, процедур.

Системы, управляемые моделями, базируются на математических моделях (статистических, финансовых, оптимизационных, имитационных). Для их построения можно использовать OLAP-системы, позволяющие осуществлять сложный анализ данных, и тогда такую систему поддержки принятия решения можно отнести к гибридным системам, которые обеспечивают моделирование, поиск и обработку данных.

На уровне данных, с которыми эти системы работают, условно можно выделить:

- оперативные;
- стратегические.

Оперативные системы поддержки принятия решений предназначены для немедленного реагирования на изменения текущей ситуации в управлении финансово-хозяйственными процессами компании.

Стратегические системы ориентированы на анализ значительных объемов разнородной информации, собираемой из различных источников.

На уровне решаемой задачи и области применения выделяют системы поддержки принятия решений:

- первого класса;
- второго класса;
- третьего класса.

Системы поддержки принятия решений *первого класса*, обладающие наибольшими функциональными возможностями, предназначены для применения в органах государственного управления высшего уровня (администрация президента, министерства) и органах управления больших компаний (совет директоров корпорации) при планировании крупных комплексных целевых программ для обоснования решений относительно включения в программу различных политических, социальных или экономических мероприятий и распределения между ними ресурсов на

основе оценки их влияния на достижение основной цели программы. Системы поддержки принятия решений этого класса являются системами коллективного пользования, базы знаний которых формируются многими экспертами – специалистами в различных областях знаний.

Системы поддержки принятия решений *второго класс*а являются системами индивидуального пользования, базы знаний которых формируются непосредственным пользователем. Они предназначены для использования государственными служащими среднего ранга, а также руководителями малых и средних фирм для решения оперативных задач управления.

Системы поддержки принятия решений *третьего класса* являются системами индивидуального пользования, адаптирующимися к опыту пользователя. Они предназначены для решения часто встречающихся прикладных задач системного анализа и управления (например, выбор субъекта кредитования, выбор исполнителя работы, назначение на должность и пр.). Такие системы обеспечивают получение решения текущей задачи на основе информации о результатах практического использования решений этой же задачи, принятых в прошлом. Кроме того, системы этого класса предназначены для использования в торговых предприятиях, торгующих дорогими товарами длительного пользования, в качестве средства «интеллектуальной рекламы», позволяющего покупателю выбрать товар на основе своего опыта применения товаров аналогичного назначения.

На уровне архитектуры системы поддержки принятия решений делятся на:

- функциональные системы поддержки принятия решений;
- независимые витрины данных;
- двухуровневое хранилище данных;
- трехуровневое хранилище данных.

Они отличаются организацией серверной стороны системы поддержки принятия решений. Характерной чертой функциональной системы является то, что анализ осуществляется с использованием данных из оперативных систем. На клиентской стороне располагается пользовательский интерфейс системы поддержки принятия решений, а на серверной – источники данных для задач принятия решений.

Витрина данных– база данных, функционально-ориентированная и, как правило, содержащая данные по одному из направлений деятельности организации. Она отвечает тем же требованиям, что и хранилище данных, но,

в отличие от хранилища, нейтрального к приложениям, в витрине данных информация хранится оптимизированно с точки зрения решения конкретных задач.

Кроме того, под витриной данных иногда понимают относительно небольшое хранилище данных или же часть более общего хранилища данных, специфицированную для использования конкретным подразделением в организации и/или определенной группой пользователей. Если в корпоративной системе имеется две витрины данных, то общие данные, имеющиеся в обеих секциях одновременно, должны быть представлены в секциях идентично.

Независимые витрины данных часто появляются в организации исторически и встречаются в крупных организациях с большим количеством независимых подразделений, зачастую имеющих свои собственные отделы информационных технологий.

Хранилище данных – предметно-ориентированный, интегрированный, неизменчивый, поддерживающий хронологию набор данных, организованный для целей поддержки принятия решений; может состоять из нескольких баз данных, имеет свою собственную модель хранения данных. Информация, поступившая в хранилище, не удаляется, не изменяется.

Двухуровневое хранилище данных строится централизованно для предоставления информации в рамках компании. Для поддержки такой архитектуры необходимы специалисты в области хранилищ данных. Это означает, что вся организация должна согласовать все определения и процессы преобразования данных.

