Министерство образования и науки РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

ЭКСПЕРТНЫЕ СИСТЕМЫ

сборник учебно-методических материалов

для направления подготовки 09.03.02 – Информационные системы и технологии

Благовещенск, 2007

Печатается по решению
редакционно-издательского совета
факультета математики и информатики
Амурского государственного
университета

Составитель: Акилова И.М.

Экспертные системы: сборник учебно-методических материалов для направления подготовки 09.03.02 «Информационные системы и технологии». – Благовещенск: Амурский гос. ун-т, 2017.

КРАТКОЕ ИЗЛОЖЕНИЕ ЛЕКЦИОННОГО МАТЕРИАЛА

Лекция №1 ***Тема: Понятие экспертной системы. Смысл экспертного анализа.***

Компьютерную программу можно назвать экспертом, если:

1) если такая программа *обладает знаниями*;

2) знания, которыми обладает программа, должны быть сконцентрированы на *определенную предметную область.* Случайный набор имен, дат и мест событий и т.п. – это отнюдь не те знания, которые могут послужить основой для программы, претендующей, претендующей на способность выполнить экспертный анализ. Знания предполагают определенную организацию и интеграцию – то есть определенные сведения должны соотноситься друг с другом и образовывать нечто вроде цепочки, в которой одно звено "тащит" за собой следующее;

3) из этих знаний должно непосредственно вытекать *решение проблем.*

Экспертная система – это программа для компьютера, которая оперирует со знаниями в определенной предметной области с целью выработки рекомендаций или решения проблем.

Экспертная система может полностью взять на себя функции, выполнение которых обычно требует привлечения опыта человека-специалиста, или играть роль ассистента для человека, принимающего решение. Другими словами, система (техническая или социальная), требующая принятия решения, может получить его непосредственно от программы или через промежуточное звено – человека, который общается с программой. Тот, кто принимает решение, может быть экспертом со своими собственными правами, и в этом случае программа может "оправдать" свое существование, повышая эффективность его работы. Альтернативный вариант – человек, работающий в сотрудничестве с такой программой, может добиться с ее помощью результатов более высокого качества. Вообще говоря, правильное распределение функций между человеком и машиной является одним из ключевых условий высокой эффективности внедрения экспертных систем.

Технология экспертных систем является одним из направлений новой области исследования, которая получила наименование *искусственного интеллекта*. Исследования в этой области сконцентрированы на разработке и внедрении компьютерных программ, способных эмулировать (имитировать, воспроизводить) те области деятельности человека, которые требуют мышления, определенного мастерства и накопленного опыта. К ним относятся задачи принятия решений, распознавания образов и понимания человеческого языка. Эта технология уже успешно применяется в некоторых областях техники и жизни общества – органической химии, поиске полезных ископаемых, медицинской диагностике. Перечень типовых задач, решаемых экспертными системами, включает:

- ♦ извлечение информации из первичных данных (таких как сигналы, поступающие от гидролокатора);
- ♦ диагностика неисправностей (как в технических системах, так и в человеческом организме);
- ♦ структурный анализ сложных объектов (например, химических соединений);
- ♦ выбор конфигурации сложных многокомпонентных систем (например, распределенных компьютерных систем);
- ♦ планирование последовательности выполнения операций, приводящих к заданной цели (например, выполняемых промышленными роботами).

Характеристики экспертных систем

Экспертная система отличается от прочих прикладных программ наличием следующих признаков:

- ♦ Моделирует не столько физическую (или иную) природу определенной проблемной области, сколько *механизм мышления человека* применительно к решению задач в этой проблемной области. Это существенно отличает экспертные системы от систем математического моделирования или компьютерной анимации. Нельзя сказать, что программа полностью воспроизводит психологическую модель специалиста в этой предметной области (эксперта), но важно, что основное внимание все-таки уделяется воспроизведению компьютерными средствами методики решения проблем, которая применяется экспертом, т.е. выполнению не-

которой части задач так же (или даже лучше), как это делает эксперт.

♦ Система, помимо выполнения вычислительных операций, формирует определенные *соображения и выводы, основываясь на тех знаниях,* которыми она располагает. Знания в системе представлены, как правило, на некотором специальном языке и хранятся отдельно от собственно программного кода, который и формирует выводы и соображения. Этот компонент программы принято называть *базой знаний.*

♦ При решении задач основными являются *эвристические и приближенные методы,* которые, в отличие от алгоритмических, не всегда гарантируют успех. Эвристика по существу, является *правилом влияния {rule of thumb),* которое в машинном виде представляет некоторое знание, приобретенное человеком по мере накопления практического опыта решения аналогичных проблем. Такие методы являются *приблизительными* в том смысле, что, во-первых, они не требуют исчерпывающей исходной информации, и, во-вторых, существует определенная степень уверенности (или неуверенности) в том, что предлагаемое решение является верным.

Экспертные системы отличаются и от других видов программ из области искусственного интеллекта.

♦ Экспертные системы имеют дело с предметами *реального мира,* операции с которыми обычно требуют наличия значительного опыта, накопленного человеком. Множество программ из области искусственного интеллекта являются сугубо исследовательскими и основное внимание в них уделяется абстрактным математическим проблемам или упрощенным вариантам реальных проблем (иногда их называют "игрушечными" проблемами), а целью выполнения такой программы является "повышение уровня интуиции" или отработка методики. Экспертные системы имеют ярко выраженную практическую направленность в научной или коммерческой области.

♦ Одной из основных характеристик экспертной системы является ее *производительность,* т.е. скорость получения результата и его достоверность (надежность). Исследовательские программы искусственного интеллекта могут и не быть очень быстрыми, можно примириться и с существованием в них отказов в отдельных ситуациях, поскольку, в конце концов, — это инструмент исследования, а не программный продукт. А вот экспертная система должна за приемлемое время найти решение, которое было бы не хуже, чем то, которое может предложить специалист в этой предметной области.

♦ Экспертная система должна обладать способностью *объяснить,* почему предложено именно такое решение, и *доказать его обоснованность.* Пользователь должен получить всю информацию, необходимую ему для того, чтобы быть уверенным, что решение принято "не с потолка". В отличие от этого, исследовательские программы "общаются" только со своим создателем, который и так (скорее всего) знает, на чем основывается ее результат. Экспертная система проектируется в расчете на взаимодействие с разными пользователями, для которых ее работа должна быть, по возможности, прозрачной.

Зачастую термин *система, основанная на знаниях* используется в качестве синонима термина *экспертная система,* хотя, строго говоря, экспертная система – это более широкое понятие. Система, основанная на знаниях, – это любая система, процесс работы которой основан на применении правил отношений к символическому представлению знаний, а не на использовании алгоритмических или статистических методов. Таким образом, программа, способная рассуждать о погоде, будет системой, основанной на знаниях, даже в том случае, если она не способна выполнить метеорологическую экспертизу. А вот чтобы иметь право называться метеорологической экспертной системой, программа должна быть способна давать прогноз погоды (другой вопрос – насколько он будет достоверен).

Суммируя все сказанное, отметим — экспертная система содержит знания в определенной предметной области, накопленные в результате практической деятельности человека (или человечества), и использует их для решения проблем, специфичных для этой области. Этим экспертные системы отличаются от прочих, "традиционных" систем, в которых предпочтение отдается более общим и менее связанным с предметной областью теоретическим ме-

тодам, чаще всего математическим. Процесс создания экспертной системы часто называют *инженерией знаний* и он рассматривается в качестве "применения методов искусственного интеллекта".

Базовые функции экспертных систем

Поскольку теория экспертных систем выросла из более общей концепции искусственного интеллекта, то нет ничего удивительного в том, что проблематика этих областей имеет много общего.

1. Приобретение знаний

Приобретение знаний - это передача потенциального опыта решения проблемы от некоторого источника знаний и преобразование его в вид, который позволяет использовать эти знания в программе.

Передача знаний выполняется в процессе достаточно длительных и пространных собеседований между специалистом по проектированию экспертной системы (будем в дальнейшем называть его *инженером по знаниям*) и экспертом в определенной предметной области, способным достаточно четко сформулировать имеющийся у него опыт. По существующим оценкам, таким методом можно сформировать от двух до пяти "элементов знания" (например, правил влияния) в день. Конечно, это очень низкая скорость, а потому многие исследователи рассматривают функцию приобретения знаний в качестве одного из главных "узких мест" технологии экспертных систем.

Причинами такой низкой производительности являются:

- специалисты в узкой области, как правило, пользуются собственным жаргоном, который трудно перевести на обычный "человеческий" язык. Но смысл жаргонного "словечка" отнюдь не очевиден, а потому требуется достаточно много дополнительных вопросов для уточнения его логического или математического значения. Например, специалисты по военной стратегии говорят об "агрессивной демонстрации" иностранной военной мощи, но при этом не могут объяснить, чем такая "агрессивная" демонстрация отличается от демонстрации, не несущей угрозы;

- факты и принципы, лежащие в основе многих специфических областей знания эксперта, не могут быть четко сформулированы в терминах математической теории или детерминированной модели, свойства которой хорошо понятны. Так, эксперту в финансовой области может быть известно, что определенные события могут стать причиной роста или снижения котировок на фондовой бирже, но он ничего вам не скажет точно о механизмах, которые приводят к такому эффекту, или о количественной оценке влияния этих факторов. Статистические модели могут помочь сделать общий долговременный прогноз, но, как правило, такие методы не работают в отношении курсов конкретных акций на коротких временных интервалах.

- для того чтобы решить проблему в определенной области, эксперту недостаточно просто обладать суммой знаний о фактах и принципах в этой области. Например, опытный специалист знает, какого рода информацией нужно располагать для формулировки того или иного суждения, насколько надежны различные источники информации и как можно расчленить сложную проблему на более простые, которые можно решать более или менее независимо. Выявить в процессе собеседования такого рода знания, основанные на личном опыте и плохо поддающиеся формализации, значительно сложнее, чем получить простой перечень каких-то фактов или общих принципов.

- экспертный анализ даже в очень узкой области, выполняемый человеком, очень часто нужно поместить в довольно обширный контекст, который включает и многие вещи, кажущиеся эксперту само собой разумеющимися, но для постороннего отнюдь таковыми не являющиеся. Возьмем для примера эксперта-юриста, который принимает участие в судебном процессе. Очень трудно очертить количество и природу знаний общего рода, которые оказываются вовлечены в расследование того или иного дела.

Неудовлетворительные результаты подобных собеседований пробудили у некоторых исследователей интерес к автоматизации процесса передачи знаний специалистом машине. Одно из направлений исследований в этой области — *автоматизированное извлечение знаний* — появилось как побочный продукт в развитии систем человеко-машинного диалога. Другие исследователи полагают, что "расшить" это узкое место можно, двигаясь по пути *машинного*

обучения. Идея состоит в том, чтобы машина училась решать проблемы примерно так, как учится человек.

2. Представление знаний

Представление знаний— еще одна функция экспертной системы. Теория представления знаний — это отдельная область исследований, тесно связанная с философией формализма и когнитивной психологией. Предмет исследования в этой области — методы ассоциативного хранения информации, подобные тем, которые существуют в мозгу человека. При этом основное внимание, естественно, уделяется логической, а не биологической стороне процесса, опуская подробности физических преобразований.

В 70-х годах исследования в области представления знаний развивались в направлениях раскрытия принципов работы памяти человека, создания теорий извлечения сведений из памяти, распознавания и восстановления. Некоторые из достигнутых в теории результатов привели к созданию компьютерных программ, которые моделировали различие способы связывания понятий *(концептов).* Появились компьютерные приложения, которые могли некоторым образом отыскивать нужные "элементы" знания на определенном этапе решения некоторой проблемы. Со временем психологическая достоверность этих теорий отошла на второй план, а основное место, по крайней мере с точки зрения проблематики искусственного интеллекта, заняла их способность служить инструментом для работы с новыми информационными и управляющими структурами.

В области экспертных систем представление знаний интересует нас в основном как средство отыскания методов формального описания больших массивов полезной информации с целью их последующей обработки с помощью *символических вычислений. Формальное описание* означает упорядочение в рамках какого-либо языка, обладающего достаточно четко формализованным *синтаксисом* построения выражений и такого же уровня семантикой, увязывающей смысл выражения с его формой.

Символические вычисления означают выполнение нечисловых операций, в которых могут быть сконструированы символы и символьные структуры для представления различных концептов и отношений между ними. В области искусственного интеллекта ведется интенсивная работа по созданию *языков представления.* Под этим термином понимаются компьютерные языки, ориентированные на организацию описаний объектов и идей, в противовес статическим последовательностям инструкций или хранению простых элементов данных. Основными критериями доступа к представлению знаний являются логическая адекватность, эвристическая мощность и естественность, органичность нотации. Эти термины, скорее всего, нуждаются в пояснениях.

- *Логическая адекватность* означает, что представление должно обладать способностью распознавать все отличия, которые вы закладываете в исходную сущность. Например, невозможно представить идею, что каждое лекарство имеет какой-либо побочный нежелательный эффект, если только нельзя будет провести отличие между *предназначением* конкретного лекарственного препарата и его *побочным эффектом* (например, аспирин усугубляет язвенную болезнь). В более общем виде выражение, передающее этот эффект, звучит так: "каждое лекарство обладает нежелательным побочным эффектом, специфическим для этого препарата".
- *Эвристическая мощность* означает, что наряду с наличием выразительного языка представления должно существовать некоторое средство использования представлений, сконструированных и интерпретируемых таким образом, чтобы с их помощью можно было решить проблему. Часто оказывается, что язык, обладающий большей выразительной способностью в терминах количества семантических отличий, оказывается и больше сложным в управлении описанием взаимосвязей в процессе решения проблемы. Способность к выражению у многих из найденных формализмов может оказаться достаточно ограниченной по сравнению с английским языком или даже стандартной логикой. Часто уровень эвристической мощности рассматривается по результату, т.е. по тому, насколько легко оказывается извлечь нужное знание применительно к конкретной ситуации.

Знать, какие знания более всего подходят для решения конкретной проблемы, — это одно из качеств, которое отличает действительно специалиста, эксперта в определенной области, от новичка или просто начитанного человека.

♦ *Естественность нотации* следует рассматривать как некую добродетель системы, поскольку большинство приложений, построенных на базе экспертных систем, нуждается в накоплении большого объема знаний, а решить такую задачу довольно трудно, если соглашения в языке представления слишком сложны. Любой специалист скажет вам, что при прочих равных характеристиках лучше та система, с которой проще работать. Выражения, которыми формально описываются знания, должны быть по возможности простыми для написания, а их смысл должен быть понятен даже тому, кто не знает, как же компьютер интерпретирует эти выражения. Примером может служить *декларативный* программный код, который сам по себе дает достаточно четкое представление о процессе его выполнения даже тому, кто не имеет представления о деталях реализации компьютером отдельных инструкций.

За прошедшие годы было предложено немало соглашений, пригодных для кодирования знаний на языковом уровне. Среди них отметим *порождающие правила, структурированные объекты* и *логические программы.* В большинстве экспертных систем используется один или несколько из перечисленных формализмов, а доводы в пользу и против любого из них до сих пор представляют собой тему для оживленных дискуссий среди теоретиков.

3. Управление процессом поиска решения

При проектировании экспертной системы серьезное внимание должно быть уделено и тому, как осуществляется доступ к знаниям и как они используются при поиске решения. Знание о том, какие знания нужны в той или иной конкретной ситуации, и умение ими распорядиться — важная часть процесса функционирования экспертной системы. Такие знания получили наименование *метазнаний* — т.е. знаний о знаниях. Решение нетривиальных проблем требует и определенного уровня *планирования* и *управления* при выборе, какой вопрос нужно задать, какой тест выполнить, и т.д.

Использование разных стратегий перебора имеющихся знаний, как правило, оказывает довольно существенное влияние на характеристики эффективности программы. Эти стратегии определяют, каким способом программа отыскивает решение проблемы в некотором пространстве альтернатив. Как правило, не бывает так, чтобы данные, которыми располагает программа работы с базой знаний, позволяли точно "выйти" на ту область в этом пространстве, где имеет смысл искать ответ.

Большинство формализмов представления знаний может быть использовано в разных *режимах управления*, и разработчики экспертных систем продолжают экспериментировать в этой области.

4. Разъяснение принятого решения

Вопрос о том, как помочь пользователю понять структуру и функции некоторого сложного компонента программы, связан со сравнительно новой областью взаимодействия человека и машины, которая появилась на пересечении таких областей, как искусственный интеллект, промышленная технология, физиология и эргономика. На сегодня вклад в эту область исследователей, занимающихся экспертными системами, состоит в разработке методов представления информации о поведении программы в процессе формирования цепочки логических заключений при поиске решения.

Представление информации о поведении экспертной системы важно по многим причинам.

♦ *Пользователи,* работающие с системой, нуждаются в подтверждении того, что в каждом конкретном случае заключение, к которому пришла программа, в основном корректно.

♦ *Инженеры, имеющие дело с формированием базы знаний,* должны убедиться, что сформулированные ими знания применены правильно, в том числе и в случае, когда существует прототип.

♦ *Экспертам в предметной области* желательно проследить ход рассуждений и способ использования тех сведений, которые с их слов были введены в базу знаний. Это позволит судить,

насколько корректно они применяются в данной ситуации.

♦ *Программистам,* которые сопровождают, отлаживают и модернизируют систему, нужно иметь в своем распоряжении инструмент, позволяющий заглянуть в "ее нутро" на уровне более высоком, чем вызов отдельных языковых процедур.

♦ *Менеджер системы,* использующей экспертную технологию, который в конце концов несет ответственность за последствия решения, принятого программой, также нуждается в подтверждении, что эти решения достаточно обоснованы.

Способность системы объяснить методику принятия решения иногда называют *прозрачностью* системы. Под этим понимается, насколько просто персоналу выяснить, что делает программа и почему. Эту характеристику системы следует рассматривать в совокупности с режимом управления, поскольку последовательность этапов принятия решения тесно связана с заданной стратегией поведения.

Отсутствие достаточной прозрачности поведения системы не позволит эксперту повлиять на ее производительность или дать совет, как можно ее повысить. Прослеживание и оценка поведения системы — задача довольно сложная и для ее решения необходимы совместные усилия эксперта и специалиста по информатике.

Лекция № 2 ***Тема: Приобретение знаний экспертными системами***

Термин *приобретение знаний* носит обобщенный характер и совершенно нейтрален к способу передачи знаний. Например, передача может осуществляться с помощью специальной программы, которая в процессе обработки большого массива историй болезни устанавливает связь между симптомами и заболеваниями. А вот термин *извлечение знаний* относится именно к одному из способов передачи знаний – опросу экспертов в определенной проблемной области, который выполняется аналитиком или *инженером по знаниям.* Последний затем создает компьютерную программу, представляющую такие знания (или поручает это кому-нибудь другому, обеспечивая его всей необходимой информацией).

