

ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

В. М. Зюзьков

Искусственный интеллект

Учебное пособие

Томск – 2007

УДК 681.3.07
ББК 32.973
3981

Зюзьков В. М.

3981 Искусственный интеллект: Учебное пособие. – Томск: Изд-во НТЛ, 2007. – 152 с.
ISBN 5-89503-

Данное пособие посвящено одной из наиболее перспективных и привлекательных областей развития научного знания – методологии искусственного интеллекта. Описываются история развития и области приложения искусственного интеллекта. Обсуждается разум в ракурсе искусственного интеллекта, представление и вывод знаний. Рассматриваются структуры и стратегии поиска в пространстве состояний. Описывается проблематика экспертных систем и автоматизированного проектирования. Изложение ориентировано на использования языка Пролог в качестве языка программирования.

Пособие содержит теоретический материал по дисциплине: «Интеллектуальные подсистемы САПР» для специальности 230104 «Системы автоматизированного проектирования».

**УДК 681.3.07
ББК 32.973**

Р е ц е н з е н т : Зав. кафедрой вычислительной математики и компьютерного моделирования
Томского государственного университета, профессор, д-р. физ.-мат. наук А. В. Старченко

Редактор *Л.В. Пермякова*
Верстка *В.М. Зюзьков*

К-ОКП ОК-005-93, код продукции 954240

Изд. лиц. Подписано к печати 34.12.07
Формат 60 × 80 $\frac{1}{32}$. Бумага офсетная. Печать офсетная. Гарнитура «Times».
Усл. п. л. 8,84. Уч.-изд. л. 9,90. Тираж 50 экз.

ООО «Издательство научно-технической литературы»
63450, г. Томск, пр. Ленина, 34а, тел. (382-2) 53-33-35

Отпечатано Том. гос. унт. сист. упр. и радиоэлектроники

ISBN 5-89503-

© Зюзьков В.М., 2007

Оглавление

1. Введение в искусственный интеллект.....	4
1.1. Предмет «Искусственный интеллект».....	4
1.2. Структура исследований в области искусственного интеллекта	10
1.3. Этапы в разработке ИИ (1637–1998).....	12
1.4. Психологическая теория интеллекта.....	42
2. Основания искусственного интеллекта	50
2.1. Законы мышления с позиций ИИ.....	50
2.2. О понимании	61
2.3. Тезис Чёрча	66
2.4. Эпистемологический круг.....	72
2.5. Точка зрения Пенроуза	75
2.6. Автореферентность.....	77
2.7. Представление знаний и вывод знаний	82
3. Решение проблем посредством поиска.....	112
3.1. Задача о кубиках. Общий метод решения задач	112
3.2. Основные методы поиска	116
3.3. Сведение задач к подзадачам. И/ИЛИ-графы	124
3.4. Игры и минимаксный принцип	129
4. Экспертные системы	136
4.1. Функции и структура экспертной системы	136
4.2. Продукции и неопределенность	139
4.3. Требования к современным экспертным системам	142
5. Искусственный интеллект и автоматизированное проектирование.....	145
Рекомендуемая литература	150

1. Введение в искусственный интеллект

1.1. Предмет «Искусственный интеллект»

Типичное изучение математики (как и любой формальной теории) в школе, в вузе сопровождается **ощущением растерянности, недоумения**. Определения и доказательства преподносят как настоящую реальность, но причины явлений никогда не объясняются. Кажется, что большую часть доказательств преподаватели получают с помощью магических манипуляций с кусочком мела у доски. Как можно связывать воедино все эти линии и не выпустить из поля зрения ни одну из них от самого начала доказательства до его чудесного конца? И над всем этим: **«А для чего все это надо?»**

Ответ приходит через несколько лет активной жизни. На самом деле все это ни для чего не надо, потому что предметы, которые вы изучаете, вносятся в школьные и вузовские программы достаточно произвольно. По правде говоря, эти знания служат лишь поводом для перехода к более серьезным вещам, таким, как **учиться понимать, учиться решать задачи, учиться познавать**. Но любопытно, что эти «вещи» не признаются и не преподаются. Можно сказать, что существует определенный вид интеллектуального терроризма, когда некоторых учеников называют «нуль в математике», хотя их единственная вина состоит в том, что они не понимают то, о чем ... никогда не говорится. Некоторым удастся этого избежать, потому что они раньше сумели познакомиться с неявными правилами этой игры. Есть и такие, кто учит все наизусть...

Причины, по которым именно так происходит обучение, достаточно серьезны. Знания инженера устаревают через 5–7 лет, а 80% знаний, которые понадобятся будущим специалистам (студентам), еще никому не известны. Поэтому главное в обучении – вооружить обучающихся методологией познания, научить самостоятельно овладевать знаниями.