Трехуровневое хранилище данных представляет собой единый централизованный источник корпоративной информации. Витрины данных представляют подмножества данных из хранилища, организованные для решения задач отдельных подразделений компании. Пользователи имеют возможность доступа к детальным данным хранилища в случае, если данных в витрине недостаточно, а также для получения более полной картины состояния бизнеса.

В большинстве реальных случаев система принятия решений вне зависимости от класса и архитектуры должна помочь лицу, принимающему решения, формализовать его собственные представления о ценности полученных результатов и затратах на их получение.

Главным в системе принятия решений является не вычислительная часть, а технологическая поддержка процедуры корректного извлечения и

формализации субъективных требований и предпочтений специалистов, а также процедуры пошагового агрегирования информации под контролем аналитика. Система поддержки принятия решений – не более чем средство технологической поддержки процедуры принятия решений, последнее слово всегда должно оставаться за экспертом.

Глоссарий

Основные определения по теме «История развития искусственного интеллекта»

Искусственный интеллект (ArtificialIntelligence, AI) – научное направление, в рамках которого ставятся и решаются задачи аппаратного или программного моделирования тех видов человеческой деятельности, которые традиционно считаются интеллектуальными (представление знаний, обучение, общение и т.п.).

Интеллектуальная система– система или устройство с программным обеспечением, имеющие возможность с помощью встроенного процессора настраивать свои параметры в зависимости от состояния внешней среды.

Эвристика – процесс поиска решений; прием решения задачи, основанный не на строгих математических моделях и алгоритмах, а на соображениях, восходящих к «здравому смыслу»; отражает особенности того, как такие задачи решает человек, когда он не пользуется строго формальными приемами.

Кибернетика –наука об управлении, связи и переработке информации. Основным объектом исследования кибернетики являются абстрактные кибернетические системы – от компьютеров до человеческого мозга и человеческого общества.

Теория игр– математическая теория предсказания результатов игр, в которых участники не имеют полной информации о намерениях друг друга. Теория игр используется для описания процессов, происходящих на олигополистических рынках, и в теории фирм.

Теория принятия решений– область исследования, изучающая закономерности выбора людьми путей решения разного рода задач и исследующая способы поиска наиболее выгодных из возможных решений.

Когнитивная психология– направление в психологической науке, изучающее зависимость поведения субъекта от познавательных процессов. Главное в когнитивной психологии – выделение некоторых общих компонентов, структур, процессов, характерных для познания в целом. В этом плане когнитивная психология – это современная психология познавательных процессов.

Основные определения по теме «Направления исследований в области искусственного интеллекта»

Нейрокибернетика –научное направление, изучающее основные закономерности организации и функционирования нейронов и нейронных образований. Основным методом нейрокибернетики является математическое моделирование, при этом данные физиологического эксперимента используются в качестве исходного материала для создания моделей.

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*. Как следствие этого, введены специфические связи между элементами, которые являются предметом отдельного рассмотрения.

Нейрокомпьютинг–скороразвивающаяся область вычислительных технологий, стимулированная исследованиями мозга. Вычислительные операции выполняются огромным числом сравнимо обычных, нередко адаптивных процессорных частей. Нейрокомпьютинг, по собственному происхождению, совершенно приспособлен для сравнения образов, определения образов и синтеза систем управления.

Робот – автоматическое устройство с антропоморфным действием, которое частично или полностью заменяет человека при выполнении работ в опасных для жизни условиях или при относительной недоступности объекта. Робот может управляться оператором либо работать по заранее составленной программе. Использование роботов позволяет облегчить или вовсе заменить человеческий труд на производстве, в строительстве, при работе с тяжелыми грузами, вредными материалами, а также в других тяжелых или небезопасных для человека условиях.

Компьютерная лингвистика(computationallinguistics) – область использования компьютеров для моделирования функционирования языка в тех или иных условиях или проблемных областях, а также сфера применения компьютерных моделей языка в лингвистике и др. дисциплинах.

Распознавание образов (Patternrecognition) – разделение образов в неком пространстве на классы. Образ традиционно представляется в виде вектора измеренных величин.

Распознавание речи (Speechrecognition) – автоматическое разложение звукового вида на фонемы и слова.

Естественный язык– в лингвистике любой язык общения между людьми. Под естественностью некоторого языка понимается наличие

синонимии и омонимии слов и словосочетаний, а также свободный порядок слов в предложении.