Этот же термин применяется и для обозначения процесса взаимодействия эксперта со специальной программой, целью которого является:

♦ извлечь каким-либо систематическим способом знания, которыми обладает эксперт, например, предлагая эксперту репрезентативные задачи и фиксируя предлагаемые способы их решения;

♦ сохранить полученные таким образом знания в некотором промежуточном виде;

♦ преобразовать знания из промежуточного представления в вид, пригодный для практического использования в программе, например в набор порождающих правил.

Преимущество использования такой программы – снижение трудоемкости процесса, поскольку перенос знаний от эксперта к системе осуществляется в один прием.

Теоретический анализ процесса приобретения знаний

В лекции 1 отмечалось, что при извлечении знаний в ходе опроса экспертов за рабочий день удается сформулировать от двух до пяти "эквивалентов порождающих правил". Причин такой низкой производительности несколько:

♦ прежде чем приступить к опросу экспертов, инженер по знаниям, который не является специалистом в данной предметной области, должен потратить довольно много времени на ознакомление с ее спецификой и терминологией; только после этого процесс опроса может стать продуктивным;

♦ эксперты склонны думать о знакомой им области не столько в терминах общих принципов, сколько в терминах отдельных типических объектов, событий и их свойств;

♦ для представления специфических знаний о предметной области нужно подобрать подходящую систему обозначений и структурную оболочку, что само по себе является непростой задачей.

Как известно, любую сложную задачу лучше всего разбить на подзадачи, и именно так мы поступим с задачей приобретения знаний.

Стадии приобретения знаний

В работе *[Buchanan et ai, 1983]* предлагается выполнить анализ процесса приобретения знаний в терминах модели процесса проектирования экспертной системы (рис. 1).

Рис. 1. Стадии приобретения знаний

(1) **Идентификация.** Анализируется класс проблем, которые предполагается решать с помощью проектируемой системы, включая данные, которыми нужно оперировать, и критерии оценки качества решений. Определяются ресурсы, доступные при разработке проекта, — источники экспертных знаний, трудоемкость, ограничения по времени, стоимости и вычислительным ресурсам.

(2) **Концептуализация.** Формулируются базовые концепции и отношения между ними. Сюда же входят и характеристика различных видов используемых данных, анализ информационных потоков и лежащих в их основе структур в предметной области в терминах причинно-следственных связей, отношений частное/целое, постоянное/временное и т.п.

(3) **Формализация.** Предпринимается попытка представить структуру пространства состояний и характер методов поиска в нем. Выполняется оценка полноты и степени достоверности (неопределенности) информации и других ограничений, накладываемых на логическую интерпретацию данных, таких как зависимость от времени, надежность и полнота различных источников информации.

(4) **Реализация.** Преобразование формализованных знаний в работающую программу, причем на первый план выходит спецификация методов организации управления процессом и уточнение деталей организации информационных потоков. Правила преобразуются в форму, пригодную для выполнения программой в выбранном режиме управления. Принимаются решения об используемых структурах данных и разбиении программы на ряд более или менее независимых модулей.

(5) **Тестирование.** Проверка работы созданного варианта системы на большом числе репрезентативных задач. В процессе тестирования анализируются возможные источники ошибок в поведении системы. Чаще всего таким источником является имеющийся в системе набор правил. Оказывается, что в нем не хватает каких-то правил, другие не совсем корректны, а между некоторыми обнаруживается противоречие.

Как видно из рис. 1, проектирование экспертной системы начинается с анализа класса проблем, которые предполагается решать с помощью этой системы. Было бы ошибкой приступать к проектированию системы, заранее задавшись определенной концепцией или определенной структурной организацией знаний. Весьма сомнительно, чтобы тот вариант концепции или тот способ организации идей, которым мы задались априори, взяв за основу предыдущие разработки, был приложим и к новой предметной области.

Уровни анализа знаний

Приведенное выше разделение на этапы встречается также и в работе Уилинги, который разработал моделирующий подход к инженерии знаний в рамках созданной им среды KADS (1992). В основе этого подхода лежит идея о том, что экспертная система является не контейнером, наполненным представленными экспертом знаниями, а "операционной мо-

делью", которая демонстрирует некоторое нужное нам поведение в столкновении с явлениями реального мира. Приобретение знаний, таким образом, включает в себя не только извлечение специфических знаний о предметной области, но и интерпретацию извлеченных данных применительно к некоторой концептуальной оболочке и формализацию их таким способом, чтобы программа могла действительно использовать их в процессе работы.

В основу оболочки KADS положено пять базовых принципов.

1) Использование множества моделей, позволяющее преодолеть сложность процессов инженерии знаний.

2) Четырехуровневая структура для моделирования требуемой экспертности — набора качеств, лежащих в основе высокого уровня работы специалистов.

3) Повторное использование родовых компонентов модели в качестве шаблонов, поддерживающих нисходящую стратегию приобретения знаний.

4) Процесс дифференциации простых моделей в сложные.

5) Важность преобразования моделей экспертности с сохранением структуры в процессе разработки и внедрения.

Рассмотрим подробно два первых принципа.

Главным мотивом создания оболочки KADS было преодоление сложности знаний. На сегодняшний день у инженеров по знаниям имеется возможность использовать при построении экспертных систем множество самых разнообразных методов и технологий. Однако при этом остаются три основных вопроса:

- ♦ определение проблемы, которую необходимо решить с помощью экспертной системы;
- ♦ определений функций, которые возлагаются на экспертную систему применительно к этой проблеме;
- ♦ определение задач, которые необходимо решить для выполнения возложенной функции.

Первый из принципов, положенных в основу KADS, состоит в том, что оболочка должна содержать множество частных моделей, помогающих найти ответ на эти вопросы. Примерами таких моделей могут служить:

- ♦ организационная модель "социально-экономической среды", в которой должна функционировать система, например финансовые услуги, здравоохранение и т.п.;
- ♦ прикладная модель решаемой проблемы и выполняемой функции, например диагностика, планирование расписания работ и т.д.;
- ♦ модель задач, демонстрирующая, как должна выполняться специфицированная функция, для чего производится ее разбиение на отдельные задачи, например сбор данных о доходах, формирование гипотез о заболеваниях.

Между этой терминологией и той, которой пользовался Бучанан, нет прямого соответствия, но можно сказать, что организационная и прикладная модели аналогичны стадии идентификации в предложенной Бучананом структуре.

В подходе, который реализован при создании KADS, стадия "концептуализации" разбивается на две части: модель кооперации, или коммуникации, и модель экспертности. Первая отвечает за декомпозицию процесса решения проблемы, формирование набора простейших задач и распределение их между исполнителями, в качестве которых могут выступать и люди, и машины. Вторая модель представляет процесс, который обычно называется извлечением знаний, т.е. анализ разных видов знаний, которые эксперт использует в ходе решения проблемы.

Кроме указанных, в состав оболочки KADS входит еще и модель проектирования, включающая технологии вычислений и механизмы представления знаний, которые могут быть использованы для реализации спецификаций, сформулированных предыдущими моделями.

На первый взгляд кажется, что представленный выше анализ в какой-то степени смазывает отличие между стадиями концептуализации и формализации. Можно, конечно, и возразить, что стадия формализации представляет собой просто более детальную проработ-

ку концепций и отношений, выявленных на ранних стадиях. Модель проектирования частично включает то, что в прежней схеме было отнесено к стадии реализации, но она не предполагает создание выполняемой программы.

В своей ранней работе Уилинга немного по-другому проводил разграничение между уровнями анализа (1986). Он рассматривал четыре уровня анализа.

- Концептуализация знаний. На этом уровне предполагалось формальное описание знаний в терминах принципиальных концепций и отношений между концепциями.
- Уровень *эпистемологического анализа.* Целью такого анализа было выявление структурных свойств концептуальных знаний, в частности таксономических отношений.
- Уровень *логического анализа.* Основное внимание уделялось тому, как строить логический вывод в данной предметной области на основе имеющихся знаний.
- Уровень *анализа внедрения.* Исследовались механизмы программной реализации системы.

В более поздней разработке три первых уровня включены в состав модели экспертности, а уровень анализа внедрения — в модель проектирования. Четырехуровневая структура KADS согласуется с предложенной Кленси схемой разделения знаний различного вида в соответствии с их ролью в процессе решения проблем. В частности, знания, касающиеся конкретной предметной области, теперь разделены на знания более высокого уровня (знания, относящиеся к построению логического вывода в этой предметной области), знания выбора решаемых задач и знания стратегии решения задач.

Эти уровни знаний представлены в табл. 1.

Категория знаний	Организация	Виды знаний
Стратегическая	Стратегии	Планы, метаправила
Задача	Задачи	Цели, управляющие термы, структуры задач
Логический вывод	Структура логического вывода	Источники знаний, метаклассы, схема предметной области
Предметная область	Теория предметной области	Концепции, свойства, отношения

Стратегический уровень управляет процессом выполнения задач, использующих при решении проблем методы логического вывода, подходящие для конкретной предметной области, и знания из этой области.

Описанная схема дифференциации знаний приводит к довольно простой архитектуре экспертной системы. В частности, оказывается, что даже в рамках традиционной архитектуры, предполагающей наличие базы знаний и машины логического вывода, можно неявным образом включить задачи и стратегии и в структуру знаний о предметной области, и в механизм построения логических заключений. Явное выделение этих задач и стратегий является главным моментом как в процессе приобретения знаний, так и в процессе проектирования структуры экспертной системы.

Онтологический анализ

Александер и его коллеги предложили еще один уровень анализа знаний, который получил название онтологического анализа. В основе этого полхода лежит описание системы в терминах сущностей, отношений между ними и преобразования сущностей, которое выполняется в процессе решения некоторой задачи. Автор указанной работы используют для структурирования знаний о предметной области три основные категории:

- *статическая онтология* — в нее входят сущности предметной области, их свойства и отношения;
- *динамическая онтология* — определяет состояния, возникающие в процессе решения проблемы, и способ преобразования одних состояний в другие;
- *эпистемическая онтология* — описывает знания, управляющие процессом перехода из

одного состояния в другое.

В этой схеме просматривается совершенно очевидное соответствие с уровнями *концептуализации знаний* и *эпистемологического анализа* в структуре, предложенной в уже упоминавшейся работе. Но на нижних уровнях — *логического анализа* и *анализа внедрения* — такое соответствие уже не просматривается. Онтологический анализ предполагает, что решаемая проблема может быть сведена к проблеме поиска, но при этом не рассматривается, каким именно способом нужно выполнять поиск. Примером практического применения такого подхода является система OPAL.

Рассматриваемая схема онтологического анализа выглядит довольно абстрактной, но ее ценность в том, что она упрощает анализ плохо структурированных задач. Каждый, кто сталкивался с выявлением знаний в процессе опроса человека-эксперта, знает, как трудно найти подходящую схему организации таких знаний. Чаще всего в таких случаях говорят: "Давайте воспользуемся фреймами или системой правил", откладывая таким образом выбор подходящего метода реализации на будущее, когда природа знаний эксперта станет более понятна.

Оболочки экспертных систем

На раннем этапе становления экспертных систем проектирование каждой очередной системы начиналось практически с нуля, в том смысле, что проектировщики для представления знаний и управления их применением использовали самые примитивные структуры данных и средства управления, которые содержались в обычных языках программирования. В редких случаях в существующие языки программирования включались специальные языки представлений правил или фреймов.

Такие специальные языки, как правило, обладали двумя видами специфических средств:

- *модулями* представления знаний (в виде правил или фреймов);
- *интерпретатором,* который управлял активизацией этих модулей.

Совокупность модулей образует базу знаний экспертной системы, а интерпретатор является базовым элементом машины логического вывода.

Эти компоненты могут быть повторно используемыми, т.е. служить основой для создания экспертных систем в разных предметных областях. Использование этих программ в качестве базовых компонентов множества конкретных экспертных систем позволило называть их *оболочкой* системы.

Лекция № 3 Тема: Оболочки экспертных систем. Система EMYCIN

Примером такой оболочки может служить система EMYCIN, которая является предметно-независимой версией системы MYCIN, т.е. это система MYCIN, но без специфической медицинской базы знаний. По мнению разработчиков, EMYCIN вполне может служить "скелетом" для создания консультационных программ во многих предметных областях, поскольку располагает множеством инструментальных программных средств, облегчающих задачу проектировщика конкретной экспертной консультационной системы. Она особенно удобна для решения дедуктивных задач, таких как диагностика заболеваний или неисправностей, для которых характерно большое количество ненадежных входных измерений (симптомов, результатов лабораторных гестов и т.п.), а пространство решений, содержащее возможные диагнозы, может быть достаточно четко очерчено.

Некоторые программные средства, впервые разработанные для EMYCIN, в дальнейшем стали типовыми для большинства оболочек экспертных систем. Среди таких средств следует отметить следующие.

- Язык представления правил. В системе EMYCIN такой язык использует систему обозначений, аналогичную языку ALGOL. Этот язык, с одной стороны, более понятен, чем LICP, а с другой — более строг и структурирован, чем тот диалект обычного английского, который использовался в MYC1N.

- Индексированная схема применения правил, которая позволяет сгруппировать правила, используя в качестве критерия группировки параметры, на которые ссылаются эти правила.

- Использование обратной цепочки рассуждений в качестве основной стратегии управления. Эта стратегия оперирует с И/ИЛИ-деревом, чьи листья представляют собой данные, которые могут быть найдены в таблицах или запрошены пользователем.

- Интерфейс между консультационной программой, созданной на основе EMYCIN, и конечным пользователем. Этот компонент оболочки обрабатывает все сообщения, которыми обмениваются пользователь и т.п.

- Интерфейс между разработчиком и программой, обеспечивающий ввод и редактирование правил, редактирование знаний, представленных в форме таблиц, тестирование правил и выполнение репрезентативных задач.

Значительная часть интерфейса реализуется отдельным компонентом EMYCIN — программой TEIRESIAS. Эта программа представляет собой "редактор знаний", который упрощает редактирование и сопровождение больших баз знаний. Редактор проверяет синтаксическую корректность правил, анализирует взаимную непротиворечивость правил в базе знаний и следит за тем, чтобы новое правило не являлось частным случаем существующих. Противоречие возникает, когда два правила с одинаковыми антецедентами имеют противоречивые консеквенты. Одно правило является частью другого в том случае, когда совокупность условий антецедента одного правила представляет собой подмножество совокупности условий другого правила, а их консеквенты одинаковы. По в состав TEIRESIAS не включены знания о какой-либо конкретной предметной облает или о стратегии решения проблем, которая может быть использована в проектируемой экспертной системе.

Используемые в системе методы синтаксического анализа могут быть применены к правилам, относящимся к любой предметной области.

Сопровождение и редактирование баз знаний с помощью программы TEIRESIAS

Как правило, человек-эксперт знает о той предметной области, в которой он является специалистом, гораздо больше, чем может выразить на словах. Вряд ли можно добиться от него многою, задавая вопросы в общем виде, например: "Что вам известно об инфекционных заболеваниях крови?" Гораздо продуктивнее подход, реализованный в про грамме TTIRHSIAS, который предполагает вовлечение эксперта в решение несложных репрезентативных задач из определенной предметной области и извлечение необходимых знаний в процессе такого решения.

Задавшись определенным набором базовых правил, представляющих прототип экспертной системы, TEIRESIAS решает в соответствии с этими правилами какую-нибудь из сформулированных репрезентативных проблем и предлагает эксперту критиковать результаты. В ответ эксперт должен сформулировать новые правила и откорректировать введенные ранее, а программа отслеживает внесенные изменения, анализирует их на предмет сохранения целостности и непротиворечивости всего набора правил, используя при этом *модели правил.* В процессе анализа используется обобщение правил различного вила.

Модели правил являются, по существу метаправилами, поскольку они предназначены для выработки суждений о правилах, а не об объектах предметной области приложения. В частности, в программе TEIRESIAS имеются метаправила, относящиеся к атрибутам правил объектного уровня. Такие правила обращают внимание пользователя на то, что в данных обстоятельствах целесообразно сначала исследовать определенные параметры, а уж затем в процессе отладки набора правил пытаться отслеживать влияние других параметров.

Существуют также средства, помогающие эксперту добавить новые варианты типов данных. Ошибки, которые обычно возникают при решении подобных задач, состоят в том, что новые типы данных имеют структуру, не согласующуюся с типами, уже существующими в системе.

Абстрактные данные, которые используются для формирования новых экземпляров структур данных, называются *схемами.* Эти схемы представляют собой обобщенные описания типов данных, точно так же, как структуры данных являются обобщениями конкретных данных.

Следовательно, схемы также могут быть организованы в виде иерархической структуры, в которой каждая схема, во-первых, наследует атрибуты, ассоциированные с ее предшественницей в иерархии, а во-вторых, имеет еще и собственные дополнительные атрибуты.

В таком случае процесс создания нового типа данных включает прослеживание пути от корня иерархии схем к той схеме, которая представляет интересующий нас тип данных. На каждом уровне имеются атрибуты, которые нужно конкретизировать, причем процесс продолжается до тех пор, пока не будет конкретизирована вся структура. Отношения между схемами в иерархии определяют последовательность выполнения задач обновления структур данных в системе.

Таким образом, в программе TLIRHSIAS можно выделить три уровня обобщения:

- знания об объектах данных, специфические для предметной области;
- знания о типах данных, специфические для метода представления знаний;
- знания, независимые от метода представления.

В составе TEIRESIAS имеются и средства, которые помогают оболочке EMYCIN следить за поведением экспертной системы в процессе применения набора имеющихся правил.

- *Режим объяснения* (EXPLAIN). После выполнения каждого очередною задания консультации — система дает объяснение, как она пришла к такому заключению. Распечатываются каждое правило, к которому система обращалась в процессе выполнения задания, и количественные параметры, связанные с применением этого правила, в том числе и коэффициенты уверенности.

- *Режим тестирования* (TEST). В этом режиме эксперт может сравнить результаты, полученные при протоне отлаживаемой программы, с правильными результатами решения этой же задачи, хранящимися в специальной базе данных, и проанализировать имеющиеся отличия. Оболочка EMYCIN позволяет эксперту задавал системе вопросы, почему она пришла к тому или иному заключению и почему при этом не были получены известные правильные результаты.

- *Режим просмотра* (REVIEW). В этом режиме эксперт может просмотреть выводы, к которым приходила система при выполнении одних и тех же запросов и библиотеки типовых задач. Это помотает просмотреть эффект, который дают изменения, вносимые в набор правил в процессе наладки системы. В этом же режиме можно проанализировать, как отражаются изменения в наборе правил на производительности системы.

Система EMYCIN была одной из первых попыток создать программный инструмент позволяющий перенести архитектору экспертной системы, уже эксплуатируемой в одной предметной области, на другие предметные области. Опыт, полученный в процессе работы с EMYCIN, показал, что те инструментальные средства, которые были включены в состав EMYCIN, пригодны для решения одних проблем и мало что дают при решении других.