Но существует область исследований, в которой первым желанием исследователей как раз является **стремление понять**, как система обработки инфор-

мации – будь то человек или машина – способна воспринимать, анализировать, передавать и обобщать то, чему ее обучают, и с помощью этих данных исследовать конкретные ситуации и решать задачи.

Данная область исследования – **искусственный интеллект** (artificial intelligence, ИИ) – старший сын информатики.

Предмет исследования ИИ – любая интеллектуальная деятельность человека, подчиняющаяся заранее **неизвестным** законам. Его можно также определить как «все то, что еще не сделано в информатике».

Искусственный интеллект – это область исследований, находящаяся на стыке наук. Специалисты, работающие в этой области, пытаются понять, какое поведение считается разумным (анализ), и создать работающие модели этого поведения (синтез). Исследователи ставят вопрос о том, как с помощью новых теорий и моделей научиться понимать принципы и механизмы интеллектуальной деятельности. Практической целью является создание методов и техники, необходимой для программирования «разумности», и ее передача компьютерам, а через них – всевозможным системам и средствам. Инженерные методы и навыки в области ИИ стали называть **технологией знаний** (knowledge engineering).

В области ИИ у нас имеются **трудности** двух типов:

- В большинстве случаев, выполняя какие-то действия, **мы сами не осознаем, как мы это делаем** – отсутствует алгоритм.
- Компьютеры **априори далеки** от человеческого уровня компетенции. До начала работы необходимо составить соответствующую программу. Но языки программирования позволяют выразить только элементарные понятия.

По своим методам ИИ – **экспериментальная научная дисциплина**.

Эксперимент в ИИ – это проверка и уточнение моделей (компьютерных программ) на многочисленных примерах – наблюдениях над человеком с целью раскрыть эти модели и лучше понять функционирование человеческого разума.

Есть и другие мнения. Станислав Лем предложил (1997):

«ИИ должен стать своего рода экспериментальной философией. Только таким образом удастся, наконец, совместить философские построения, основывающиеся на некоторых достаточно общих допущениях, и реальные выводы из той или иной философской системы. Сама эта система, таким образом, будет верифицирована, причем без вреда для цивилизации».

Впервые после фундаментального пересмотра картины мира (Коперник, Дарвин) разработка методов ИИ возвращает нас к вопросу о месте человека в природе. По существу впервые **оспаривается исключительность разума**.

Исследования в области ИИ рекурсивны – так как мы с помощью своего мышления пытаемся понять, как мы мыслим.

Теорема о прогрессе в ИИ

Как только какая-нибудь функция мышления окажется запрограммирована, люди тут же перестают считать ее ингредиентом «настоящего мышления».

«ИИ – это то, что еще не сделано».

Уточнение определения области ИИ:

Всякая задача, для которой неизвестен алгоритм решения, априорно относится к ИИ.

К сфере ИИ относятся все те различные области, где мы действуем, не имея абсолютно точного метода решения проблемы.

Две характерные особенности:

- В них используется информация в символьной форме: буквы, слова, знаки, рисунки.
- Предполагается наличие выбора:
 - нет алгоритма = нужно сделать выбор между многими вариантами в условиях неопределенности;
 - недетерминизм, свобода действия – существенная составляющая интеллекта.

Если в численных вычислениях внимание уделяется количественной стороне и численным значениям данных, то в символьных вычислениях важна качественная сторона данных: особенности их строения и функциональные особенности.

Поскольку ИИ имеет дело с символьной обработкой, то полезны специальные языки. Два широко распространенных языка в этой области – Лисп и Пролог. Мы будем использовать в качестве инструмента для задач ИИ язык SWI-prolog.