Проблемная область интеллектуальной системы определяется предметной областью и решаемыми в ней задачами.

Предметную область можно характеризовать описанием области в терминах пользователя, а задачи – их типом. С точки зрения разработчика, выделяются статические и динамические предметные области. Предметная область называется статической, если описывающие ее исходные данные не изменяются во времени. При этом производные данные (выводимые из исходных) могут появляться заново и изменяться (не изменяя при этом исходных данных). Если исходные данные, описывающие предметную область, изменяются за время решения задачи, то предметную область называют динамической.

Основные определения по теме «Представление знаний»

Данные– это отдельные факты, характеризующие объекты, процессы и явления предметной области, а также их свойства; сведения, полученные путем измерения, наблюдения, логических или арифметических операций, представленные в форме, пригодной для постоянного хранения, передачи и (автоматизированной) обработки.

Знания– это закономерности предметной области (принципы, связи, законы), полученные в результате практической деятельности и профессионального опыта, позволяющие специалистам ставить и решать задачи в этой области; это хорошо структурированные данные, или данные о данных, или метаданные.

Поверхностные знания– знания о видимых взаимосвязях между отдельными событиями и фактами в предметной области.

Глубинные знания– абстракции, аналогии, схемы, отображающие структуру и природу процессов, протекающих в предметной области. Эти знания объясняют явления и могут использоваться для прогнозирования поведения объектов.

Процедурные знания– знания, «растворенные» в алгоритмах.

Декларативными знаниями считаются предложения, записанные на языках представления знаний, приближенных к естественному и понятных неспециалистам.

Эмпирические знания– знания, которые могут добываться ИС путем наблюдения за окружающей средой.

Поле знаний– поле,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.

Семантическая сеть– это ориентированный граф, вершины которого – понятия, а дуги – отношения между ними.

Фрейм–это абстрактный образ для представления некоего стереотипа восприятия.

Основные определения по теме «Нейронные сети»

Нейрон(биологический) – клетка мозга, способная генерировать электрический импульс в случае, когда суммарный потенциал превысит критическую величину. Соединяясь друг с другом, нейроны образуют сеть, по которой путешествуют электрические импульсы; именно нейронные сети мозга обрабатывают информацию. При этом «обучение» сети и запоминание информации базируется на настройке значений весов связей между нейронами.

Синапс (вес, синаптический вес) – межнейронное соединение, однонаправленная входная связь нейрона, соединенная с выходом другого нейрона.

Аксон– выходная связь нейрона: при помощи аксона нейрон передает собственный выходной сигнал.

Искусственная нейронная сеть (Artificialneuralnetwork)– это система, состоящая из многих простых вычислительных элементов, работающих параллельно, функция которых определяется структурой сети, силой взаимных связей, а вычисления производятся в самих элементах или узлах.

Нейронные сети–класс моделей, основанных на биологической аналогии с мозгом человека и предназначенных после прохождения этапа так называемого обучения на имеющихся данных для решения разнообразных задач анализа данных.

Нейронная сеть – это процессор с массивным распараллеливанием операций, обладающий естественной способностью сохранять экспериментальные знания и делать их доступными для последующего использования. Он похож на мозг в двух отношениях: сеть приобретает знания в результате процесса обучения и для хранения информации

используются величины интенсивности межнейронных соединений, которые называются синаптическими весами.

Нейрокомпьютер– это вычислительная система с архитектурой аппаратного и программного обеспечения, адекватной выполнению алгоритмов, представленных в нейросетевом логическом базисе.

Обучение нейронной сети (Training) – целенаправленный процесс изменения межслойных синаптических связей, итеративно повторяемый до тех пор, пока сеть не приобретет необходимые свойства.

Обучение с учителем, или обучение контролируемое, или обучение управляемое (Supervisedlearning, Associativelearning). Обучение с учителем предполагает, что для каждого входного вектора существует целевой вектор, представляющий собой требуемый выход. Вместе они называются обучающей парой. Предъявляется выходной вектор, вычисляется выход сети и сравнивается с соответствующим целевым вектором, разность(ошибка) с помощью обратной связи подается в сеть, и веса изменяются в соответствии с алгоритмом, стремящимся минимизировать ошибку. Векторы обучающего множества предъявляются последовательно, вычисляются ошибки, и веса подстраиваются для каждого вектора до тех пор, пока ошибка по всему обучающему массиву не достигнет приемлемо низкого уровня.