Использование опроса экспертов для извлечения знаний в системе COMPASS

Для переключения номеров в телефонной сети используется довольно сложная система, которая может занимать большую часть здания телефонной станции. Основная задача при обслуживании системы переключений минимизировать число вызовов, которые необходимо перебросить на запасные маршруты из-за неисправности основных линий подключений, и быстро восстановить работу всей системы. Неисправность линий подключения может быть вызвана отказом каких-либо электронных схем, обеспечивающих связь между парой абонентов.

В процессе работы в системе переключения непрерывно выполняется самотестирование. При этом проверяется, нет ли разрыва в цепях короткого замыкания, замедления срабатывания переключающих схем и т.д. При возникновении каких-либо нестандартных ситуаций система самотестирования формирует соответствующее сообщение. Причина появления неисправности в системе переключения может быть выявлена только на основании множества таких сообщений, причем на помощь приходит опыт специалистов-экспертов. Эти сообще-

ния поступают в экспертную систему COMPASS, которая может предложить провести какой-либо специальный дополнительный тест или заменить определенный узел в системе (реле или плату). Система разработана компанией GTE и эксплуатируется во множестве ее филиалов *(1990).*

Накопление в системе знаний экспертов осуществлялось в процессе опроса. Эксперты описывали применяемые ими эвристические способы поиска неисправности, а инженеры по знаниям формулировали их в виде правил "если ... то". Затем эксперты повторно анализировали результаты формализации и проверяли, насколько эти правила согласуются с их опытом и интуицией. При обнаружении разночтений инженеры по знаниям изменяли формулировку правил и совместными усилиями с экспертами добивались, чтобы правила были приемлемыми.

Обычно правила вводились в систему в виде одного или нескольких производящих правил на языке KEE. Сформулированные па английском языке правила накапливались в библиотеке "документированных знаний", которая являлась одним из компонентов комплекта документации экспертной системы. Эта библиотека помогала сохранить "первоисточник знаний", что очень помогло в процессе настройки и опытной эксплуатации системы.

В процессе приобретения знаний большое внимание, по крайней мере, на первых порах, уделялось моделированию применения правил при поиске неисправностей "вручную", т.е. с помощью карандаша и бумаги. Цикл приобретения знаний при разработке системы COMPASS включал следующие этапы.

(1) В процессе собеседования с экспертом извлечь определенные знания.

(2) Задокументировать извлеченные знания.

(3) Проверить новые знания:

- попробовать применить их на разных наборах данных;
- смоделировать вручную, к каким результатам приведет использование этих знаний;
- сравнить результаты моделирования с теми, которые должны получиться по мнению эксперта;
- если результаты отличаются, то определить, какие именно правила и процедуры внесли "наибольший вклад" в это отличие; вернуться к п. (1) и выяснить у эксперта, как следует скорректировать подозрительное правило или процедуру.

Графически циклическая процедура приобретения знаний представлена на рис. 1.

После того как объем накопленных знаний превысит некоторый минимум, можно проверять работу системы на практике. При этом между этапами документирования и проверки знаний появляется еще один — внедрение знаний в систему. После этого можно проверять адекватность новых знаний не только моделированием вручную, но и выполнением программы на разных наборах входных данных. Конечно, анализ и сравнение результатов при этом усложняются, поскольку на ошибки в процессе формализации могут накладываться и ошибки реализации правил в работающей программе.

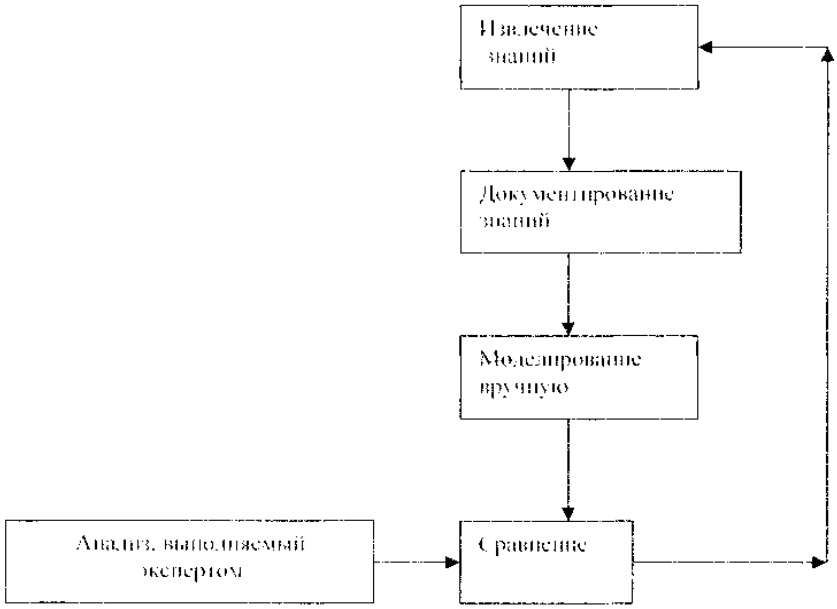


Рис. 1.

Прего (Prerau), ведущий разработчик системы, отметил, что по мере накопления опыта в процессе извлечения знаний инженеру по знаниям легче было общаться с экспертами. Последние постепенно освоились с методикой формализации знаний в виде правил, а инженер по знаниям достаточно глубоко ознакомился со спецификой предметной области. Такое сближение "стилей мышления" можно было рассматривать как признак успешного хода работы над проектом. Определенную помощь в этом, по наблюдению Прего, сыграло совместное участие инженера по знаниям и эксперта в ручном моделировании процесса принятия решений на основе полученных знаний и последующей проверке результатов.

Автоматизация процесса извлечения знаний в системе OPAL

Проект COMPASS можно считать одним из наиболее ярких примеров использования традиционной методики приобретения знаний, базирующейся на соответствующим образом организованном опросе экспертов. Такая методология "выросла" из предложенной Пьюэллом и Саймоном методики *анализа протокола.* Остановимся на проекте OPAL, в котором использована другая методика, отличающаяся от традиционной в двух важных аспектах.

- Эта методика ориентирована на частичную автоматизацию процесса извлечения знаний в ходе активного диалога интервьюируемого эксперта с программой.

Методика приобретения знаний предполагает использование стратегии, направляемой знаниями о предметном области.

Программа OPAL, пытается "вытянуть" из пользователя как можно больше деталей, касающихся представления знаний и их использования. OPAL не является программой общего назначения. Она разработана специально для диагностики онкологических заболеваний и предназначена для формирования правил принятия решений на основе полученных от эксперта знаний о планах лечения в том или ином случае.

Программа OPAL, упрощает процесс извлечения знаний, предназначенных для использования в экспертной системе ONCOCIN. Последняя формирует план лечения больных онкозаболеваниями и заинтересована в использовании модели предметной области для получения знаний непосредственно от эксперта с помощью средств графического интерфейса. Понятие *модель предметной области* можно трактовать в терминах знаний различного вида, которыми обладает эксперт.

Независимо от того, о какой конкретной предметной области идет речь, игре в шахматы или медицинской диагностике, всегда существуют некоторые предварительные условия или предварительный опыт, которыми должен обладать субъект или техническая си-

стема, чтобы воспринимать знания об этой предметной области. Если речь идет об игре в шахматы, то по крайней мере нужно знать правила этой игры. Применительно к медицинской диагностике нужно иметь представление о пациентах, заболеваниях, клинических тестах и т.п. Этот вид *фоновых,* или *фундаментальных.* знаний иногда в литературе по экспертным системам называют *глубокими знаниями,* противопоставляя их *поверхностным знаниям,* которые представляют собой хаотичный набор сведений о связях "стимул-реакция".

OPAL представляет собой программу извлечения знаний, которая обладает некоторыми фундаментальными знаниями в области терапии онкологических заболеваний. Программа использует эти базовые знания в процессе диалога с экспертом для извлечения дополнительных, более детальных знаний. Знания о предметной области нужны программе и для того, чтобы преобразовать информацию, полученную с терминала в процессе диалога, в исполняемый код — порождающие правила или таблицу состояний. Такая комбинация процесса наращивания знаний и их компиляции является одной из наиболее привлекательных возможностей той методологии построения экспертных систем, которая положена в основу системы OPAL. Графически основная идея представлена на рис. 2, где на человека-эксперта возлагается задача расширения и уточнения модели предметной области. Эта модель затем компилируется в программу, состоящую из процедур и порождающих правил. Поведение программы снова анализируется экспертом, который при необходимости вносит коррективы в модель и замыкает таким образом цикл итеративного процесса.

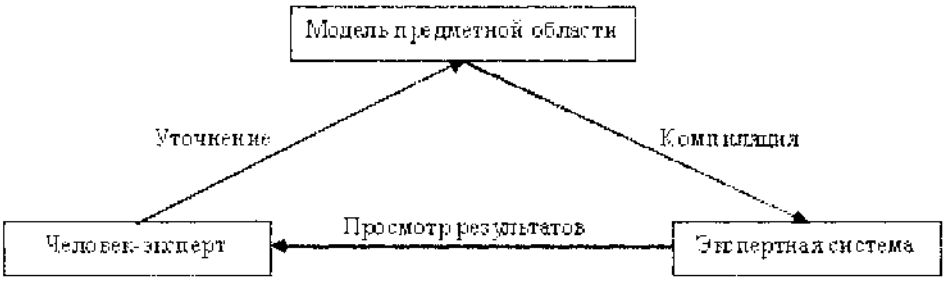


Рис. 2. Процесс приобретения знаний с использованием модели предметной области

В экспертной системе ONCOCIN используются три разных метода представления знаний:

- *иерархия объектов,* представляющая протоколы и их компоненты, в частности медикаменты;
- *порождающие прибила,* которые связаны с фреймами и формируют заключения о значениях медицинских параметров в процессе уточнения плана;
- *таблицы конечных состоянии* представляют собой последовательности терапевтических курсов (назначение и использование этих таблиц будет описано ниже).

Включение в систему ONCOCIN нового протокола влечет за собой формирование иерархии, которая представляет его компоненты, связывание подходящих порождающих правил с новыми объектами и заполнение таблицы конечных состояний, которая определяет порядок назначения определенных компонентов курса лечения. Программа OPAL формирует элементы нового протокола в процессе "собеседования" с экспертом с помощью средств графического интерфейса. При этом полученные знания преобразуются сначала в промежуточную форму представления, а затем транслируются в формат, используемый в системе ONCOCIN. На последней стадии формируются соответствующие порождающие правила. Для упрощения реализации промежуточных стадий, трансляции и формирования порождающих правил в программе OPAL используется модель предметной области лечения онкологических заболеваний.

В модели предметной области можно выделить четыре основных аспекта, которые явились следствием применения онтологического анализа.

- *Сущности и отношения.* Сущностями в этой предметной области являются элементы (компоненты) курса лечения — назначаемые медикаменты. Эти сущности образуют часть статической онтологии предметной области. Большая часть знаний о предметной области касается атрибутов альтернативных медикаментов, например доз и их приема. Отношения между элементами курса лечения довольно запутаны в том смысле, что они связывают различные уровни спецификации в плане лечения. Так, медикаменты могут быть частью химиотерапии, а химиотерапия может быть частью протокола.

- *Действия в предметной области.* При заданных отношениях между элементами для уточнения плана приема медикаментов потребуется обращение к перечню планов. Другими словами, уточнение плана является неявным в иерархической организации сущностей предметной области. Таким образом, модель предметной области в OPAL позволяет сконцентрировать основное внимание на задачах, а не на используемых методах поиска. Однако может потребоваться изменить планы для отдельных пациентов, например изменить дозировку или заменить один препарат другим. Такие концепции, как изменение дозировки или замена препаратов в курсе лечения, образуют часть *динамической онтологии* предметной области.

- *Предикаты предметной одласти.* Этот аспект модели касается условий, при которых обращаются к модификации назначенного плана лечения. Сюда могут входить результаты лабораторных анализов и проявления у пациента определенных симптомов (например, токсикоз на определенные препараты). Такие знания образуют часть *эпистемической онтологии* предметной области, т.е. эти знания направляют и ограничивают возможные действия. На уровне реализации правила, изменяющие курс лечения, основываются на этих условиях. Такие предикаты появляются в левой части порождающих правил ONCOCIN. Подобное правило подключается к объекту в иерархи планирования таким образом, что оно применяется только в контексте определенною препарата или определенного курса химиотерапии в конкретном протоколе.

Процедурные знания. Поскольку планы курса лечения предполагают определенное расписание приема назначенных пациенту препаратов, знания о способе реализации протокола составляют существенную часть модели предметной области. Эти знаки позволяют программе OPAL извлекать информацию, которая йотом направляется в таблицы конечных состояний, описывающие возможные последовательности этапов курса терапии, и таким образом образуют другую часть *эпистемической онтологии* предметной области. На уровне реализации программа OPAL использует для описания таких процедур специальный язык программирования, который позволяет эксперту представлять достаточно сложные алгоритмы, манипулируя пиктограммами на экране дисплея.

Используя эту модель, программа OPAL может извлекать и отображать в разной форме знания о планах лечения — в виде пиктограмм, представляющих отдельные элементы плана, формуляра, заполненного информацией об отдельных препаратах, в виде предложений специального языка, представляющих процедуры, связанные с реализацией плана лечения.

Сущности и отношения между ними вводятся с помощью экранных формуляров, в которых пользователь выбирает элементы из меню. Затем заполненный формуляр преобразуется в фрейм, причем отдельные поля формуляра образуют слоты фрейма, а введенные в них значения — значения слотов *(заполнители слотов).* Эти новые объекты затем автоматически связываются с другими объектами в иерархии. Например, медикаменты связываются с объектами курсов химиотерапии, компонентами которых они являются.

Операции предметной области также вводятся с помощью заполнения экранных формуляров. В этом случае формуляр представляет собой пустой шаблон плана, в котором представлены поля для назначения расписания приема препаратов, а меню возможных действий включает такие операции, как *изменение дозировки, временное прекращение приема* и т.д. Поскольку список возможных действий довольно короткий, эта методика позволяет эксперту достаточно легко ввести нужную последовательность операции.

Процесс приобретения знании в значительной мере облегчается при использовании *языков визуального программирования.* Графический интерфейс позволяет пользователю создавать пиктограммы, представляющие элементы плана, и формировать из них графические структуры. Расставляя такие элементы на экране и вычерчивая связи между ними, пользователь формирует мнемоническую схему управления потоками, которая обычно представляется в виде программы на каком-нибудь языке программирования.

На последующих этапах такие программы преобразуются в таблицы конечных состояний, хорошо известные специалистам в области теории вычислительных машин. Для любого текущего состояния системы такая таблица позволяет определить, в какое новое состояние перейдет система, получив определенный набор входных сигналов, и какой набор выходных сигналов при этом будет сформирован.

Приобретение новых знаний на основе существующих

В ходе экспериментов по созданию интеллектуальных обучающих систем на основе технологии экспертных систем исследователи пришли к более глубокому пониманию того, какими видами знаний пользуется эксперт в процессе решения проблем. При создании инструментальных средств общего назначения, аналогичных EMYCIN и предназначенных для построения широкого класса экспертных систем, разработчики столкнулись с интересной проблемой: как преобразовать знания, имеющие отношение к любой проблемной области, во фреймы или порождающие правила.

Такие попытки заставили исследователей глубже проанализировать роль знаний о предметной области и специфических для нее правил логического вывода, в частности рассмотреть их с точки зрения разных стилей рассуждения, характерных для разных областей.

Программа извлечения знаний нуждается в некоторых базовых знаниях о той предметной области, в которой специализируется интервьюируемый эксперт. И точно такими же знаниями должен обладать инженер по знаниям. Только в этом случае он сможет достичь взаимопонимания и диалоге с экспертом.

Вряд ли стоит надеялся на то, что со временем появится такая методика извлечения знаний у эксперта, которая будет одинаково эффективна в любой предметной области. Знания, которыми нужно обладать для того, чтобы воспринимать новые знания, можно рассматривать как метазнания. В основном к ним относятся знания о структуре и стратегии, включая информацию о методах классификации явлений и сущностей в определенной предметной области (например, заболеваний) и способах выбора альтернативных действий (например, курсов терапии). Существуют также и отдельные знания, необходимые для того, чтобы объяснить, почему получено именно такое, а не иное решение проблемы.

Извлечение знаний посредством опроса экспертов на основе модели предметной области — отнюдь не последнее слово в автоматизации этого процесса.

Лекция № 4 Тема: Классификация задач экспертных систем

В сборнике статей, опубликованном под общей редакцией Хейеса-Рота *[1983],* была предложена классификация экспертных систем, которая отражает специфику задач, решаемых с помощью этой технологии. С тех пор эта классификация неоднократно критиковалась различными авторами, в основном из-за того, что в ней были смешаны разные характеристики, а это привело к тому, что сформулированные категории нельзя рассматривать как взаимно исключающие. Тем не менее мы кратко представим эту классификацию и будем рассматривать ее как отправную точку для дальнейшего совершенствования.

* *Интерпретирующие системы* предназначены для формирования описания ситуаций по результатам наблюдений или данным, получаемым от различного рода сенсоров. Типичные задачи, решаемые с помощью интерпретирующих систем,— распознавание образов и определение химической структуры вещества.

* *Прогнозирующие системы* предназначены для логического анализа возможных последствий заданных ситуаций или событий. Типичные задачи для экспертных систем этого типа — предсказание погоды и прогноз ситуаций на финансовых рынках.

* *Диагностические системы* предназначены для обнаружения источников неисправностей по результатам наблюдений за поведением контролируемой системы (технической или биологической). В эту категорию входит широкий спектр задач в самых различных предметных областях — медицине, механике, электронике и т.д.

* *Системы проектирования* предназначены для структурного синтеза конфигурации объектов (компонентов проектируемой системы) при заданных ограничениях. Типичными задачами для таких систем является синтез электронных схем, компоновка архитектурных планов, оптимальное размещение объектов в ограниченном пространстве.

* *Системы планирования* предназначены для подготовки планов проведения последовательности операций, приводящей к заданной цели. К этой категории относятся задачи планирования поведения роботов и составление маршрутов передвижения транспорта.

* *Системы мониторинга* анализируют поведение контролируемой системы и, сравнивая полученные данные с критическими точками заранее составленного плана, прогнозируют вероятность достижения поставленной цели. Типовые области приложения таких систем — контроль движения воздушного транспорта и наблюдение за состоянием энергетических объектов.

* *Наладочные системы* предназначены для выработки рекомендаций по устранению неисправностей в контролируемой системе. К этому классу относятся системы, помогающие программистам в отладке программного обеспечения, и консультирующие системы.

* *Системы оказания помощи при ремонте* оборудования выполняют планирование процесса устранения неисправностей в сложных объектах, например в сетях инженерных коммуникаций.