Эвристическое решение задачи как противоположность алгоритмическому

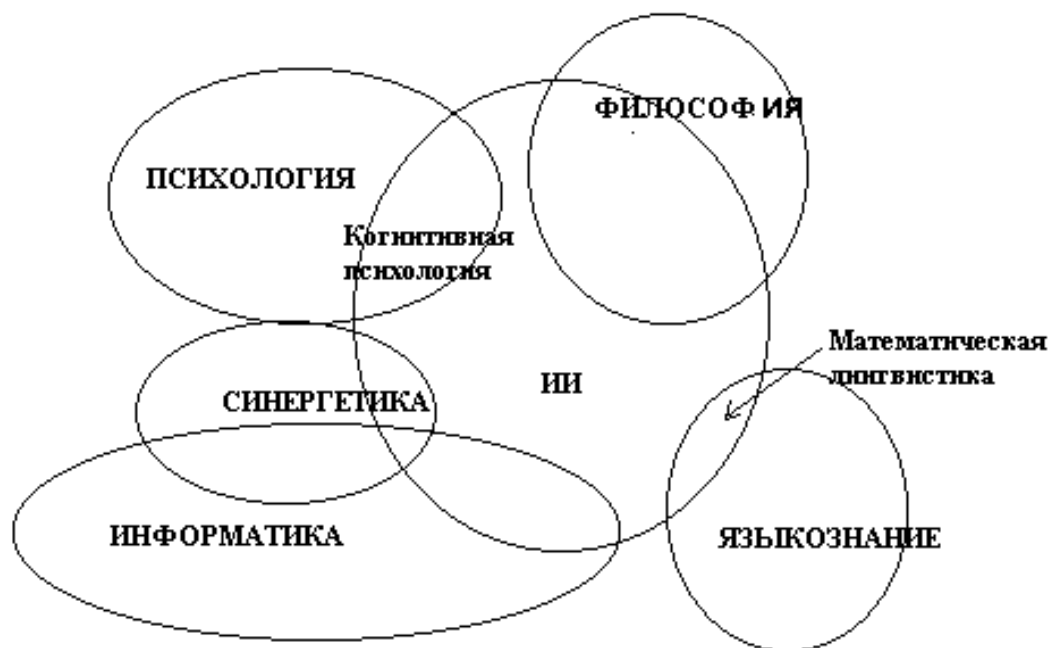
Эвристические решения, методы основываются на связанных с задачей специальных знаниях, простейших правилах, интуитивных критериях, базирующихся на предыдущем опыте и на других ненадежных методах вплоть до угадывания. Эвристические методы не всегда приводят к цели, даже когда решение существует, или они могут привести к неверному решению.

Когда используются эвристические методы?

- Неизвестен алгоритм.
- Доказано отсутствие алгоритмического решения.
- Существуют ограничения, налагаемые вычислительной техникой.

На основе эвристических решений во многих случаях потихоньку изобретаются алгоритмы. Эвристические программы в своей реализации базируются на алгоритмах.

ИИ – сфера исследования многих наук



Задача ИИ

Компьютеры, по определению, являются самыми негибкими, безвольными и послушными приказам существами. Несмотря на свою быстроту, они, тем не менее, сама бессознательность. Как же в таком случае можно запрограммировать разумное поведение? Не является ли уже само это предположение кричащим противоречием? Как преодолеть кажущуюся неприступной пропасть между формальным и неформальным, одушевленным и неодушевленным, гибким и негибким?

Работа специалистов по ИИ кажется странной и удивительной именно потому, что они разрабатывают строго формальные правила, говорящие негибким машинам, как стать гибкими.

Что это за правила такие, могущие описать всю сложность поведения разумных существ? Безусловно, это должны быть правила самых разных уровней: «простые» правила, «метаправила» для модификации «простых», «метаметаправила» для модификаций метаправил и так далее. Гибкость нашего разума зависит именно от огромного количества правил и сложности их иерархии. Не-

которые ситуации вызывают стереотипные реакции, для которых годятся «простые» правила. Другие ситуации представляют собой комбинации из стереотипных ситуаций; тут нужны правила, говорящие, какие из «простых» правил приложимы к данной ситуации. Некоторые ситуации вообще не поддаются классификации – следовательно, требуются правила для изобретения новых правил... и т.д. и т.п. Без сомнения, Странные Петли, правила, изменяющие сами себя, находятся в самом центре разума.

Рабочее определение ИИ (Д. Ф. Люгер)

ИИ – это дисциплина, исследующая закономерности, лежащие в основе разумного поведения, путем построения и изучения артефактов, предопределяющих эти закономерности.

Согласно этому определению, ИИ в меньшей степени представляет собой теорию закономерностей, лежащих в основе интеллекта, и в большей – эмпирическую методологию создания и исследования всевозможных моделей, на которые эта теория опирается.

Сильный и слабый ИИ

Так как искусственный интеллект по-разному понимается разными людьми, то используется следующая классификация.

- *Сильный ИИ* представляет собой программное обеспечение, благодаря которому компьютеры смогут думать так же, как люди. Помимо возможности думать компьютер обретет и сознание разумного существа.
- *Слабый ИИ* представляет собой широкий диапазон технологии ИИ. Эти функции могут добавляться в существующие системы и придавать им различные «разумные» свойства.