Обучение без учителя, или самообучение, или обучение неконтролируемое, или обучение неуправляемое (Unsupervisedlearning, Self-organization). Алгоритм обучения без учителя подстраивает веса сети так, чтобы получались согласованные выходные векторы, то есть чтобы предъявление достаточно близких входных векторов давало одинаковые выходы. Процесс обучения выделяет статистические свойства обучающего множества и группирует сходные векторы в классы.

Переобучениесети (Over training, Overfitting). Если в результате обучения нейронная сеть хорошо распознает примеры из обучающего множества, но не приобретает свойство обобщения, то есть не распознает или плохо распознает любые другие примеры, кроме обучающих, то говорят, что сеть переобучена. Переобучение– это результат чрезмерной подгонки сети к обучающим примерам.

Сходимость процесса обучения (Coincidenceofthelearningalgorithm). Целью процедуры минимизации является отыскание глобального минимума, достижение его называется сходимостью процесса обучения.

Задача классификации(Classificationproblem) заключается в разбиении объектов на классы, когда основой разбиения служит вектор параметров

объекта. Объекты в пределах одного класса считаются эквивалентными с точки зрения критерия разбиения. Сами классы часто бывают неизвестными заранее и формируются динамически (как, например, в сетях Кохонена). Классы зависят от предъявляемых объектов, и поэтому добавление нового объекта требует корректирования системы классов.

Кластеризация (Clustering) – это один из методов анализа данных, позволяющих классифицировать многомерные наблюдения, каждое из которых описывается набором переменных X1,X2, …, Xn. Целью кластеризации является образование групп схожих между собой объектов.

Основные определения по теме «Эволюционное моделирование»

Эволюционное моделирование– направление в математическом моделировании, объединяющее компьютерные методы моделирования эволюции, а также близкородственные по источнику заимствования идей другие направления в эвристическом программировании. Включает в себя как разделы генетические алгоритмы, эволюционные стратегии, эволюционное программирование, искусственные нейронные сети, нечеткую логику.

Генетический алгоритм (Geneticalgorithm, как направление исследований) – раздел эволюционного моделирования, заимствующий методические приемы из теоретических положений популяционной генетики. Представляет собой своего рода модель машинного исследования поискового пространства, построенную на эволюционной метафоре. Характерные особенности: использование строк фиксированной длины для представления генетической информации, работа с популяцией строк, использование генетических операторов для формирования будущих поколений. Генетические алгоритмы, являясь одной из парадигм эволюционных вычислений, представляют собой алгоритмы случайного направленного поиска для построения (суб)оптимального решения данной проблемы, который моделирует процесс естественной эволюции.

Генетические алгоритмы (как метод) – адаптивные методы поиска, которые используются для решения задач функциональной оптимизации.

Кроссовер, скрещивание(Crossover) – процедура или оператор в генетических алгоритмах, используемые для получения разнообразия в процессе воспроизводства. При одноточечном кроссовере берутся две хромосомы потомка, на них случайным образом выбирается точка, и для этой точки происходит обмен генетического материала потомков. При

двухточечном кроссовере происходит тоже самое, только выбираются случайным образом две точки.

Мутация (Mutation) – оператор в генетических алгоритмах, предназначенный для внесения разнообразия в процесс размножения; с очень малой вероятностью двоичная мутация заменяет биты в хромосоме случайными битами, значение этой вероятности является параметром генетического алгоритма.

Ген (Gene) в генетических алгоритмах представляет собой основную единицу информации, определяющую характеристику особи. Например, если мы используем генетический алгоритм для обучения нейронной сети, то тогда в качестве генов мы будем использовать веса связей между нейронами. Гены в реализации генетических алгоритмов обычно представляют собой битовые строки фиксированной длины.

Генотип(Genotype) – представление особи в терминах генетического алгоритма.

Фенотип (Phenotype)– представление особи в виде, имеющемся в реальном мире.

Хромосома, особь (Chromosome) –базовый элемент генетического алгоритма. Она представляет собой набор генов, описывающих параметры особи и однозначно определяющих ее. Например, в задаче обучения нейронной сети хромосома будет содержать в себе полностью все настраиваемые параметры. Обычно при реализации генетического алгоритма размер хромосом от эпохи к эпохе не изменяется, хотя встречаются и исключения.