* *Обучающие системы* проводят анализ знаний студентов по определенному предмету, отыскивают пробелы в знаниях и предлагают средства для их ликвидации.

* *Системы контроля* обеспечивают адаптивное управление поведением сложных человеко-машинных систем, прогнозируя появление возможных сбоев и планируя действия, необходимые для их предупреждения. Областью применения таких систем является управление воздушным транспортом, военными действиями и деловой активностью в сфере бизнеса.

Как уже упоминалось, множество исследователей отмечали наличие ряда существенных недостатков в приведенной классификации. Рейхгелт и Ван Гармелен обратили внимание на то, что некоторые из категорий в ней перекрываются или включают друг друга *[1986]*. Например, категорию *системы планирования* в этой классификации вполне можно рассматривать как составную часть категории *системы проектирования,* поскольку планирование можно трактовать как *проектирование последовательности операций* (на это, кстати, обратили внимание и авторы классификации *[1983]).* Кленси также задался вопросом: "Является ли автоматизация программирования проблемой *планирования* или *проектирования!" [1985]*. Совершенно очевидно, что подобное замечание можно высказать и по отношению к таким категориям, как *диагностические системы, системы мониторинга, системы оказания помощи при ремонте* и *обучающие системы*.

Кленси предложил альтернативный метод классификации, взяв за основу набор *родовых* операций, выполняемых в рассматриваемых системах. Вместо того чтобы пытаться разделить анализируемые программы решения проблем по признакам особенностей тех проблем, на решение которых они ориентированы, он предложил поставить во главу угла те виды операций, которые выполняются по отношению к реальной обслуживаемой системе (механической, биологической или электрической).

Кленси предложил разделять *синтетические* операции, результатом которых является изменение структуры *(конструкции)* системы, и *аналитические* операции, которые *интерпретируют* характеристики и свойства системы, не изменяя ее как таковую. Эта обобщенная концепция может быть конкретизирована, в результате чего построена иерархическая схема видов операций, выполнение которых может быть затребовано от программы. На рис. 1 и 2 представлены такие иерархические схемы для аналитических и синтетических операций.

Рис. 1. Иерархия родовых аналитических операций

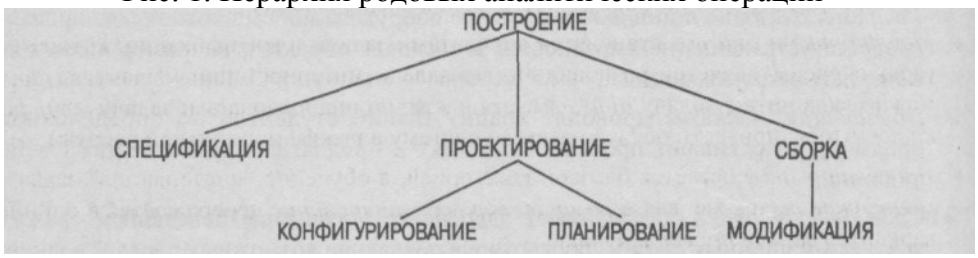


Рис. 2. Иерархия родовых синтетических операций

Классификация методов решения проблем

Классификация — это одна из наиболее распространенных проблем в любой предметной области. Например, эксперты в области ботаники или зоологии первым делом пытаются определить место в существующей таксономии для вновь открытого растения или животного. Как правило, система классов имеет явно выраженную иерархическую организацию, в которой подклассы обладают определенными свойствами, характерными для своих суперклассов, причем классы-соседи на одном уровне иерархии являются взаимно исключающими в отношении наличия или отсутствия определенных наборов свойств.

1. Эвристическое сопоставление

Кленси отметил, что одна из важнейших особенностей классификации состоит в том, что эксперт *выбирает* категорию из ряда возможных решений, которые можно заранее перечислить. Когда мы имеем дело с простыми вещами или явлениями, то для их классификации вполне достаточно бросающихся в глаза свойств объектов. Это позволяет почти мгновенно сопоставлять данные и категории. В более сложных случаях таких лежащих на поверхности свойств может оказаться недостаточно для того, чтобы правильно определить место объекта в иерархической схеме классификации. В этом случае нам остается уповать на тот метод, который Кленси назвал *эвристической классификацией.* Суть его состоит в установлении неиерархических ассоциативных связей между данными и категориями классификации, которое требует выполнения промежуточных логических заключений, включающих, возможно, и концепции из *другой* таксономии.

На рис. 3 показаны три основных этапа выполнения эвристической классификации: абстрагирование от данных, сопоставление абстрактных категорий данных с абстрактными категориями решений (утолщенная стрелка) и конкретизация решения. Рассмотрим их по очереди.

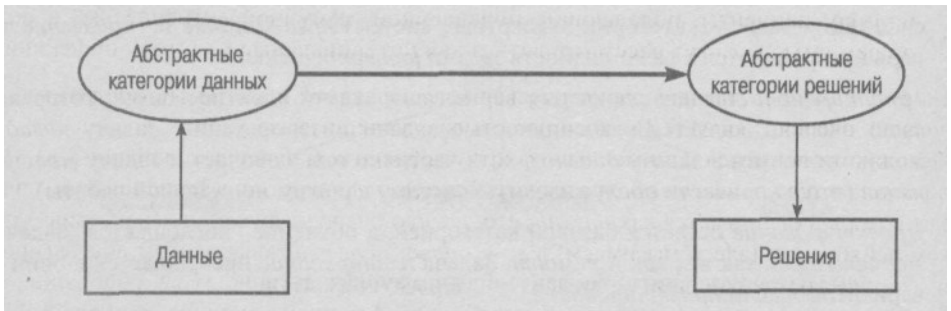


Рис. 3. Структура логических связей при эвристической классификации

* *Абстрагирование от данных.* Часто бывает полезно абстрагироваться от данных, характеризующих конкретный случай. Так, при диагностировании заболевания за частую важно не столько то, что у пациента высокая температура (скажем, 39.8°), а то, что она

выше нормальной. То есть врач обычно рассуждает в терминах *диапазона* температур, а не в терминах *конкретного ее значения.*

* *Эвристическое сопоставление.* Выполнить сопоставление первичных данных в конкретном случае и окончательного диагноза довольно трудно. Гораздо легче сопоставить более абстрактные данные и достаточно широкий класс заболеваний. Например, повышенная температура может служить индикатором лихорадки, наводящей на мысль о инфекционном заражении. Данные "включают" гипотезы, но на относительно высоком уровне абстракции. Такой процесс сопоставления имеет ярко выраженный эвристический характер, поскольку соответствие между данными и гипотезами на любом уровне не бывает однозначным и из общего правила может быть множество исключений. Анализ данных, которые "вписываются" в определенную абстрактную категорию, просто позволяет отбирать решения, лучше согласующиеся с абстрактами решений.

* *Конкретизация решений.* После того как определена абстрактная категория, которая сужает пространство решений, нужно определить в этом пространстве конкретные решения-кандидаты и каким-то образом их ранжировать. Это может потребовать дальнейших размышлений, в которые включаются уже количественные параметры данных, или даже сбора дополнительной информации. В любом случае целью этой процедуры является отбор "соревнующихся" гипотез в пространстве решений и последующее их ранжирование — сортировка по степени правдоподобия.

Кленси различает три варианта построения абстрактных категорий данных.

* *Определительный.* В этом варианте в первую очередь рассматриваются характерные признаки класса объектов, и он во многом напоминает таксономический подход в ботанике и зоологии.

* *Количественный.* В этом варианте абстрагирование выполняется исходя из количественных характеристик, как это было сделано в упоминавшемся выше примере с температурой пациента.

* *Обобщение.* Этот вариант основывается на иерархии характерных свойств. Например, пациенты, обладающие подавленной иммунной активностью, в более общем смысле могут рассматриваться как потенциальные носители инфекции.

На рис. 4 представлена эвристическая классификация в контексте программы медицинской диагностики MYCIN.

Исходными являются данные анализа крови пациента (количество лейкоцитов). Сначала выполняется *количественное* абстрагирование от конкретного значения этого показателя, который оценивается как *низкий,* что, в свою очередь, является характерным признаком лейкопении (здесь мы имеем дело с *определительным* вариантом абстрагирования). *Обобщение* лейкопении — подавленная иммунная активность, а обобщение последней— повышенная склонность к переносу инфекции (т.е. такие пациенты более подвержены воздействию различных микроорганизмов). Повышенная склонность к переносу инфекции является уже *родовой* категорией и наводит на мысль о наличии инфекции, вызванной грамотрицательными микроорганизмами (т.е. инфекции, связанной с *определенным классом* бактерий). Затем это родовое решение *конкретизируется* и предполагается, что источником инфекции являются бактерии E.Coli.

В системе MYCIN сопоставление данных и абстрактных категорий решений выполняется с помощью порождающих правил, а эвристическая природа такого сопоставления выражается коэффициентами уверенности. Эти коэффициенты можно рассматривать как заложенную в порождающее правило меру "строгости" соответствия между предпосылкой и выводом. Другие правила затем будут уточнять выполненное сопоставление и таким образом "подстраивать" коэффициент уверенности.

2. Общность эвристической классификации

Интересной особенностью методики анализа, предложенной Кленси, является то, что она подходит для большого спектра экспертных систем. Оказывается, что, несмотря на различие областей применения, многие экспертные системы функционируют, в принципе,

одинаково. В своей статье Кленси продемонстрировал применение предложенной методики на множестве систем, помимо MYCIN.

Нужно отметить, что предложенная Кленси методика анализа фундаментальных аспектов решения проблем с помощью технологии экспертных систем не является единственной. Например, получило дальнейшее развитие понятие *родовой задачи.* Родовая задача представляет собой, по сути, спецификацию задачи, включающую описание различных форм знаний о предметной области и их организации для выполнения задачи вручную и набор режимов выполнения задачи.

Примерами таких задач являются иерархическая классификация, сопоставление или оценка гипотез и передача информации, направляемая знаниями.

Задача *иерархической классификации* включает отбор гипотез из иерархически организованного пространства альтернатив, а затем уточнение этих гипотез с учетом имеющихся данных. Гипотезами могут быть, например, заболевания или неисправности оборудования. Каждая такая гипотеза может быть порождена имеющимися данными, но прежде чем развивать ее дальше, необходимо выполнить определенные проверки или уточнить гипотезу таким образом, чтобы она соответствовала более широкому набору имеющихся фактов.

Под *сопоставлением гипотез* понимается выполнение количественной оценки свидетельств в пользу достоверности гипотез или того, насколько полным является соответствие между имеющимися данными и гипотезами. В такую оценку иногда включается априорная вероятность гипотез, поиск возможных конкурирующих гипотез и т.п. Все операции, связанные с обработкой коэффициентов уверенности в системе MYCIN, можно рассматривать как реализацию механизма родовых задач этого вида.

Передача информации, направляемой знаниями, проявляется в виде неочевидных логических связей, которые трудно классифицировать, но которые характерны для способа мышления эксперта. Например, врач не оставит без внимания тот факт, что пациент недавно перенес хирургическую операцию, поскольку, скорее всего, он подвергся анестезии, а это может косвенно сказаться на формировании диагноза. В логических рассуждениях такого типа используются "фоновые" знания о предметной области, а не те порождающие правила, с которыми мы связываем свое представление об экспертной системе медицинской диагностики.

Итак, одно из отличий между подходами Кленси и Чандрасекарана состоит в том, что для первого характерно стремление разбить решение проблемы на мелкие абстрактные категории. Подход Кленси приводит к поглощению этапа сопоставления гипотез эвристической классификацией, в то время как Чандрасекаран делает акцент на том, что обработка гипотез может рассматриваться как самостоятельная родовая задача со своими собственными правами вне контекста классификации. Таким образом, появляется проблема, до какого уровня "зернистости" целесообразно доводить абстрагирование и какие критерии следует выбрать для оценки этой "грануляции".

Лекция № 5

Тема: Назначение и особенности методов искусственного интеллекта для разработки экспертных систем

Знания, которыми обладает специалист в какой-либо области (дисциплине), можно разделить на формализованные (точные) и неформализованные (неточные). *Формализованные знания* формулируются в книгах и руководствах в виде общих и строгих суждений (законов, формул, моделей, алгоритмов и т. п.), отражающих универсальные знания. *Неформализованные знания,* как правило, не попадают в книги и руководства в связи с их конкретностью, субъективностью и приблизительностью. Знания этого рода являются результатом обобщения многолетнего опыта работы и интуиции специалистов. Они обычно представляют собой многообразие эмпирических (эвристических) приемов и правил.

В зависимости от того, какие знания преобладают в той или иной области (дисциплине), ее относят к формализованным (если преобладают точные знания) или к неформализованным (если преобладают неточные знания) описательным областям. *Задачи,* решаемые на ос-

нове точных знаний, называют *формализованными,* а задачи, решаемые с помощью неточных знаний - *неформализованными.* (Речь идет не о неформализуемых, а о неформализованных задачах, т. е. о задачах, которые, возможно, и формализуемы, но эта формализация пока неизвестна).

Традиционное программирование в качестве основы для разработки программы использует алгоритм, т. е. формализованное знание. Поэтому до недавнего времени считалось, что ЭВМ не приспособлены для решения неформализованных задач. Расширение сферы использования ЭВМ показало, что неформализованные задачи составляют очень важный класс задач, вероятно значительно больший, чем класс формализованных задач Неумение решать неформализованные задачи сдерживает внедрение ЭВМ в описательные науки. Основной задачей информатики является внедрение ее методов в описательные науки и дисциплины. На основании этого можно утверждать, что исследования в области ЭС занимают значительное место в информатике.

Особенности неформализованных задач:

- алгоритмическое решение задачи неизвестно (хотя, возможно, и существует) или не может быть использовано из-за ограниченности ресурсов ЭВМ (времени, памяти);
- задача не может быть определена в числовой форме (требуется символьное представление);
- цели задачи не могут быть выражены в терминах точно определенной целевой функции.

Как правило, неформализованные задачи обладают неполнотой, ошибочностью, неоднозначностью и (или) противоречивостью знаний (как данных, так и используемых правил преобразования)

Экспертные системы не отвергают и не заменяют традиционного подхода к программированию, они отличаются от традиционных программ тем, что ориентированы на решение неформализованных задач и обладают следующими особенностями:

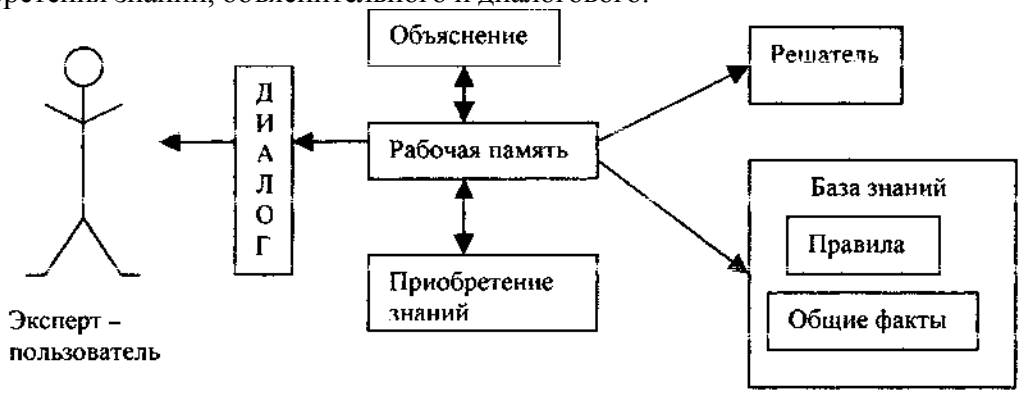
- алгоритм решений не известен заранее, а строится самой ЭС с помощью символических рассуждений, базирующихся на эвристических приемах;
- ясность полученных решений, т. е- система «осознает» в терминах пользователя, как она получила решение;
- способность анализа и объяснения своих действий и знаний;
- способность приобретения новых знаний от пользователя-эксперта, не знающего программирования, и изменения в соответствии с ними своего поведения;
- обеспечение «дружественного», как правило, естественно-языкового (ЕЯ) интерфейса с пользователем.

Обычно к ЭС относят *системы, основанные на знании,* т. е. системы, вычислительная возможность которых является в первую очередь следствием их наращиваемой базы знаний (БЗ) и только во вторую очередь определяется используемыми методами. *Методы инженерии знаний* (методы ЭС) в значительной степени инвариантны тому, в каких областях они могут применяться. Области применения ЭС весьма разнообразны: военные приложения, медицина, электроника, вычислительная техника, геология, математика, космос, сельское хозяйство, управление, финансы, юриспруденция и т. д. Более критичны методы инженерии знаний к типу решаемых задач. В настоящее время ЭС используются при решении задач следующих типов:

- принятие решений в условиях неопределенности (неполноты);
- интерпретация символов и сигналов;
- предсказание;
- диагностика;
- конструирование;
- планирование;
- управление;
- контроль.

Структура и режимы экспертных систем.

Типичная ЭС состоит из следующих основных компонентов: решателя (интерпретатора), рабочей памяти (РП), называемой также базой данных (БД), базы знаний (БЗ), компонентов приобретения знаний, объяснительного и диалогового.



База данных предназначена для хранения исходных и промежуточных данных решаемой в текущий момент задачи. Этот термин совпадает по названию, но не по смыслу с термином, используемым в информационно-поисковых системах (ИПС) и системах управления базами данных (СУБД) для обозначения всех данных (и в первую очередь не текущих, а долгосрочных), хранимых в системе.

База знаний в ЭС предназначена для хранения долгосрочных данных, описывающих рассматриваемую область (а не текущих данных), и правил, описывающих целесообразные преобразования данных этой области.

Решатель, используя исходные данные из РП и знания из БЗ, формирует такую последовательность правил, которые, будучи примененными к исходным данным, приводят к решению задачи.

Компонент приобретения знаний автоматизирует процесс наполнения ЭС знаниями, осуществляемый пользователем-экспертом.

Объяснительный компонент объясняет, как система получила решение задачи (или почему она не получила решения) и какие знания она при этом использовала, что облегчает эксперту тестирование системы и повышает доверие пользователя к полученному результату,

Диалоговый компонент ориентирован на организацию дружелюбного общения со всеми категориями пользователей, как в ходе решения задач, так и приобретения знаний, объяснения результатов работы

В разработке ЭС участвуют представители следующих специальностей:

- эксперт в той проблемной области, задачи которой будет решать ЭС;
- инженер по знаниям – специалист по разработке ЭС;
- программист - специалист по разработке инструментальных средств (ИС).

Необходимо отметить, что отсутствие среди участников разработки инженера по знаниям (т. е. его замена программистом) либо приводит к неудаче процесс создания ЭС, либо значительно удлиняет его.

Эксперт определяет знания (данные и правила), характеризующие проблемную область, обеспечивает полноту и правильность введенных в ЭС знаний.