Приспособленность, фитнесс (Fitness) – параметр, определяющий, насколько хорошо данная особь отвечает требованиям. Приспособленность рассчитывается для каждой особи на основе данных, закодированных в генотипе, и используется для выбора наиболее приспособленных особей.

Популяция (Population)– совокупность особей, участвующих в генетических операциях. В классических реализациях алгоритма ее размер постоянен.

Эпоха (Epoch) – один этап функционирования генетического алгоритма. На нем осуществляется вычисление приспособленности каждой особи популяции. Затем на основании приспособленности отбираются хромосомы, участвующие в формировании следующей эпохи. Затем к ним применяются генетические операции, такие как скрещивание, мутация и т.д.

Основные определения по теме «Нечеткие множества и нечеткая логика»

Нечеткая логика (Fuzzylogic) –умозаключение с использованием нечетких множеств или множеств нечетких правил. Это направление восходит к первым работам по нечетким множествам, выполеннымЛофти Заде (LoftiZaden) в 1960-1970 гг.

Неопределенность является неотъемлемой частью процессов принятия решений. Неопределенности принято разделять на три класса:

- неопределенность, связанная с неполнотой наших знаний о проблеме, по которой принимается решение;
- неопределенность, которая возникает в связи с непредсказуемостью реакции окружающей среды на наши действия;
- неопределенность, связанная с неточным пониманием цели непосредственно самим ЛПР.

Нечеткое множество $\mathbf{A \subseteq X}$ представляет собой набор пар $A = \{\langle x, \mu_A(x)\rangle | x \in U\}$,где $x \in X$ и μ_A – функция принадлежности, то есть $\mu_A : U \to [0,1]$, которая представляет собой некоторую субъективную меру соответствия элемента нечеткому множеству и может принимать значения от нуля, который обозначает абсолютную непринадлежность, до единицы, которая, наоборот, говорит об абсолютной принадлежности элемента x нечеткому множеству A.

Нечетким числом называется выпуклое нормальное нечеткое множество с кусочно-непрерывной функцией принадлежности, заданное на множестве действительных чисел.

Лингвистическую переменную можно определить как переменную, значениями которой являются не числа, а слова или предложения естественного (или формального) языка.

Терм-множеством (termset) называется множество всех возможных значений лингвистической переменной.

Термом(term) называется любой элемент терм-множества. В теории нечетких множеств терм формализуется нечетким множеством с помощью функции принадлежности.

Дефаззификацией(defuzzification) называется процедура преобразования нечеткого множества в четкое число.

Фаззификацией(fuzzification) называется процедура преобразования четких значений в степени уверенности.

Нечетким логическим выводом называется получение заключения в виде нечеткого множества, соответствующего текущим значениям входов, с использованием нечеткой базы знаний и нечетких операций.

Нечеткой базой знаний называется совокупность нечетких правил «если… то…», определяющих взаимосвязь между входами и выходами исследуемого объекта. Обобщенный формат нечетких правил такой: *если <посылка правила>, то <заключение правила>*.

Посылка правила, или антецедент, представляет собой утверждение типа «*x* есть *низкий*», где «*низкий*»– это терм (лингвистическое значение), заданный нечетким множеством на универсальном множестве лингвистической переменной *x*. Квантификаторы «*очень*», «*более-менее*», «*не*», «*почти*» и т.п. могут использоваться для модификации термов антецедента.

Заключение, или следствие, правила представляет собой утверждение типа «*y* есть *d*», в котором значение выходной переменной *d* может задаваться:

- нечетким термом: «*y* есть *высокий*»;
- классом решений: «*y* есть *бронхит*»;
- четкой константой: «*y*=5»;
- четкой функцией от входных переменных: «*y*=5+4**x*».

Нечеткая система– множество нечетких правил, преобразующих входные данные в выходные. В простейшем случае эти правила устанавливает эксперт, в более сложном – например, нейросеть.

Нечеткое правило– условное высказывание вида «если X есть A, то Y есть B», где A и B – нечеткие множества.

Основные определения по теме «Экспертные системы»

Экспертные системы –это сложные программные комплексы, аккумулирующие знания специалистов в конкретных предметных областях и тиражирующие эти знания для консультации менее квалифицированных специалистов.