Инженер по знаниям помогает эксперту выявить и структурировать знания, необходимые для работы ЭС, осуществляет выбор того инструментального средства, которое наиболее подходит для данной проблемной области и определяет способ представления знаний в этом инструментальном средстве, выделяет и программирует (традиционными средствами) стандартные функции (типичные для данной проблемной области), которые будут использоваться в правилах, вводимых экспертом.

Программист разрабатывает ИС, содержащее в пределе все основные компоненты ЭС, осуществляет сопряжение ИС с той средой, в которой оно будет использовано.

Экспертная система работает в двух режимах: приобретения знаний и решения задач (режим консультации или режим использования ЭС).

В *режиме приобретения знаний* общение с ЭС осуществляет через посредничество инженера по знаниям эксперт. Эксперт описывает проблемную область в виде совокупности данных и правил. Данные определяют объекты, их характеристики и значения, существующие в области экспертизы. Правила определяют способы манипулирования данными, характерные для рассматриваемой проблемной области. Эксперт, используя компонент приобретения знаний, наполняет систему знаниями, которые позволяют ЭС в режиме решения самостоятельно (без эксперта) решать задачи из проблемной области.

Важную роль в режиме приобретения знаний играет объяснительный компонент. Именно благодаря ему эксперт на этапе тестирования локализует причины неудачной работы ЭС, что позволяет эксперту целенаправленно модифицировать старые или вводить новые знания. Обычно объяснительный компонент сообщает следующее: как правила используют информацию пользователя; почему использовались или не использовались данные или правила; какие были сделаны выводы и т. п. Все объяснения делаются, как правило, на ограниченном естественном языке или языке графики.

Отметим, что режиму приобретения знаний при традиционном подходе к разработке программ соответствуют этапы алгоритмизации, программирования и отладки, выполняемые программистом. Таким образом, в отличие от традиционного подхода разработку программ осуществляет эксперт (с помощью ЭС), не владеющий программированием, а не программист.

В *режиме консультации* общение с ЭС осуществляет конечный пользователь, которого интересует результат и (или) способ получения решения. Пользователь в зависимости от назначения ЭС может не быть специалистом в данной проблемной области, в этом случае он обращается к ЭС за советом, не умея получить ответ сам, или быть специалистом, в этом случае он обращается к ЭС, чтобы либо ускорить процесс получения результата, либо возложить на ЭС рутинную работу. Термин «пользователь» является многозначным, так как кроме конечного пользователя применять ЭС может и эксперт, и инженер по знаниям, и программист. Поэтому, когда хотят подчеркнуть, что речь идет о том, для кого делалась ЭС, используют термин «конечный пользователь».

В режиме консультации данные о задаче пользователя обрабатываются диалоговым компонентом, который выполняет следующие действия:

- распределяет роли участников (пользователя и ЭС) и организует их взаимодействие в процессе кооперативного решения задачи;
- преобразует данные пользователя о задаче, представленные на привычном для пользователя языке, во внутренний язык системы;
- преобразует сообщения системы, представленные на внутреннем языке, в сообщения на языке, привычном для пользователя (обычно это ограниченный естественный язык или язык графики).

После обработки данные поступают в РП. На основе входных данных из РП, общих данных о проблемной области и правил из БЗ решатель (интерпретатор) формирует решение задачи.

В отличие от традиционных программ ЭС в режиме решения задачи не только исполняет предписанную последовательность операций, но и предварительно формирует ее. Если ответ ЭС не понятен пользователю, то он может потребовать объяснения, как ответ получен.

ХАРАКТЕРИСТИКИ ЭС.

Экспертные системы как любой сложный объект можно определить только совокупностью характеристик

Выделим следующие характеристики ЭС:

1) назначение;
2) проблемная область;
3) глубина анализа проблемной области;
4) тип используемых методов и знаний;
5) класс системы;

6) стадия существования;

7) инструментальные средства (ИС).

Перечисленный набор характеристик не претендует на полноту (в связи с отсутствием общепринятой классификации), а определяет ЭС как целое, не выделяя отдельных компонентов (способ представления знаний, решения задачи и т. п.).

Назначение определяется следующей совокупностью параметров: цель создания ЭС—для обучения специалистов, для решения задач, для автоматизации рутинных работ, для тиражирования знаний экспертов и т. п.; основной пользователь—не специалист в области экспертизы, специалист, учащийся.

Проблемная область может быть определена совокупностью параметров: предметной областью и задачами, решаемыми в предметной области, каждый из которых может рассматриваться с точки зрения как конечного пользователя, так и разработчика ЭС.

С точки зрения пользователя, предметную область можно характеризовать описанием области в терминах пользователя, включающим наименование области, перечень и взаимоотношение подобластей и т. п., а задачи, решаемые существующими ЭС, — их типом. Обычно выделяют следующие *типы задач:*

интерпретация символов или сигналов — составление смыслового описания по входным данным;

предсказание—определение последствий наблюдаемых ситуаций;

диагностика—определение неисправностей (заболеваний) по симптомам;

конструирование—разработка объекта с заданными свойствами при соблюдении установленных ограничений:

планирование — определение последовательности действий, приводящих к желаемому состоянию объекта;

слежение — наблюдение за изменяющимся состоянием объекта и сравнение его показателей с установленными или желаемыми;

управление—воздействие на объект для достижения желаемого поведения.

С точки зрения разработчика целесообразно выделять *статические* и *динамические предметные области.* Предметная область называется статической, если описывающие ее исходные данные не изменяются во времени (точнее, рассматриваются как не изменяющиеся за время решения задачи). Статичность области означает неизменность описывающих ее исходных данных. При этом производные данные (выводимые из исходных) могут и появляться заново, и изменяться (не изменяя, однако, исходных данных). Если исходные данные, описывающие предметную область, изменяются за время решения задачи, то предметную область называют динамической. Кроме того, предметные области можно характеризовать следующими аспектами: числом и сложностью сущностей; их атрибутов и значений атрибутов; связностью сущностей и их атрибутов; полнотой знаний; точностью знаний (знания точны или правдоподобны: правдоподобность знаний представляется некоторым числом или высказыванием).

Решаемые задачи, с точки зрения разработчика ЭС, также можно разделить на статические и динамические. Будем говорить, что ЭС решает *динамическую* или *статическую* задачу, если процесс решения задачи изменяет или не изменяет исходные данные о текущем состоянии предметной области.

В подавляющем большинстве существующих ЭС исходят из предположения статичности предметной области и решают статические задачи, будем называть такие ЭС *статическими.* ЭС, которые имеют дело с динамическими предметными областями и решают статические или динамические задачи, будем называть *динамическими. В* настоящее время начинают появляться первые динамические ЭС. Решение многих важнейших практических неформализованных задач возможно *только с помощью динамических, я не статических* ЭС. Следует подчеркнуть, что на традиционных (числовых) последовательных ЭВМ с помощью существующих методов инженерии знаний можно решать только статические задачи, а для решения динамиче-

ских задач, составляющих большинство реальных приложений, необходимо использовать специализированные символьные ЭВМ.

Решаемые задачи, кроме того, можно характеризовать следующими аспектами:

числом и сложностью правил, используемых в задаче;

связностью правил;

пространством поиска;

количеством активных агентов, изменяющих предметную область;

классом решаемых задач.

По степени сложности выделяют *простые* и *сложные правила*. К сложным относят правила, текст записи которых на естественном языке занимает 1/3 страницы и больше. Правила, текст которых занимает менее 1/3 страницы, относят к простым.

По степени связности правил задачи делят на связные и малосвязные. К связным относятся задачи (подзадачи), которые не удается разбить на независимые задачи. Малосвязные задачи удается разбить на некоторое количество независимых подзадач.

Степень сложности задачи определяется не просто общим количеством правил данной задачи, а количеством правил в ее наиболее связанной независимой подзадаче.

Пространство поиска может быть определено по крайней мере тремя подаспектами: размером, глубиной и шириной. Размер пространства поиска дает обобщенную характеристику сложности задачи. Выделяют малые (до 10! состояний) и большие (свыше 10! состояний) пространства поиска. Глубина пространства поиска характеризуется средним числом последовательно применяемых правил, преобразующих исходные данные в конечный результат, ширина пространства—средним числом правил, пригодных к выполнению в текущем состоянии.

Количество активных агентов существенно влияет на выбор метода решения. Выделяют следующие значения данного аспекта: ни одного агента, один агент, несколько агентов.

Класс решаемых задач характеризует методы, используемые ЭС для решения задачи. Данный аспект в существующих ЭС принимает следующие значения: задачи расширения, доопределения, преобразования. Задачи расширения и доопределения являются статическими, а задачи преобразования—динамическими.

К задачам расширения относятся задачи, в процессе решения которых осуществляется только увеличение информации о предметной области, не приводящее ни к изменению ранее введённых данных, ни к выбору другого состояния области. Типичной задачей этого класса являются задачи классификации.

К задачам доопределения относятся задачи с неполной или неточной информацией о реальной предметной области, цель решения которых—выбор из множества альтернативных текущих состояний предметной области того, которое адекватно исходным данным. В случае неточных данных альтернативные текущие состояния возникают как результат ненадежности данных и правил, что приводит к многообразию различных доступных выводов из одних и тех же исходных данных. В случае неполных данных альтернативные состояния являются результатом доопределения области, т. е. результатом предположений о возможных значениях недостающих данных.

К задачам преобразования относятся задачи, которые осуществляют изменения исходной или введённой ранее информации о предметной области, являющиеся следствием изменений либо реального мира, либо его модели.

Большинство существующих ЭС решают задачи расширения, в которых нет ни изменений предметной области, ни активных агентов, преобразующих предметную область. Подобное ограничение неприемлемо при работе в динамических областях.

По степени сложности структуры ЭС делят на поверхностные и глубинные. *Поверхностные ЭС* представляют знания об области экспертизы в виде правил (условие $\rightarrow$ действие). Условие каждого правила определяет образец некоторой ситуации, при соблюдении которой правило может быть выполнено. Поиск решения состоит в выполнении тех правил, образцы которых сопоставляются с текущими данными (текущей ситуацией в РП). При этом предполагается,

что в процессе поиска решения последовательность формируемых таким образом ситуаций не оборвется до получения решения, т. е. не возникнет неизвестной ситуации, которая не сопоставится ни с одним правилом. *Глубинные* ЭС, кроме возможностей поверхностных систем, обладают способностью при возникновении неизвестной ситуации определять с помощью некоторых общих принципов, справедливых для области экспертизы, какие действия следует выполнить.

По типу используемых методов и знаний ЭС делят на традиционные и гибридные. *Традиционные ЭС* используют в основном неформализованные методы инженерии знаний и неформализованные знания, полученные от экспертов. *Гибридные* ЭС используют и методы инженерии знаний, и формализованные методы, а также данные традиционного программирования и математики.

Совокупность рассматриваемых выше характеристик позволяет определить особенности конкретной ЭС. Однако пользователи зачастую стремятся охарактеризовать ЭС каким-либо одним обобщенным параметром. В этой связи говорят о поколениях ЭС. В настоящее время выделяют ЭС первого и второго поколения. По нашему мнению, целесообразно говорить о трех поколениях ЭС. К *первому поколению* следует относить статические поверхностные ЭС, ко *второму* - статические глубинные ЭС (иногда ко второму поколению относят гибридные ЭС), а к *третьему*— динамические ЭС (вероятно, они, как правило, будут глубинными и гибридными).

В последнее время выделяют два больших класса ЭС (существенно отличающихся по технологии их проектирования), которые условно называют простыми и сложными ЭС. *Простая ЭС* может быть охарактеризована следующими значениями основных параметров:

- поверхностная ЭС;
- традиционная ЭС (реже гибридная);
- выполненная на персональной ЭВМ (ПЭВМ);
- коммерческая стоимостью от 100 до 25 тыс. дол.;
- стоимость разработки от 50 тыс. до 300 тыс. дол.;
- время разработки от 3 мес. до одного года (при использовании развитых инструментальных средств);
- от 200 до 1000 правил.

Сложная ЭС может быть охарактеризована следующими значениями параметров:

- глубинная ЭС;
- гибридная ЭС;
- выполненная либо на символьной ЭВМ, либо на мощной универсальной ЭВМ, либо на интеллектуальной рабочей станции;
- коммерческая стоимость от 50 тыс. до 1 млн. дол.;
- средняя стоимость разработки 5-10 млн. дол.;
- время разработки от 1 до 5 лет;
- от 1500 до 10 тыс. правил.

Стадия существования характеризует степень проработанности и отлаженности ЭС. Обычно выделяют следующие стадии:

- демонстрационный прототип;
- исследовательский прототип;
- действующий прототип;
- промышленная система;
- коммерческая система.

Демонстрационным прототипом называют ЭС, которая решает часть требуемых задач, демонстрируя жизнеспособность метода инженерии знаний. При наличии развитых ИС для разработки демонстрационного прототипа требуется в среднем примерно 1—2 мес., а при отсут-

ствии—12—18 мес. Демонстрационный прототип работает, имея в БЗ 50—100 правил. Развитие демонстрационного прототипа приводит к исследовательскому прототипу.

Исследовательским прототипом называют систему, которая решает все требуемые задачи, но неустойчива в работе и не полностью проверена. На доведение системы до стадии исследовательского прототипа уходит 3—6 мес. Исследовательский прототип обычно имеет в БЗ 200—500 правил, описывающих проблемную область.

Действующий прототип надежно решает все задачи, но для решения сложных задач может потребоваться чрезмерно много времени и (или) памяти. Для доведения системы до стадии действующего прототипа требуется 6— 12 мес., при этом количество правил в БЗ увеличивается до 500—1000.

Экспертная система, достигшая *промышленной стадии,* обеспечивает высокое качество решений всех задач при минимуме времени и памяти. Обычно процесс преобразования действующего прототипа в промышленную систему состоит в расширении БЗ (до 1000—1500 правил) и переписывании программ с использованием более эффективных ИС, например в перепрограммировании на языках низкого уровня. Для доведения ЭС от начала разработки до стадии промышленной системы требуется 1—1,5 года.

Обобщение задач, решаемых ЭС на стадии промышленной системы, позволяет перейти к стадии *коммерческой системы* — к системе, пригодной не только для собственного использования, но и для продажи различным потребителям. Для доведения системы до коммерческой стадии требуется 1,5—3 года и 0.3-5 млн. дол. При этом в БЗ системы 1500—3000 правил.

Инструментальные средства определяют *программные* и *аппаратные* средства, используемые в рассматриваемой ЭС.

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ПРАКТИЧЕСКИМ ЗАНЯТИЯМ

Практическая работа № 1

Представление знаний и получение выводов С помощью логики предикатов

Предикатом [2], или *логической функцией,* называется функция от любого числа аргументов, принимающая истинностные значения: истинно (1) и ложно (0). Аргументы принимают значения из произвольного, конечного или бесконечного множества D, называемого предметной областью. Предикат от *n* аргументов называют *n*-местным предикатом.

Предикат F(x), определенный на множестве D, задает определенное свойство элементам множества D и интерпретируется как высказывание “x обладает свойством F”, причем F принимает значение “истинно”, если это высказывание истинно, и значение “ложно”, если оно ложно. Предикат $F(x_1,x_2,\ldots,x_n)$ задает отношение между элементами $x_1,x_2,\ldots,x_n$ и интерпретирует как высказывание “ $x_1,x_2,\ldots,x_n$ находятся между собой в отношении F”.

Например: D – множество натуральных чисел. F(x) может означать, что x – четное или x – нечетное. G(x,y) - x > y или x делится на y.

Алфавит языка предикатов первого порядка состоит из:

1) разделители: запятая, открывающая и закрывающая скобки;

2) константы, обозначаемые строчными буквами или соединением таких букв, – например: a, st, друг;

3) переменные, обозначаемые прописными буквами, – например: X, ST, АДРЕС;

4) предикаты, обозначаемые прописными буквами, – например: P, Q, БОЛЬШЕ;

5) функции, устанавливающие зависимость и отображающие значения одной предметной области в значения другой (или той же), n-местные функции могут служить аргументами предиката. Функции будем обозначать строчными буквами: f,g,t;

6) логические функции:

¬ (отрицание или дополнение). Высказывание “¬A” читается “не A”. Оно истинно (И), если высказывание A – ложно (Л);

$\wedge$ (конъюнкция). Высказывание “A$\wedge$B” читается “A и B”. Оно истинно в том случае, когда истинно как A, так и B;

$\vee$ (дизъюнкция). Высказывание “A$\vee$B” читается “A или B”. Оно истинно, если истинно хотя бы одно из высказываний;

$\rightarrow$ (импликация). Высказывание “A$\rightarrow$B” читается “если A, то B”. Оно ложно в том и только в том случае, если A истинно, а B ложно;

$\leftrightarrow$ (эквивалентность). Высказывание “A$\leftrightarrow$B” читается “A тогда и только тогда, когда B”. Оно истинно в тогда и только тогда, когда A и B имеют одно и тоже истинностное значение;

$\exists$ (квантор существования). Высказывание “$\exists$A” читается “существует A”;

$\forall$ (квантор общности). Высказывание “$\forall$A” читается “для любого A”.

Пропозициональной формой [2], или *формулой алгебры логики*, называют всякое высказывание, составленное из некоторых исходных высказываний посредством логических операций, т. е. если F и G – пропозициональные формы, то $\neg$F, (F$\vee$G), (F$\wedge$G), (F$\rightarrow$G), (F$\leftrightarrow$G) – пропозициональные формы.

Пропозициональной функцией [5] называется выражение, содержащее переменную и превращающееся в истинное или ложное высказывание при подстановке вместо переменной имени предмета из определенной предметной области.

Пропозициональные функции делятся на одноместные, содержащие одну переменную, называемые свойствами (например, «x – композитор», «x-7=3»), и содержащие две и более переменных, называемые отношениями (например, «x-c=16», «объем куба x равен объему куба y).

Определение формулы – основного объекта логики предикатов – включает понятие “терм”.

Терм – выражение, включающее константы, переменные или n-местные функции $f(t_1,t_2,\ldots,t_n)$, где $t_1,t_2,\ldots,t_n$ – термы.

Например: f(X,Y), f(b,вес(Z)) – термы, P(X,голубой), вес(P(b)) – не термы, т.к. P – предикат.

Атом, или *элементарная (атомная) формула* – это выражение, включающее константы, переменные, функции и предикаты. Таким образом, если P – n-местный предикат, а $t_1,t_2,\ldots,t_n$ – термы, то $P(t_1,t_2,\ldots,t_n)$ – атом.

Например: P(X,голубой), ВХОД(стол, X, под(окно)) – являются атомами, f(X,Y) – не атом.

Формула или правильно построенная формула (ППФ) определяется следующим образом: всякий атом есть ППФ.

Если F и G – ППФ, а X – переменная, тогда $\neg$H, (G$\vee$H), (G$\wedge$H), ($\exists$X)G, ($\forall$X)H – ППФ.

Выражение “первого порядка” во фразе “исчисление предикатов первого порядка” связано с определением ППФ, в которых запрещается квантифицировать символы предикатов и функций.

Например:

$(\forall P)P(a)$

$(\forall f)(\forall X)P(f(X),b)$ – не являются ППФ логики предикатов первого порядка.

На практике ППФ используется для представления знаний. Не всегда легко представить знания, выраженные на естественном языке, с помощью ППФ. Например, выражение ”если два объекта равны, то они имеют одинаковые свойства” можно представить так;

$$(\forall P)(\forall X)(\forall Y)(PABHO(X,Y)\rightarrow(P(X)\leftrightarrow P(Y))).$$

Но это выражение не является формулой первого порядка, так как квантифицируется предикат.

Покажем, каким образом можно выявлять логическую структуру мысли. Приведем несколько сложных суждений, структуру которых надо выразить в виде формул, используя введенные логические термины.

1. Если у меня будет свободное время (a) и я сдам экзамены по математике (b) и физике (c), то поеду отдыхать в Крым (d) или на Кавказ (e).

Формула: $(a \wedge b \wedge c) \rightarrow (d \vee e)$.

2. «Если человек с детства и юности своей не давал нервам властвовать над собой, то они не привыкнут раздражаться и будут ему послушны» (К.Д.Ушинский).

Формула: $(\neg a \wedge \neg b) \rightarrow (\neg c \wedge d)$.

Здесь буква *a* обозначает суждение: «человек с детства давал нервам властвовать над собой». А так как у нас имеется отрицание («не давал»), то запишем $\neg a$.

3. Если добродетель неправильно приложат, то она может стать пороком.

Формула: $a \rightarrow b$.

4. Если у меня будет свободное время, то я почитаю книгу или посмотрю телевизор.

Формула: $a \rightarrow (b \vee c)$.

5. Некоторые элементарные частицы имеют положительный заряд.

Формула: $(\exists X)(S(X) \wedge P(X))$.

Правилом вывода называют процедуру, которая из одной или нескольких ППФ производит другие ППФ. Задавая фиксированное множество правил вывода, можно рассматривать следующее семейство проблем: исходя из выбранного множества ППФ применением некоторого числа раз правил вывода, можно получить заранее заданную ППФ.

Исходные ППФ называют *аксиомами*, а ППФ, полученные из правил вывода, называют *теоремами*. Цель применения правил вывода, ведущих от аксиомы к теореме, называют *доказательством теоремы*.

*Интерпретация формул***.** Формула имеет определенный смысл, т. е. обозначает некоторое высказывание, если существует какая-либо интерпретация. Интерпретировать формулу – это значит связать с ней определенное непустое множество D, т. е. конкретизировать предметную область, называемую также *областью интерпретации,* и указать:

для каждой константы в формуле – конкретный элемент из D;

для каждой n-местной функциональной буквы в формуле – конкретную n-местную функцию на D;

для каждой n-местной предикатной буквы в формуле – конкретное отношение между n элементами из D.

Свойства правильно построенных формул. При заданной интерпретации значения истинности правильно построенную формулу (ППФ) можно вычислить по правилам, объединенным в таблице истинности.

Если F и G – любые две ППФ, то значения истинности составного выражения, построенного из этих ППФ, даются таблицей истинности (табл. 1).

Таблица 1. Таблица истинности

F	G	$\neg$F	F$\vee$G	F$\wedge$G	F$\rightarrow$G	F$\leftrightarrow$G
И	И	Л	И	И	И	И
Л	И	И	И	Л	И	Л
И	Л	Л	И	Л	Л	Л
Л	Л	И	Л	Л	И	И

Если истинности двух ППФ независимо от интерпретации совпадают, то говорят, что эти ППФ являются эквивалентными. Пользуясь таблицей истинности, легко установить следующие эквивалентности (табл. 2.).

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что называется предикатом?
2. Из чего состоит алфавит языка предикатов первого порядка?
3. Что называют формулой алгебры логики?
4. Что такое элементарная формула?

5. Какую процедуру называют правилом вывода?
6. Что значить интерпретировать правильно построенную формулу?

Таблица 2. Таблица эквивалентности

Эквивалентные формулы		
$\neg(\neg F)$	F	
$F\to G$	$\neg F\vee G$	
$F\leftrightarrow G$	$(F\to G)\wedge(G\to F)$	
$\neg(F\wedge G)$	$\neg F\vee\neg G$	Законы де Моргана
$\neg(F\vee G)$	$\neg F\wedge\neg G$	
$F\wedge(G\vee H)$	$(F\wedge G)\vee(F\wedge H)$	Законы дистрибутивности
$F\vee(G\wedge H)$	$(F\vee G)\wedge(F\vee H)$	
$F\vee G$	$G\vee F$	Законы коммуникативности
$F\wedge G$	$G\wedge F$	
$(F\wedge G)\wedge H$	$F\wedge(G\wedge H)$	Законы ассоциативности
$(F\vee G)\vee H$	$F\vee(G\vee H)$	
$F\to G$	$\neg G\to\neg F$	
$F\wedge\neg F$	0	
$F\vee\neg F$	1	
$F\wedge 0$	0	
$F\wedge 1$	F	
$F\vee 0$	F	
$F\vee 1$	1	
$\forall XF(X)$ $\exists XF(X)$	$\forall YF(Y)$ $\exists YF(Y)$	
$\neg(\exists XF(X))$ $\neg(\forall XF(X))$	$\forall X(\neg F(X))$ $\exists X(\neg F(X))$	
$\forall X(F(X)\wedge G(X))$ $\exists X(F(X)\vee G(X))$	$\forall X\,F(X)\wedge\forall X\,G(X)$ $\exists X\,F(X)\vee\exists X\,G(X)$	

ЗАДАНИЕ

Выразить в символической форме следующие сложные суждения:

1. Если встать рано на рассвете и пойти в сад или парк, то можно услышать чудесные песни птиц.
2. Если эта фигура квадрат, то диагонали в ней равны, взаимно перпендикулярны и в точке пересечения делятся пополам.
3. Некоторые спортсмены не являются мастерами спорта.
4. «Некоторые лекарства опаснее самих болезней» (Сенека).
5. Некоторые люди не изучают логику.
6. Если мне дадут отпуск летом, то я поеду отдыхать к морю или по туристической путевке в Канаду.
7. Добро не умрет, а зло пропадет.

8. «Видеть несправедливость и молчать – это значит самому участвовать в ней» (Ж.-Ж. Руссо).

9. Если вы любите детей, полны жажды познания, имеете доброе сердце, мечтаете посвятить себя интересному творческому труду, то смело выбирайте профессию учителя.

10. «Если больному после разговора с врачом не становится легче, то это не врач» (В. М. Бехтерев).

11. «Если верный конь, поранив ногу,
Вдруг споткнулся, а потом опять,
не вини коня, вини дорогу
и коня не торопись менять» (Р. Гамзатов).

Практическая работа № 2

Преобразование правильно построенных формул в предложение

Аппарат логики предикатов используется для представления задачи в виде теоремы: формула H логически следует из формулы G, т.е. G→H. Доказательство этой теоремы состоит в том, чтобы показать, что каждая интерпретация, удовлетворяющая G, удовлетворяет и H, или, что ¬G∨H истинно для любой интерпретации; так как ¬(¬G∨H)= G∧¬H, то на практике обычно доказывают невыполнимость множества предложений G∧¬H.

Чтобы упростить доказательства теоремы, все формулы представляются в виде дизъюнкций литералов. *Литералом* (или *литерой*) называют атом или его отрицание. Формулу, представляющую собой дизъюнкцию литералов, называют *предложением* (или *дизъюнктом*).

Например: R(Z,a,g(X))∨(¬T(U))∨(¬U(b,k(c))).

Иными словами, в каждой формуле необходимо исключить все логические операции (включая кванторы), кроме дизъюнкции, и уменьшить область действия знака отрицания до одной предикатной буквы.

Преобразование ППФ в предложение. Перед тем как объяснить процесс резолюции, покажем, что любую ППФ исчисления предикатов первого порядка можно преобразовать во множество предложений, используя законы эквивалентности. Процесс такого преобразования состоит из последовательных этапов, которые покажем на следующем примере ППФ:
(∀X)(((P(X)∧Q(X,a))→R(X,b))∧((∀Y)((∀Z)(R(Y,Z)→T(X,Y)))))∨((∀X)S(X)).

Исключение символов эквивалентности и импликации. Для этого воспользуемся следующими законами эквивалентности: (F→G) заменим на (¬F∨G). В нашем примере такая подстановка даст:
(∀X)((¬(P(X)∧Q(X,a))∨R(X,b))∧((∀Y)(¬(∀Z)(R(Y,Z)∨T(X,Y)))))∨((∀X)S(X)).

Уменьшение области действия знаков отрицания. Нужно выполнить такое преобразование, чтобы каждый знак отрицания применялся не более чем к одной атомной формуле. Для этого используем следующие законы эквивалентности:

(¬(¬F)) и F
(¬ (F∧G)) и ¬F)∨(¬G)
(¬ (F∨G)) и ¬F)∧(¬G)
(¬(∃X)F(X)) и (∀X)(¬F(X))
(¬(∀X)F(X)) и (∃X)(¬F(X))

Полученная нами ППФ преобразуется к виду:
(∀X)((¬P(X)∨¬Q(X,a)∨(R(X,b))∧((∀Y)((∃Z)(¬R(Y,Z))∨T(X,Y)))))∨((∀X)S(X))

Разделение переменных. В пределах области действия квантора, переменная, связываемая с этим квантором, представляет собой немую переменную. Такую переменную в области действия квантора можно заменить любой другой (не встречающейся) переменной, при этом значение истинности ППФ не изменится. Стандартизация переменных в пределах ППФ означает переименование немых переменных с той целью, чтобы каждый квантор имел свою, свойственную только ему, немую переменную. Так, вместо записи

$$(\forall X)(P(X) \vee (\exists X)Q(X))$$

можно написать

$$(\forall X)(P(X) \vee (\exists Y)Q(Y)).$$

Полученная на предыдущем этапе ППФ преобразуется к виду:

(∀X)((¬P(X)∨¬Q(X,a)∨(R(X,b))∧((∀Y)((∃Z)(¬R(Y,Z))∨T(X,Y)))))∨((∀U)S(U))

Исключение кванторов существования. Рассмотрим ППФ

(∀X)(∃Y)P(X,Y)),

которую можно прочитать как: "Для всех X существует некоторое Y (возможно, зависящее от X) такое, что P(X,Y)". Поскольку квантор существования находится в области некоторого квантора общности, то можно допустить, что "существующий" Y зависит от X. Пусть эта зависимость определяется функцией Y=g(X), отображающей каждое X в Y. Такая функция называется *сколемовской*, или *функцией Сколема*. Заменив Y на g(X), мы исключим квантор существования, и наша ППФ примет вид:

(∀X)(P(X,q(X)).

Пусть ППФ (∃Y)P(Y) является подформулой другой ППФ, в которой $X_1,...,X_n$ квантифицированы универсально. Тогда удаляют Y и заменяют встречающуюся Y сколемовской функцией $f(X_1,...,X_n)$. Заметим, что эта функция будет содержать столько аргументов, сколько имеется универсальных квантификаторов слева от формулы (∃Y)P(Y). Эта функция, как мы уже говорили, просто выражает существование соответствия априори между совокупностью значений (∃Y)P(Y) и значениями Y. Поскольку эта функция заранее неизвестна, то в каждой замене Y мы должны записывать новую функциональную букву, которая отличается от уже имеющихся. Когда слева от символа "∃" отсутствуют символы "∀", то вводимая функция Сколема не будет содержать аргументов: следовательно, это будет новая константа, называемая константой Сколема. Так, ППФ (∃Y)P(Y) заменится на P(a), где про константу "a" известно, что она существует.

После выполнения замены Z функцией Сколема g(X,Y) наша ППФ будет иметь вид

(∀X)((¬P(X)∨¬Q(X,a)∨(R(X,b))∧((∀Y)(¬R(Y,g(X,Y))∨T(X,Y)))))∨((∀U)S(U)).

Перемещение всех квантификаторов общности в начало ППФ (без изменения их относительного порядка). Каждый квантор общности имеет свою переменную, и поэтому такие кванторы можно переместить в начало формулы, считая, что область действия каждого из них включает всю последующую часть ППФ. Говорят, что результирующая ППФ находится в префиксной форме. Она состоит из цепочки кванторов, называемой *префиксом*, и следующей за ней бескванторной формулы, называемой *матрицей*.

Префиксная форма для нашей ППФ имеет вид:

(∀X)(∀Y)(∀U)(((¬P(X)∨¬Q(X,a)∨R(X,b))∧(¬R(Y,Z)∨T(X,Y)))∨S(U)).

Приведение матрицы к конъюнктивной нормальной форме. Матрица находится в *конъюнктивной нормальной форме*, если она записана как конъюнкция конечного множества дизъюнкций литералов. Например:

$$(P(X) \vee Q(X,Y)) \wedge (P(Y) \vee \neg R(Y))$$

$$Q(X) \wedge R(X,Y)$$

$$\neg R(Y)$$

Любую матрицу можно привести к конъюнктивной нормальной форме, используя законы ассоциативности и дистрибутивности операций ∨ и ∧. После приведения матрицы нашего примера ППФ в конъюнктивную нормальную форму эта ППФ примет следующий вид:

(∀X)(∀Y)(∀U)(¬P(X)∨¬Q(X,a))∨(R(X,b)∨S(U))∧

(¬R(Y,g(X,Y))∨T(X,Y)∨S(U)).

Исключение кванторов общности. Поскольку все переменные в ППФ должны быть связаны, то можно предположить, что для каждой из них имеется квантор общности. При этом порядок расположения этих кванторов роли не играет, так что можно их исключить из

ППФ. После исключения кванторов из ППФ у нас остается лишь матрица в конъюнктивной нормальной форме.

Исключение символов конъюнкции. Теперь можно исключить символы $\wedge$, путем замены выражения типа (P$\wedge$Q) на множество ППФ {P,Q}. В результате выполнения таких замен получим конечное множество ППФ, каждая из которых – дизъюнкция литералов. А любая ППФ, содержащая дизъюнкцию литералов, как мы условились, является предложением.

Наш пример ППФ преобразуется в следующее множество предложений:

{$\neg$P(X)$\vee\neg$Q(X,a)$\vee$(R(X,b)$\vee$S(U)}

{$\neg$R(Y,g(X,Y))$\vee$T(X,Y)$\vee$S(U)}.

Переименование (разделение) переменных. Символы переменных должны быть изменены так, чтобы каждый появился не более чем в одном предложении. Напомним, что $(\forall X(P(X) \wedge Q(X))$ эквивалентно $(\forall X)P(X) \wedge (\forall Y)Q(Y)$. Теперь наши предложения приобретают вид:

$\neg$P(X)$\vee\neg$Q(X,a)$\vee$(R(X,b)$\vee$S(U)

$\neg$R(Y,g(Z,Y)) $\vee$T(A,B) $\vee$S(C)

Вместо переменных в литералах могут находиться термы, не содержащие переменных. Тогда говорят, что имеет место основной частный случаи. Например, P(a,f(g(b))) является основным частным случаем P(X,Y).

По структуре высказывания делятся на простые (они имеют логическую форму «S есть P» либо «S не есть P») и сложные (грамматически выражаются сложными предложениями). Простые высказывания бывают

атрибутивные (в них выражается принадлежность или непринадлежность какого-то свойства объекту или классу объектов);

об отношениях между несколькими объектами;

о существовании или несуществовании какого-либо объекта или явления).

В атрибутивные высказывания часто включаются кванторные связки.

По качеству простые высказывания делятся на утвердительные и отрицательные. С количественной точки зрения высказывания делятся на единичные, частные и общие.

Перечислим основные типы высказываний и их обозначения [5]:

A – общеутвердительные («Всякий S есть P»),

E – общеотрицательные («Всякий S не есть P»),

I – частноутвердительные («Некоторый S есть P»),

O – частноотрицательные («Некоторый S не есть P»).

В частных суждениях слово «некоторые» не исключает варианта «все».

Пример.

Преобразовать высказывание "Все дети имеют матерей" в предложение, т.е. в дизъюнкцию литералов. Это высказывание можно представить следующей ППФ:

$$\forall X \exists Y(\textit{ребенок}(X) \rightarrow \textit{мать}(Y) \wedge \textit{относится}(X,Y)).$$

Для удаления квантора существования заменим переменную Y функцией Сколема g(A), т.е. Y=g(X), откуда получим:

$$\forall X(\textit{ребенок}(X) \rightarrow \textit{мать}(g(X)) \wedge \textit{относится}(X,g(X))).$$

Теперь можно исключить квантор общности, который распространяет свое действие на все выражения. Кроме того, заменим символ "импликация" дизъюнкцией, используя закон P$\rightarrow$Q $\equiv \neg$P$\vee$Q, откуда получаем:

$$\neg\textit{ребенок}(X) \vee (\textit{мать}(g(X)) \wedge \textit{относится}(X,g(X))).$$

Используя закон

$$P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R),$$

можем написать

$\neg ребенок(X) \vee мать(g(X))$,

$\neg ребенок \vee относится(X, g(X))$.

Два полученные предложения можно интерпретировать так: "Если X –ребенок, то он относится к матери, выражаемой функцией g(X)". Из них следует:
ребенок(X)→ мать(g(X)) и
ребенок(X)∨ относится(X,g(X)).

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Дайте определение литерала.
2. Что называют дизъюнктором?
3. Какая функция называется функцией Сколема?
4. Как любую ППФ можно преобразовать во множество предложений?
5. Когда ППФ находится в префиксной форме?
6. Какая ППФ называется матрицей?
7. Когда матрица находится в конъюктивной нормальной форме?

ЗАДАНИЕ

Преобразовать высказывания в дизъюнкцию литералов.

1. Некоторые растения являются грибами.
2. Все предложения со второстепенными членами являются распространенными.
3. Все одноклеточные не являются червяками.
4. Некоторые космонавты – летчики.
5. Некоторые студенты – не лодыри.
6. Все дельфины – теплокровные.
7. Ни один кит не является рыбой.
8. Все свидетели дают истинные показания.
9. Ни одна балалайка не является клавишным инструментом.
10. Некоторые люди не любят природу.

Практическая работа № 3

Сущность принципа резолюций.
Примеры использования принципа резолюций

Резольвенты [2]. Говорят, что литерал является *конкретным*, если он не содержит никакой переменной. Например, ¬P(a) или Q(a,f(b)) являются конкретными литералами, в то время как ¬P(X) или Q(X,f(y)) не являются таковыми. Конкретным предложением является дизъюнкция конкретных литералов.

Пусть имеются два конкретных предложения

$P1 \vee P2 \vee \ldots \vee P_n \; и \; \neg P1 \vee Q2 \vee \ldots \vee Q_n$

Здесь литерал ¬P1 является отрицанием литерала P1. Из этих двух предложений можно вывести новое предложение, называемое *резольвентой,* или *резолюцией*. Резольвентой является дизъюнкция этих предложений с последующим исключением пары P1 и ¬P1.