Пользователь– специалист предметной области, для которого предназначена система. Обычно его квалификация недостаточно высока, и поэтому он нуждается в помощи и поддержке своей деятельности со стороны экспертной системы.

Инженер по знаниям– специалист в области искусственного интеллекта, выступающий в роли промежуточного буфера между экспертом и базой знаний. Синонимы: *когнитолог, инженер-интерпретатор, аналитик.*

Интерфейс пользователя– комплекс программ, реализующих диалог пользователя с ЭС как на стадии ввода информации, так и при получении результатов.

База знаний– ядро ЭС, совокупность знаний предметной области, записанных на машинный носитель в форме, понятной эксперту и пользователю (обычно на некотором языке, приближенном к естественному). Параллельно такому «человеческому» представлению существует БЗ во внутреннем «машинном» представлении.

Решатель– программа, моделирующая ход рассуждений эксперта на основании знаний, имеющихся в БЗ. Синонимы: *дедуктивная машина, машина вывода, блок логического вывода.*

Подсистема объяснений– программа, позволяющая пользователю получить ответы на вопросы: «Как была получена та или иная рекомендация?» и «Почему система приняла такое решение?» Ответ на вопрос «как» – это трассировка всего процесса получения решения с указанием использованных фрагментов БЗ, то есть всех шагов цепи умозаключений. Ответ на вопрос «почему» – ссылка на умозаключение, непосредственно предшествовавшее полученному решению, то есть отход на один шаг назад. Развитые подсистемы объяснений поддерживают и другие типы вопросов.

Интеллектуальный редактор базы знаний– программа, представляющая инженеру по знаниям возможность создавать БЗ в диалоговом режиме. Включает в себя систему вложенных меню, шаблонов языка представления знаний, подсказок и других сервисных средств, облегчающих работу с базой.

Решение– процесс и результат выбора способа и цели действий из ряда альтернатив в условиях неопределенности.

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., то есть некоторые *объективизированные знания,* переведенные в форму, которая делает их доступными для потребителя.

Экспертные знания –знания, которые имеются у специалистов, но не зафиксированы во внешних по отношению к нему хранилищах. Экспертные знания являются *субъективными.*

Формализация. Процесс формализации знаний, полученных у эксперта, состоит из следующих шагов: выбор метода измерения нечеткости, получение исходных данных посредством опроса эксперта, реализация алгоритма построения функции принадлежности.

Интерпретация данных. Это одна из традиционных задач для экспертных систем. Под интерпретацией понимается процесс определения смысла данных, результаты которого должны быть согласованными и корректными. Обычно предусматривается многовариантный анализ данных.

Диагностика. Под диагностикой понимается процесс соотнесения объекта с некоторым классом объектов и/или обнаружение неисправности в некоторой системе. Неисправность – это отклонение от нормы. Такая трактовка позволяет с единых теоретических позиций рассматривать и неисправность оборудования в технических системах, и заболевания живых организмов, и всевозможные природные аномалии. Важной спецификой является здесь необходимость понимания функциональной структуры («анатомии») диагностирующей системы.

Мониторинг. Основная задача мониторинга – непрерывная интерпретация данных в реальном масштабе времени и сигнализация о выходе тех или иных параметров за допустимые пределы. Главные проблемы – «пропуск» тревожной ситуации и инверсная задача «ложного» срабатывания. Сложность этих проблем – в размытости симптомов тревожных ситуаций и необходимости учета временного контекста.

Проектирование. Проектирование состоит в подготовке спецификаций на создание «объектов» с заранее определенными свойствами. Под спецификацией понимается весь набор необходимых документов: чертеж, пояснительная записка и т.д. Основные проблемы здесь – получение четкого структурного описания знаний об объекте и проблема «следа». Для организации эффективного проектирования и в еще большей степени перепроектирования необходимо формировать не только сами проектные решения, но и мотивы их принятия. Таким образом, в задачах проектирования тесно связываются два основных процесса, выполняемых в рамках соответствующей ЭС: *процесс вывода решения и процесс объяснения.*

Прогнозирование. Прогнозирование позволяет предсказывать последствия некоторых событий или явлений на основании анализа имеющихся данных. Прогнозирующие системы логически выводят вероятные следствия из заданных ситуаций. В прогнозирующей системе обычно используется параметрическая динамическая модель, в которой значения параметров «подгоняются» под заданную ситуацию. Выводимые из этой

модели следствия составляют основу для прогнозов с вероятностными оценками.