Из предложений P1 и¬P1 получается *пустое предложение*, которое является признаком противоречия и обозначается символом □.

Два конкретных предложения могут не иметь резольвенты или могут иметь множество резольвент. Например, из $P \vee Q \vee R \; и \; \neg P \vee \neg Q \vee S$ получаются резольвенты $Q \vee \neg Q \vee R \vee S \; или \; P \vee \neg P \vee R \vee S$ (которые в действительности являются эквивалентными).

Сущность принципа резолюций. На практике невыполнимость множества предложений устанавливается посредством принципа резолюций. Речь идет о процедуре логического вывода новых предложений из множества исходных.

Принцип резолюций состоит:

1) в получении новых предложений на основании множества исходных и вновь получаемых предложений;

2) в отыскании частных случаев формул $F(t_1,\ldots,t_n)$ из $F(V_1,...,V_n)$ при подстановке вместо V_i произвольных термов t_i, т.е. $F(V_1,...,V_n)\rightarrow F(t_i,...,t_n)$.

Резольвенту двух предложений можно получить следующим образом:

1) переименовать переменные двух предложений так, чтобы последние стали одинаковыми;

2) найти подстановку, преобразующую литерал одного предложения в дополнительный по отнию к некоторому литералу другого предложения и произвести эту замену в обоих предложениях;

3) вычеркнуть дополнительные друг к другу литералы;

4) удалить одинаковые литералы в предложении кроме одного;

5) дизъюнкция литералов, оставшихся в обоих предложениях, является резольвентой.

Если некоторая последовательность резолюций, применяемых к исходному множеству предложений E и множеству резольвент, полученных в процессе резолюции, приводит к пустому предложению, то множество E является *невыполнимым.*

Для доказательства невыполнимости множества предложений пользуются *опровержением*. Доказательство выполнимости множества $G\rightarrow H$, что эквивалентно $\neg G\vee H$, – это опровержение его невыполнимости, или, что то же самое, доказательство невыполнимости $\neg(\neg G\vee H)$, что эквивалентно $G\wedge\neg H$.

Общий алгоритм опровержения с помощью резолюций может иметь следующий вид:

резолюция (G,H)

1) C: – множество предложений, полученных путем преобразования формул множества G;
2) добавить к множеству C предложения, полученные из $\neg H$;
3) пока пустое предложение не появится в C, **выполнить:**

начало: выбрать два различных предложения в C;

если они имеют резольвенты, то найти одну резольвенту и добавить ее к множеству C.

конец.

Примеры использования метода резолюций.

Пример 1. Предположим, что покупательная способность людей падает, если цены растут на товары и услуги. Предположим также, что люди несчастны, когда их покупательная способность падает. Предположим, что цены растут. Из этого можно заключить, что люди несчастны.

Применим следующие обозначения:

1) P – цены растут;
2) S – покупательная способность уменьшается;
3) U – люди несчастны.

В этом примере четыре утверждения:

1) если цены растут, то покупательная способность падает;
2) если покупательная способность падает, то люди несчастны;
3) цены растут;
4) люди несчастны.

Эти утверждения можно записать в виде следующих формул:

1) $P \rightarrow S$,
2) $S \rightarrow U$,
3) P,
4) U.

Преобразуем эти формулы в предложения:

1) $\neg P\vee S$,
2) $\neg S\vee U$,
3) P,

4) U.

Докажем путем опровержения, что U – логическое следствие из 1), 2), 3). Отрицаем 4) и получаем следующее доказательство:

1) $\neg P \vee S$,
2) $\neg S \vee U$,
3) P,
4) $\neg U$ отрицание заключения,
5) S резольвента 3) и 1),
6) U резольвента 5) и 2),
7) □ резольвента 6) и 4).

Пример 2. Все доктора имеют пациентов и некоторые пациенты любят своих докторов. Ни один пациент не любит знахаря. Следовательно, никакой доктор не является знахарем.

Обозначим:

1) P(X): X – пациент;
2) D(X): X – доктор;
3) Q(X): X – знахарь;
4) L(X,Y): X любит Y.

Тогда факты и заключения можно записать следующим образом:

F1: $(\exists X)(P(X) \wedge (\forall Y)(D(Y) \rightarrow L(X,Y)))$,
F2: $(\forall X)(P(X) \rightarrow (\forall Y)(Q(Y) \rightarrow \neg L(X,Y)))$,
G: $(\forall X)(D(X) \rightarrow \neg Q(X))$.

Преобразуем эти формулы в предложения:

1) P(a) из F1,
2) $\neg D(Y) \vee L(a,Y)$ из F1,
3) $\neg P(X) \vee \neg Q(Y) \vee \neg L(X,Y)$ из F2,
4) D(b) из G,
5) Q(b) из G.

Используя метод резолюций, получим следующее доказательство:

6) L(a,b) резольвента 4) и 2),
7) $\neg Q(Y) \vee \neg L(a,Y)$ резольвента 3) и 1),
8) $\neg L(a,b)$ резольвента 5) и 7),
9) □ резольвента 6) и 8).

Поясним естественность этого доказательства.

Из F1 можно предположить, что существует пациент a, который любит каждого доктора 1) и 2).

Предположим, что заключение неверно, т.е. b – одновременно и доктор, и знахарь.

Так как пациент любит каждого доктора, то a любит b 6).

Так как a – пациент, то a не любит никакого знахаря 7).

Однако b – знахарь. Следовательно, a не любит b 8).

Это невозможно из-за 6).Таким образом, мы закончили доказательство.

Пример 3. Допустим, что если Верховный Совет отказывается принимать новые законы, то забастовка не будет закончена, если только она не длится более года и зарплата не повышается. Закончится ли забастовка, если Верховный Совет отказывается действовать и забастовка только началась?

Применим следующие обозначения:

1) P – Верховный Совет отказывается действовать;
2) Q – забастовка заканчивается;
3) R – зарплата повышается;
4) S – забастовка длится более года.

Тогда приведенные выше утверждения можно представить следующими формулами:

1) F1: (P → (¬Q∨(R∧S))) – если Верховный Совет отказывается принимать новые законы, то забастовка не будет закончена, если она не длится более года и зарплата не повышается;
2) F2: P – Верховный Совет отказывается действовать;

3) F3: ¬S – забастовка только началась.

Требуется доказать, что ¬Q – логическое следствие F1∧F2∧иF3. Отрицаем Q и преобразуем F1, F2 и F3 в предложения:

1) ¬P∨¬Q∨R из F1,
2) ¬P∨¬Q∨S из F1,
3) P из F2,
4) ¬S из F3,
5) Q отрицание заключения.

Используя резолюции, получим следующее доказательство:

6) ¬Q∨S резольвента из 3) и 2),

7) S резольвента из 6) и 5),
8) □ резольвента из 7) и 4).

Пример 4. Таможенники обыскивают каждого, кто въезжает в страну, кроме высокопоставленных лиц. Некоторые люди, способствующие провозу наркотиков, въезжали в страну и были обысканы людьми, также способствующими провозу наркотиков. Никто из высокопоставленных лиц не способствовал провозу наркотиков. Доказать, что некоторые из таможенников способствовали провозу наркотиков.

Примем следующие обозначения:

1) E(X): X въезжал в страну;
2) V(X): X – высокопоставленное лицо;
3) S(X,Y): Y обыскивает X;
4) C(X): X – таможенник;
5) P(X): X способствует провозу наркотиков.

Заключение представляется формулой:

(∃X)(P(X)∧C(X)).

Отрицаем заключение и преобразуем формулы в предложения:

1) ¬E(X)∨V(X)∨S(X,f(X)),
2) ¬E(X)∨V(X)∨C(f(X)),
3) P(a),
4) E(a),
5) ¬S(a,Y)∨P(Y),
6) ¬P(X)∨¬V(X),
7) ¬P(X)∨¬C(X).

Доказательство методом резолюций выглядит следующим образом:

8) ¬V(a) из 3) и 6),
9) V(a)∨C(f(a)) из 2) и 4),
10) C(f(a)) из 8) и 9),
11) V(a)∨S(a,f(a)) из 1) и 4),
12) S(a,f(a)) из 8) и 11),
13) P(f(a)) из 12) и 50,
14) ¬C(f(a)) из 13) и 7)
15) □ из 10) и 14).

Заключение доказано.

Пример 5. Каждый, кто хранит деньги в кассе, получает проценты. Докажем, что если не существует процентов, то никто не хранит денег.

Примем следующие обозначения:

1) S(X,Y): X хранит деньги Y;
2) M(X): X есть деньги;
3) I(X): X есть проценты;
4) E(X,Y): X получает Y проценты.

Посылка записывается в виде:

$$(\forall X)(\exists Y)(S(X,Y)\wedge M(Y)) \rightarrow (\exists Y)(I(Y)\wedge E(X,Y))),$$

а заключение в виде:

$$\neg(\exists X)I(X) \rightarrow (\forall X)(\forall Y)(S(X,Y)) \rightarrow \neg M(Y)).$$

Преобразуем посылку в предложения:

1) $\neg S(X,Y)\vee\neg M(Y)\vee I(f(X))$,
2) $\neg S(X,f(X))\vee\neg M(Y)\vee E(X,f(X))$.

Преобразованное в предложение отрицание заключения имеет вид:

3) $\neg I(z)$,
4) $S(a,b)$,
5) $M(b)$.

Доказательство заключения:

6) $\neg S(X,Y)\vee\neg M(Y)$ из 3) и 1),
7) $\neg M(b)$ из 6) и 4),
8) □ из 7) и 5).

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. В чем смысл процедуры резолюции?

2. Что такое резольвента?

3. Какое предложение называется пустым?

ЗАДАНИЕ

Используя метод резолюций, доказать истинность заключения.

1. Лошадь есть животное, поэтому голова лошади есть голова животного.

2. Полиция обыскивает всех въехавших в страну, за исключением дипломатов. Шпион въехал в страну, однако распознать личность шпиона может только шпион. Шпион не является дипломатом. Среди полицейских имеется шпион.

3. Благородный труд заслуживает уважения, так как благородный труд способствует прогрессу общества. Труд учителя есть благородный труд, так как труд учителя заключается в обучении и воспитании подрастающего поколения. Труд учителя заслуживает уважения.

4. Все тюльпаны – цветы. Все цветы – растения. Все растения используют для питания углекислый газ атмосферы и выделяют в нее кислород. Все растения, использующие для питания углекислый газ атмосферы и выделяющие в нее кислород, содержат хлорофилл. Все тюльпаны содержат хлорофилл.

5. Все, что требует мужества и героизма, есть подвиг. Первый полет человека в космос требовал мужества и героизма. Первый полет человека в космос есть подвиг.

6. Если животное млекопитающее, то оно относится к типу хордовых. Это животное не является млекопитающим. Это животное не относится к типу хордовых.

7. Всякое преступление карается законом, поскольку оно общественно опасно. Грабеж есть преступление, так как грабеж – это открытое хищение личного имущества граждан. Грабеж карается законом.

8. Если должностное лицо получает взятку, то оно совершает преступление. Должностное лицо не получает взятку. Данное должностное лицо не совершает преступления.

9. Если на металле появились следы ржавчины, то началась коррозия. Коррозия не началась. На металле не появились следы ржавчины.

10. Если эта машина – двигатель внутреннего сгорания, то она является тепловым двигателем. Если эта машина является тепловым двигателем, то в ней топливо сжигается внутри цилиндра. Если эта машина – двигатель внутреннего сгорания, то в ней топливо сжигается внутри цилиндра.

11. Некоторый остров заселен исключительно рабами и сеньорами. На этом острове царствует некая принцесса. Сеньоры всегда говорят правду, а рабы все время лгут. Некоторые сеньоры бедны, другие богаты. Рабы также делятся на богатых и бедных. Принцесса желает выйти замуж за богатого раба. Каким же образом раб может всего лишь одним заявлением убедить принцессу в том, что он достоин быть ее супругом (какая фраза могла бы убедить принцессу в том, что говорящий эту фразу является богатым рабом)?

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ЛАБОРАТОРНЫМ ЗАНЯТИЯМ

ЛАБОРАТОРНАЯ РАБОТА № 1.
РАЗРАБОТКА ЭКСПЕРТНЫХ СИСТЕМ, БАЗИРУЮЩИХСЯ НА ПРАВИЛАХ.

Цель

1. Ознакомление с принципами построения экспертных систем.
2. Изучение структуры экспертных систем, базирующихся на правилах.
3. Построение простейшей экспертной системы, базирующейся на правилах.

Общие сведения

Экспертная система - это компьютерная программа, созданная для выполнения тех видов деятельности, которые под силу только человеку- эксперту, – например, проектирования, планирования, постановки диагноза, перевода, реферирования, ревизии, выдачи рекомендаций. Сферы применения экспертных систем – бизнес, проектирование, исследования, управление.

Программы ЭС обычно работают таким способом, который воспринимается как “интеллектуальный”, т. е. они имитируют образ действий человека-эксперта.

Эти программы специфичны, поскольку, как правило, используют механизм автоматического рассуждения (вывода) и так называемые слабые методы – такие как поиск, или эвристика. Они существенно отличаются от точных и хорошо аргументированных алгоритмов и не похожи на математические процедуры большинства традиционных разработок.

Основными компонентами экспертных систем являются:

- база знаний (БЗ), содержащая формализованное описание методов и знаний, привлекаемых при решении задач из области применения экспертных систем;
- механизм вывода (МВ), содержащий формализованное описание правил извлечения знаний из БЗ;
- система пользовательского интерфейса (СПИ), осуществляющая передачу знаний от МВ к пользователю.

В процессе работы экспертной системы (консультации) входные данные сопоставляются с данными из БЗ. Результатом сопоставления является утвердительный или отрицательный ответ. В экспертных системах, базирующихся на правилах, утвердительный ответ является результатом наличия в БЗ соответствующего продукционного правила. Выбор и активизацию продукционного правила реализует интерпретатор МВ. В каждом цикле работы интерпретатора (называемом распознавание – действие) производятся следующие действия:

- образец правила сопоставляется с элементами данных из БЗ;
- если можно активизировать более одного правила, то для выбора правила используется механизм разрешения конфликта (здесь не рассматривается);
- применяется выбранное правило.

Пример реализации экспертной системы выбора породы собаки, базирующейся

на правилах, приведен в прил. 1 (LAB01.PRO).

В программу включен дополнительный раздел **database,** содержащий определение предикатов динамической базы данных (БД).

Запись данных в БЗ производится стандартным предикатом

asserta (Факт),

в результате активизации которого указанный в скобках факт будет добавлен в начало БД.

Удаление фактов из БД производится стандартным предикатом

retract (Факт),

в результате активизации которого из БД удаляется первый факт, отождествленный с фактом, указанным в скобках.

При сопоставлении правил с содержимым БД используется стандартная функция отрицания (not), считающаяся выполненной успешно, если заданный в ней атом представляет собой цель, которая не достигается.

Задание к лабораторной работе

1. Провести тестирование программы LABO1.PRO (см. прил. 1).

2. Изменить программу LABO1.PRO так, чтобы перед окончанием работы выводилось содержимое БД, например, путем использования предиката вида wr_bd:-dpositive(P,Q),

```
write("dpositive(",P,", ",Q,")"), nl,
fail.
```

3. Изменить программу LABO1.PRO так, чтобы она обеспечивала распознавание животных в соответствии с правилами, приведенными в прил. 2.

Порядок выполнения задания

1. Загрузить Турбо- Пролог.
2. Загрузить программу LABO1.PRO и убедиться в правильности ее работы.
3. Внести требуемые изменения.

ЛАБОРАТОРНАЯ РАБОТА № 2
РАЗРАБОТКА ЭКСПЕРТНЫХ СИСТЕМ, БАЗИРУЮЩИХСЯ НА ЛОГИКЕ

Цель

1. Изучение структуры экспертных систем, базирующихся на логике.
2. Построение простейшей экспертной системы, базирующейся на логике.

Общие сведения

Структура экспертной системы, базирующейся на логике, аналогична структуре экспертной системы, базирующейся на правилах - БЗ состоит из утверждений в виде предложений логики предикатов; МВ реализует процесс распознавание – действие; СПИ выполняет те же функции, что и в системах, базирующихся на правилах.

Пример экспертной системы по породам собак, базирующейся на логике, приведен в прил. 1 в виде программы на Турбо-Прологе.

Программа выдает начальное меню, предлагая пользователю выбор между **consultation** (консультацией) и **exit from the system** (выходом из системы). Если пользователь выбирает консультацию, то между пользователем и системой происходит диалог. Затем пользователю сообщается результат. Результатом является либо выбранная порода, либо сообщение **Sorry. I can't help you** (Извините, я не могу вам помочь).

БЗ содержит утверждения логики предикатов, которые представлены либо в форме **rule** (правило), либо в форме **cond** (условие). В форме rule хранятся данные о породе; в форме cond-атрибуты (условия), характеризующие породу. Данные (ответы), получаемые от пользователя, динамически записываются в БД в форме предикатов yes (да) и no(нет).

МВ организован следующим образом: в результате активизации правила **go** осуществляется просмотр утверждений из БД rule и cond для выяснения существования или отсутствия подходящих значений данных. С этой целью вызывается правило **check** (проверка). Это правило содержит трассу номеров правил, номера условий и классифицированные объекты в БЗ. Оно пытается сопоставить объекты, классифицированные при помощи номеров условий. Если сопоставление происходит, то в программу добавляются сопоставленные значения и продолжается сопоставление с новыми данными, полученными от пользователя. Если сопоставления не происходит, МВ останавливает текущий процесс и выбирает другую трассу. Поиск и сопоставление продолжаются до тех пор, пока не исчерпаны все возможности. По завершении вывода go через интерфейс передает результаты пользователю.

СПИ состоит из трех частей: в первой содержатся правила для организации меню и уничтожения соответствующего окна после выбора пользователем предлагаемой ему программной функции: либо проведение консультации , либо выход из системы; вторая обеспечивает вывод списка собак и инициализацию процесса поиска и сопоставления по образу; третья запрашивает и получает ответы (yes или no) от пользователя.

Задание к лабораторной работе

1. Провести тестирование программы LABO2.PRO (см. прил. 1).

2. Изменить программу LABO2.PRO так, чтобы она обеспечивала распознавание животных в соответствии с правилами, приведенными в прил. 2.

Порядок выполнения задания

1. Загрузить Турбо- Пролог.
2. Загрузить программу LABO2.PRO и убедиться в правильности её работы.
3. Внести требуемые изменения.