Планирование. Под планированием понимается нахождение планов действий, относящихся к объектам, способным выполнять некоторые функции. В таких ЭС используются модели поведения реальных объектов с тем, чтобы логически вывести последствия планируемой деятельности.

Обучение. Под обучением понимается использование компьютера для обучения какой-то дисциплине или предмету. Системы обучения диагностируют ошибки при изучении какой-либо дисциплины с помощью ЭВМ и подсказывают правильные решения. Они аккумулируют знания о гипотетическом «ученике» и его характерных ошибках, затем в работе они способны диагностировать слабости в познаниях обучаемых и находить соответствующие средства для их ликвидации. Кроме того, они планируют акт общения с учеником в зависимости от успехов ученика с целью передачи знаний.

Управление. Под управлением понимается функция организованной системы, поддерживающая определенный режим деятельности. Такого рода ЭС осуществляют управление поведением сложных систем в соответствии с заданными спецификациями.

Оптимизация – нахождение решения, удовлетворяющего системе ограничений и максимизирующего или минимизирующего целевую функцию.

Основные определения по теме «Системы поддержки принятия решений»

Принятие решения – это особый вид человеческой деятельности, направленный на выбор лучшей из имеющихся альтернатив. Главной задачей, которую приходится разрешать при принятии решения, является выбор альтернативы, наилучшей для достижения некоторой цели, или ранжирование множества возможных альтернатив по степени их влияния на достижение этой цели.

Процесс принятия решений – получение и выбор наиболее оптимальной альтернативы с учетом просчета всех последствий. Необходимо выбирать ту альтернативу, которая наиболее полно отвечает поставленной цели, но при этом приходится учитывать большое количество противоречивых требований и, следовательно, оценивать выбранный вариант решения по многим критериям.

Системы поддержки принятия решений **(DSS, DecisionSupportSystem)** являются человеко-машинными объектами, которые позволяют лицам, принимающим решения, использовать данные, знания, объективные и субъективные модели для анализа и решения слабоструктурированных и неструктурированных проблем.

Хранилище данных – предметно-ориентированный, интегрированный, неизменчивый, поддерживающий хронологию набор данных, организованный для целей поддержки принятия решений.

Витрина данных – упрощенный вариант хранилища данных, содержащий только тематически объединенные данные.

Витрина данных, секция данных (DataMart) – база данных, функционально-ориентированная и, как правило, содержащая данные по одному из направлений деятельности организации. Она отвечает тем же требованиям, что и хранилище данных, но, в отличие от хранилища, нейтрального к приложениям, в витрине данных информация хранится оптимизированно с точки зрения решения конкретных задач.

Список использованных источников

1. Бессмертный, И.А. Искусственный интеллект[Электронный ресурс]: учебное пособиес грифом УМО/ И.А. Бессмертный. СПб: СПбГУ ИТМО, 2010. – Режим доступа: http://window.edu.ru/window/catalog?p_rid=69274
2. Вагин В.Н., Головина Е.Ю., Загорянская А.А., Фомина М.В. Достоверный и правдоподобный вывод в интеллектуальных системах. 2-е издание // Под редакцией В.Н. Вагина, Д.А. Поспелова. – М.: ФИЗМАТЛИТ, 2008. 712 с.
3. Варшавский П.Р., Куриленко И.Е., Михайлов И.С. Программное обеспечение интеллектуальных систем: учебное пособие. – М.: Издательский дом МЭИ, 2011. – 64 с.
4. Гаврилова, Т. А. Базы знаний интеллектуальных систем / Т. А. Гаврилова, В. Ф. Хорошевский. – СПб. : Питер, 2001. – 384 с.
5. Гаскаров, Д. В. Интеллектуальные информационные системы : учебник для вузов / Д. В. Гаскаров. – М. : Высшая школа, 2003. – 431 с.
6. Гладков Л.А., Курейчик В.В., Курейчик В.М. Генетические алгоритмы [Текст]/ Под ред. В.М. Курейчика. — 2-е изд., испр. и доп. — М.: ФИЗМАТЛИТ, 2006. — 320 с. — ISBN 5-9221-0510-8.
7. Глухих И. Интеллектуальные информационные системы. – М.: Академия, 2010. 112 с.
8. Джарратано, Д. Экспертные системы: принципы разработки и программирование : пер. с англ. / Д. Джарратано, Г. Райлт. – 4-е изд. – М. : ООО «И.Д. Вильямс», 2007. – 1152 с.
9. Любарский, Ю. Я. Интеллектуальные информационные системы / Ю. Я. Любарский. – М. : Наука, 1990. – 227 с.
10. Методы и модели анализа данных: OLAP и DataMining / А. А. Барсегян и др. – СПб. : БХВ-Петербург, 2004. – 336 с.
11. Мигас С.С. Интеллектуальные информационные системы. – СПбГИЭУ, 2009. 160 с.
12. Миркес, Е. М. Нейроинформатика : учеб.пособие для студентов / Е. М. Миркес. – Красноярск : ИПЦ КГТУ, 2002. – 347 с.
13. Поликарпова Н. И., Шалыто А. А. Автоматное программирование. СПб.: Питер, 2010.
14. Потапов, А.С. Технологии искусственного интеллекта[Электронный ресурс]: учебное пособиес грифом УМО/ А. С. Потапов. СПб: СПбГУ ИТМО, 2010. – Режим доступа: http://window.edu.ru/window/catalog?p_rid=69612

15. Путькина Л.В., Пискунова Т.Г. Интеллектуальные информационные системы. – СПб.: СПбГУП, 2008. – 228 с.
16. Рассел, С. Искусственный интеллект: современный подход / С. Рассел, П. Норвиг. – 2-е изд. – М. : Вильямс, 2006. – 1408 с.
17. Редько В.Г. Эволюция, нейронные сети, интеллект. Модели и концепции эволюционной кибернетики - 6-е изд. – М.: Эдиториал УРСС, 2009. – 224 с.
18. Рутковская, Д. Нейронные сети, генетические алгоритмы и нечеткие системы / Д. Рутковская, М. Пилиньский, Л. Рутковский ; пер. с польск. И. Д. Рудинского. – М. : Горячая линия – Телеком, 2008. – 452 с.
19. Рутковский Лешек Искусственные нейронные сети. Теория и практика. — М.: Горячая линия - Телеком, 2010. — 520 с. — ISBN 978-5-9912-0105-6
20. Ручкин, В. Н. Универсальный искусственный интеллект и экспертные системы / В. Н. Ручкин, В. А. Фулин. – СПб. : БХВ-Петербург, 2009. – 240 с.
21. Системы искусственного интеллекта. Практический курс [Электронный ресурс]: учебное пособиес грифом УМО/ В. А. Чулюков[и др.]. – М.: БИНОМ. Лаборатория знаний, 2008. – Режим доступа: http://window.edu.ru/window/catalog?p_rid=65335
22. Скобцов Ю. А. Основы эволюционных вычислений. — Донецк: ДонНТУ, 2008. — С. 326. — ISBN 978-966-377-056-6
23. Спицын В.Г., Цой Ю.Р. Представление знаний в информационных системах: Учебное пособие. – Томск: Изд-во ТПУ, 2008. – 152 с.
24. Частиков, А. П. Разработка экспертных систем. Среда CLIPS / А. П. Частиков, Т. А. Гаврилова, Д. Л. Белов. – СПб. : БХВ-Петербург, 2003. – 608 с.
25. Штовба С.Д. Проектирование нечетких систем средствами MATLAB. М.: Горячая линия - Телеком.- 2007.- 288 с.
26. Ярушкина, Н. Г. Основы теории нечетких и гибридных систем : учеб.пособие / Н. Г. Ярушкина. – М. : Финансы и статистика, 2004. – 320 с.

Интернет ресурсы:

1. Neuroph Studio http://neuroph.sourceforge.net/download.html
2. CLIPS - A Tool for Building Expert Systems http://clipsrules.sourceforge.net/
3. jColibri - http://gaia.fdi.ucm.es/research/colibri/jcolibri
4. DarwinBotsWikiManual - http://wiki.darwinbots.com/w/Main_Page

5. FisPro - http://www7.inra.fr/mia/M/fispro/fisprodocen/QUICKSTART/quickstart.html