ИНДИВИДУАЛЬНЫЕ ЗАДАНИЯ НА РАЗРАБОТКУ ФРАГМЕНТОВ ЭС

1. Придумайте систему управления лифтом, предусматривающую кнопки для трех этажей, а также для закрытия и открытия дверей, сенсорный датчик для обнаружения препятствия, возникающего при закрытии дверей, реле–таймер для фиксации времени, в течение которого двери открыты, и отдельные кнопки вызова на каждом этаже. Напишите систему продукций для управления лифтом. Пример типичной продукции:

Если срок действия таймера истек и
дверь открыта и
ничто не препятствует закрытию двери
То закрыть дверь

2. Требуется разработать фрагмент экспертной системы, предназначенной для прогнозирования местной погоды.

3. Требуется разработать фрагмент экспертной системы, предназначенной для обнаружения того, что делать, если автомобиль не заводится.

4. Требуется разработать фрагмент экспертной системы, предназначенной для определения стратегии гоночной яхты в регате. Система должна работать в режиме реального времени на протяжении всего периода гонок.

5. Требуется разработать фрагмент экспертной системы, предназначенной для помощи отвечающему по телефону доверия, когда отвечающий должен определить яд, который мог быть принят звонящим.

6. Требуется разработать фрагмент экспертной системы, предназначенной для определения оптимального маршрута продавца в любой данный день, причем продавец должен посетить всех клиентов и израсходовать возможное количество бензина.

7. Требуется разработать фрагмент экспертной системы, предназначенной для подбора членов экипажа с учетом их совместимости и возраста. При этом учитываются

индивидуальные особенности претендентов, их совместимость, пожелания тех или иных кандидатов работать вместе, а также их возраст (командир должен быть старше всех остальных членов экипажа).

8. Поиск информации в базе данных из другого модуля программы. В базу знаний занесена информация о сотрудниках:

а) фамилия;

б) год рождения;

в) место рождения;

г) национальность;

д) профессия;

е) место работы;

ж) занимаемая должность.

Система должна по вводимой с клавиатуры частичной информации о сотрудниках находить в базе знаний полную информацию и выводить ее на экран либо сообщать, что необходимая информация в базе данных отсутствует.

9. Требуется разработать фрагмент экспертной системы, предназначенной для поиска свободного места в салоне самолета. Система запрашивает требования к свободному месту

а) класс салона;

б) у окна или прохода;

в) для некурящих или нет

и выводит на экран номер свободного места. Программа и данные содержатся в различных модулях.

10. Требуется разработать фрагмент экспертной системы, предназначенной для диагностики неисправностей персонального компьютера. Экспертная система должна исследовать ситуацию и попытаться определить на общем уровне, допускает ли ошибки пользователь или, действительно, имеется неисправность в системном блоке, на диске, в мониторе и т.д. Возможный путь проектирования – беседа с мастером–профессионалом. При оценке ситуации подразумеваются грубые функциональные тесты, без глубокого анализа электронных элементов.

11. Требуется разработать фрагмент экспертной системы, предназначенной для подбора субоптимальной конфигурации персонального компьютера с учетом субъективных и объективных потребностей заказчика.

12. Требуется разработать фрагмент экспертной системы, предназначенной для подбора субоптимальной конфигурации локальной компьютерной сети с учетом множества эксплутационных, финансовых и прочих важных критериев.

13. Требуется разработать фрагмент экспертной системы, предназначенной для диагностики широко распространенных заболеваний человека по совокупности симптомов. Диагностируется не менее 20 болезней с учетом 15 типовых симптомов. Каждый симптом может указывать на несколько болезней (возможно, с разной степенью уверенности).

14. Требуется разработать фрагмент экспертной системы, предназначенной для консультации в отношении покупки автомобиля с учетом субъективных факторов,объективных потребностей и платежеспособности клиента, а также сезона и др.

15. Требуется разработать фрагмент экспертной системы, предназначенной для консультации в отношении покупки недвижимости с учетом связанных с этим важных факторов (надежность продавца, платежеспособность покупателя, страхование сделки, изменение цен и банковских процентных ставок и др.).

16. Требуется разработать фрагмент экспертной системы, предназначенной для выработки рекомендаций по обустройству дорожной сети (установка дорожных знаков, ограждений, нанесение разметки и т.п.). Вид ДТП - наезды на пешеходов, неожиданно выходящих на проезжую часть из-за стоящих транспортных средств.

Возможные мероприятия:

- установка знака 3.27 - "Остановка запрещена";
- установка знака 3.28 - "Стоянка запрещена";
- замена знака 3.28 знаком 3.27;
- обеспечение видимости знака в дневное и ночное время;
- установка дорожного ограждения перильного типа;
- организация поблизости пешеходного перехода;
- организация поблизости оборудованной стоянки транспортных средств;
- принятие мер по соблюдению водителями требований знака.

При выработке рекомендаций следует принять во внимание, не является ли участок дороги, где совершаются наезды, местом притяжения транспортных средств (например, рядом находится крупное учреждение) и/или пешеходов (остановка общественного транспорта, универмаг, кинотеатр и т.п.).

Дорожные знаки должны размещаться с учетом их наилучшей видимости участниками дорожного движения как в светлое, так и в темное время суток, при этом они не должны закрываться от участников дорожного движения какими-либо препятствиями – зелеными насаждениями, мачтами наружного освещения и т.п.. На участках дорог без стационарного освещения следует применять знаки со световозвращающей поверхностью, а на участках со стационарным освещением – знаки с внутренним и внешним освещением. В последнем случае возможна также установка знаков со светоотражающей поверхностью, если обеспечена их видимость с расстояния не менее 100 м.

В соответствии с правилами дорожного движения остановка и стоянка запрещены (даже при отсутствии соответствующего знака) на пешеходных переходах и ближе 5 м перед ними, а также ближе 15 м от остановочных площадок или указателей остановки общественного транспорта, если это создает помехи их движению.

ПРИЛОЖЕНИЕ 1

Текст программы работы №1 (LABO1.PRO).

```
/*      Пример экспертной системы          */
/*      базирующейся на правилах           */
/*      Эксперт по породам собак           */
domains

database
    dpositive(symbol, symbol)
    dnegative(symbol, symbol)

predicates
    do_expert_job
    do_consulting
    ask(symbol, symbol)
    dog_is(symbol)
    it_is(symbol)
    positive(symbol, symbol)
    negative(symbol, symbol)
    remember(symbol, symbol, symbol)
    clear_facts

goal
    do_expert_job.

Clauses
```

```
/* Система пользовательского интерфейса */
    do_expert_job:-
        makewindow(1,7,7, “Экспертная система”,1,16,22,58),
        nl,write(“**********************************************”),
        nl,nl,
        write(“ Добро пожаловать в ЖИВОТНУЮ экспертную систему! ;)”),
        nl,nl,write(“ Эта система легко определит название животного по “),
        nl,write(“ его признакам.                                “),
        nl,write(“  Отвечайте на вопросы : ‘Y’(Да) или ‘N’(Нет).     “),
        nl,write(“**********************************************”),
        nl,nl,
        do_consulting,
        nl,nl,
        clear_facts,
        write(“Нажмите пробел.”),nl,
        readchar(_),
        removewindow,
        exit.
    do_consulting:-dog_is(X),!,nl,
        write(“Похоже, что это - “, X, “.”).
    do_consulting:-nl, write(“Извините, но я ничем не могу вам помочь.”), nl,
        write(“И            вообще, где вы видели такое животное ?..”).
        ask(X,Y):- write(“ Вопрос: “,X,”  “,Y,” ? “),
        readln(Reply),
        remember(X,Y,Reply).

/*      Механизм вывода       */
    positive(X,Y):-dpositive(X,Y),!.
    positive(X,Y):-not(negative(X,Y)),!, ask(X,Y).
    negative(X,Y):-dnegative(X,Y),!.
    remember(X,Y,y):-asserta(dpositive(X,Y)).
    remember(X,Y,n):-asserta(dnegative(X,Y)), fail.
    clear_facts:-retract(dpositive(_,_)), fail.
    clear_facts:-retract(dnegative(_,_)), fail.

/*      Продукционные правила          */
    dog_is(“Английский бульдог”):-
            it_is(“короткая шерсть”),
            positive(has,”рост меньше 55 см”),
            positive(has,”низкопосаженный хвост”),
            positive(has,”хороший характер”),!.
     dog_is(“Гончая”):-
            it_is(“короткая шерсть”),
            positive(has,”рост меньше 55 см”),
            positive(has,”длинные уши”),
            positive(has,”хороший характер”),!.
    dog_is (“Дог”):-
            it_is(“короткая шерсть”),
            positive(has,”низкопосаженный хвост”),
            positive(has,”хороший характер”),
            positive(has,”вес больше 5 кг”),!.
    dog_is(“Американская гончая”):-
```

```
            it_is(“короткая шерсть”),
            positive(has,”рост меньше 75 см”),
            positive(has,”длинные уши”),
            positive(has, “хороший характер”),!.
      dog_is(“Коккер-спаниель”):-
            it_is(“длинная шерсть”),
            positive(has,”рост меньше 55 см”),
            positive(has,”низкопосаженный хвост”),
            positive (has,”длинные уши”),
            positive (has,”хороший характер”),!.
      dog_is(“Ирландский сеттер”):-
            it_is(“длинная шерсть”),
            positive(has,”рост меньше 75 см”),
            positive(has,”низкопосаженный хвост”),
            positive(has,”длинные уши”),!.
      dog_is (“Колли”):-
            it_is(“длинная шерсть”),
            positive(has,”рост меньше 75 см”),
            positive(has,”низкопосаженный хвост”),
            positive(has,”хороший характер”),!.
      dog_is(“Сенбернар”):-
            it_is(“длинная шерсть”),
            positive(has,”низкопосаженный хвост”),
            positive(has,”хороший характер”),
            positive(has,”вес больше 5 кг”),!.
      it_is(“короткая шерсть”):-
            positive(has,”короткая шерсть”),!.
      it_is(“длинная шерсть”):-
            positive (has,”длинная шерсть”),!.
/*     конец программы          */
```

Текст программы работы №2 (LABO2.PRO).

```
/*       Пример экспертной системы,          */
/*       базирующейся на логике.             */
/*       Эксперт по породам собак            */
domains
      conditions = bno*
      rno,bno,fno =integer
      category = symbol

database
/* предикаты базы данных */
      rule(rno,category,category,conditions)
      cond(bno,symbol)
      yes(bno)
      no(bno)
      topic(symbol)

predicates
/* предикаты системы пользовательского интерфейса */
      do_expert_job
```

```
        show_menu
        do_consulting
        process(integer)
        info(category)
        goes(category)
        listopt
        erase
        clear
        eval_reply(char)
/* предикаты механизма вывода */
        go(category)
        check(rno,conditions)
        inpo(rno,bno,string)
        do_answer(rno,string,bno,integer)

goal
        do_expert_job.

clauses
/* база знаний */
        topic("dog").
        topic("Короткошерстная собака").
        topic("Длинношерстная собака").

        rule(1,"dog","Короткошерстная собака",[1]).
        rule(2,"dog","Длинношерстная собака",[2]).
        rule(3,"Короткошерстная собака","Английский бульдог",    [3,5,7]).
        rule(4,"Короткошерстная собака","Гончая",                [3,6,7]).
        rule(5,"Короткошерстная собака","Дог",                   [5,6,7,8]).
        rule(6,"Короткошерстная собака","Американская гончая",   [4,6,7]).
        rule(7,"Длинношерстная собака","Коккер-спаниель",        [3,5,6,7]).
        rule(8,"Длинношерстная собака","Ирландский сеттер",      [4,6]).
        rule(9,"Длинношерстная собака","Колли",                  [4,5,7]).
        rule(10,"Длинношерстная собака","Сенбернар",             [5,7,8]).

        cond(1,"Короткая шерсть").
        cond(2,"Длинная шерсть").
        cond(3,"Рост меньше 55 см").
        cond(4,"Рост меньше 75 см").
        cond(5,"Низкопосаженный хвост").
        cond(6,"Длинные уши").
        cond(7,"Хороший характер").
        cond(8,"Вес более 5 кг").

/* Система пользовательского интерфейса */
        do_expert_job:-
                makewindow(1,7,7,"DOG EXPERT SYSTEM",0,0,25,80),
                show_menu,
                nl,write("Press spase bar."),
                readchar(_),
                exit.
        show_menu:-
```

```
        write("                                "),nl,
        write("**********************************"),nl,
        write("*         DOG EXPERT                      *"),nl,
        write("*                                         *"),nl,
        write("*     1. Consultation                     *"),nl,
        write("*                                         *"),nl,
        write("*     2. Exit the system                  *"),nl,
        write("*                                         *"),nl,
        write("**********************************"),nl,
        write("                                          "),nl,
        write("Please enter your choice: 1 or 2: "),nl,
        readint(Choice),
        process(Choice).
process(1):-do_consulting.
process(2):-removewindow, exit.
do_consulting:-goes(Mygoal),go(Mygoal),!.
do_consulting:-nl,write("Sorry, I can't help you."),
        clear.
do_consulting.
goes(Mygoal):-clear,clearwindow,nl,nl,
        write("                                "),nl,
        write("   WELCOME TO THE DOG EXPERT SYSTEM  "),nl,
        write("                                "),nl,
        write(" This is a dog identification system."),nl,
        write(" To begin the process of choosing  a "),nl,
        write(" dog, please type in 'dog'. If  you  "),nl,
        write(" wish to see the dog types, please   "),nl,
        write(" type in a question mark (?).        "),nl,
        write("                                "),nl,
        readln(Mygoal),
        info(Mygoal),!.
info("?"):-clearwindow,
        write("Reply from the KBS."),nl,
        listopt,nl,
        write("Please any key."),
        readchar(_),
        clearwindow,
        exit.
info(X) :- X >< "?".
listopt :-
        write("The dog types are: "),nl,nl,
        topic(Dog),
        write("        ",Dog),nl,fail.
        listopt.
inpo(Rno,Bno,Text) :-
        write("Question :-",Text," ? "),
        makewindow(2,7,7,"Response",10,54,7,20),
        write("Type 1 for 'yes': "),nl,
        write("Type 2 for 'no' : "),nl,
        readint(Response),
        clearwindow,
        shiftwindow(1),
```

```
            do_answer(Rno,Text,Bno,Response).
      eval_reply('y') :-
            write("I hope you have found this helpful !").
            eval_reply('n') :-
            write("I am sorry I can't help you !").
      go(Mygoal) :-
            not(rule(_,Mygoal,_,_)),!,nl,
            write("The dog you have indicated is a(n) ",Mygoal,"."),nl,
            write(" Is a dog you would like to have (y/n) ?"),nl,
            readchar(R),
            eval_reply(R).

/* механизм вывода */
      go(Mygoal) :-
            rule(Rno,Mygoal,Ny,Cond),
            check(Rno,Cond),
            go(Ny).
      check(Rno,[Bno|Rest]) :-
            yes(Bno),!,
            check(Rno,Rest).
            check(_,[Bno|_]) :- no(Bno),!,fail.
      check(Rno,[Bno|Rest]) :-
            cond(Bno,Text),
            inpo(Rno,Bno,Text),
            check(Rno,Rest).
      check(_,[]).
      do_answer(_,_,_,0) :- exit.
      do_answer(_,_,Bno,1) :-
            assert(yes(Bno)),
            shiftwindow(1),
            write(yes),nl.
            do_answer(_,_,Bno,2) :-
            assert(no(Bno)),
            write(no),nl,
            fail.
      erase :- retract(_),fail.
            erase.
      clear :-
            retract(yes(_)),
            retract(no(_)),
            fail,!.
            clear.
/* конец программы */
```

ПРИЛОЖЕНИЕ 2

Система правил для экспертной системы распознавания животных

ПРАВИЛО 1

ЕСЛИ животное имеет волосяной покров

ТО это животное – млекопитающее

ПРАВИЛО 2

ЕСЛИ животное дает молоко

ТО это животное – млекопитающее

ПРАВИЛО 3

ЕСЛИ животное имеет перья

ТО это – птица

ПРАВИЛО 4

ЕСЛИ животное может летать

И откладывает яйца

ТО это животное – птица

ПРАВИЛО 5

ЕСЛИ животное ест мясо

ТО это животное – хищник

ПРАВИЛО 6

ЕСЛИ животное имеет острые зубы

И животное имеет когти

И его глаза смотрят вперед

ТО это животное – хищник

ПРАВИЛО 7

ЕСЛИ животное является млекопитающим

И имеет копыта

ТО это животное – парнокопытное

ПРАВИЛО 8

ЕСЛИ животное является млекопитающим

И жует жвачку

ТО это животное – парнокопытное

ПРАВИЛО 9

ЕСЛИ животное является млекопитающим

И это животное – хищник

И это животное желто-коричневого цвета

И это животное имеет темные пятна

ТО можно предположить, что это животное – гепард

ПРАВИЛО 10

ЕСЛИ животное является млекопитающим

И это животное – хищник

И это животное желто-коричневого цвета

И это животное имеет темные полосы

ТО можно предположить, что это животное – тигр

ПРАВИЛО 11

ЕСЛИ животное является парнокопытным

И имеет длинную шею

И имеет длинные ноги

И имеет черные пятна

ТО можно предположить, что это животное – жираф

ПРАВИЛО 12

ЕСЛИ животное является парнокопытным

И имеет черные полосы

ТО можно предположить, что это животное – зебра

ПРАВИЛО 13

ЕСЛИ животное является птицей

И не может летать

И имеет длинную шею

И имеет длинные ноги

И имеет черно-белую окраску

ТО можно предположить, что это животное – страус

ПРАВИЛО 14

ЕСЛИ животное является птицей

И не может летать

И может плавать

И имеет черно-белую окраску

ТО можно предположить, что это животное – пингвин

ПРАВИЛО 15

ЕСЛИ животное является птицей

И хорошо летает

ТО можно предположить, что это животное – альбатрос

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СТУДЕНТОВ

Самостоятельная работа студентов предполагает самостоятельное изучение отдельных тем, дополнительную подготовку студентов к каждому практическому занятию.

В процессе изучения дисциплины «Экспертные системы» обучающиеся должны выполнить следующие виды самостоятельной работы:

самоподготовку к учебным занятиям по конспектам, учебной литературе и с помощью электронных ресурсов;

подготовка к тестированию по темам дисциплины.

Формой самостоятельной работы является работа с литературой. Овладение методическими приемами работы с литературой - одна из важнейших задач студента. Работа с литературой включает следующие этапы: предварительное знакомство с содержанием; углубленное изучение текста с преследованием следующих целей: усвоить основные положения; усвоить фактический материал; логическое обоснование главной мысли и выводов.

По выбранной теме студенты выполняют реферативную работу.

Работа включает следующие разделы:

1. Обоснование актуальности выбранной тематики и описание целей выполнения работы.

2. Систематизация и анализ найденных в научной печати, в сети Интернет и других источниках материалов.

3. Выводы.

4. Предложения по использованию результатов работы в конкретных областях и возможные направления дальнейших исследований.

5. Составление и отладка программы.

6. Составление отчета.