1.2. Структура исследований в области искусственного интеллекта

Направления

- **Бионическое направление**

Искусственное воспроизведение тех структур и процессов, которые характерны для живого человеческого мозга и которые лежат в основе процесса решения задач человеком. Это направление имеет четко выраженный фундаментальный характер, и его интенсивное развитие невозможно без одновременно глубокого изучения мозга нейрофизиологическими, морфологическими и психологическими методами.

- **Программно-прагматическое направление**

Создание программ, с помощью которых можно было решать те задачи, решение которых до этого считалось исключительно прерогативой человека (распознающие программы, простейшие игровые программы, программы для решения логических задач, поиска, классификации и т.п.).

Подходы

1. **Локальный или «задачный»** – основан на точке зрения, что для каждой задачи, присущей творческой деятельности человека, можно найти способ ее решения на ЭВМ, который будучи реализован в виде программы дает результат либо подобный результату, полученному человеком, либо даже лучший.
2. **Системный или основанный на знаниях** – связан с представлением о том, что решение отдельных творческих задач не исчерпывает всей проблематики искусственного интеллекта. Естественный интеллект человека способен не только решать творческие задачи, но и при необходимости обучается тому или иному виду творческой деятельности. Поэтому и программы искусственного интеллекта должны быть ориентированны не только или не столько на решение конкретных интеллектуальных задач, сколько на создание средств, позволяющих автоматически строить про-

граммы решения интеллектуальных задач, когда в таких программах возникнет необходимость.

3. Подход рассматривает проблемы создания интеллектуальных систем как **часть общей теории программирования** (как некоторый новый виток в этой теории). При этом подходе для составления интеллектуальных программ используются обычные программные средства, позволяющие писать нужные программы по описаниям задач на профессиональном естественном языке. Все метасредства, возникающие при этом на базе частичного анализа естественного интеллекта, рассматриваются здесь лишь с точки зрения создания интеллектуального программного обеспечения, т.е. комплекса средств, автоматизирующих деятельность самого программиста.

Разделы с точки зрения конечного результата

Программно-прагматическое направление

- Интеллектуальные программы (программы решения интеллектуальных задач).
- Работа со знаниями (теория и программы).
- Интеллектуальное программирование (теория и сервисные интеллектуальные программы).
- Интеллектуальные программные системы.

Интеллектуальные программы

Игровые программы

Человеческие игры

Переборные игры

Топологические игры

Стохастические игры

Компьютерные игры

Игры с жесткой схемой

Игры со сценарием

Естественно-языковые программыМашинный переводАвтоматическое реферированиеГенерация (синтез) текстов

Прозаические тексты

Поэтические тексты

Музыкальные программыСочинение музыкальных произведенийАнализ музыкальных произведенийИмитация исполнительского стиля

Распознающие и узнающие программы

Программы создания произведений графики и живописи

Прочие программыМодели поведенияПрограммы доказательства теоремЭвристические программы**1.3. Этапы в разработке ИИ (1637–1998)**

Перечислим некоторые вехи в развитии искусственного интеллекта в контексте других событий, связанных с ИИ. События, происшедшие в области ИИ, отмечены звездочкой (*).

1637 Декарт: «*Я мыслю, следовательно, я существую*».

Декарт (Descartes) Рене (латинизированное — Картезий; Cartesius)

(1596–1650), французский философ, математик, физик и физиолог. Безусловное основоположение всего знания по Декарту – непосредственная достоверность сознания («мыслию, следовательно, существую»).

- 1642 Блез Паскаль сконструировал *восьмиразрядный суммирующий механизм*.

Паскаль (Pascal) Блез (1623–1662), французский математик, физик, религиозный философ и писатель.

- 1666–1676 Лейбниц сформулировал в общих чертах подход к построению *логического исчисления* «*machina ratiocinatrix*».

Лейбниц (Leibniz) Готфрид Вильгельм (1646–1716), немецкий философ, математик, физик, языковед.

В течение всей своей философской биографии, а особенно с конца 1670-х гг., Лейбниц стремился осуществить алгебраизацию всего человеческого знания путем построения универсального «философского исчисления», позволяющего решить даже самые сложные проблемы посредством простых арифметических операций. При возникновении споров философам «достаточно было бы взять в руки перья, сесть за свои счетные доски и сказать друг другу (как бы дружески приглашая): давайте посчитаем!» Философское исчисление должно помогать как в формализации уже имеющегося знания (особое внимание Лейбниц уделял математизации силлогистики), так и в открытии новых истин, а также в определении степени вероятности эмпирических гипотез. Базисом философского исчисления является «искусство характеристики», т.е. отыскания символов (мыслившихся Лейбницем в виде чисел или же иероглифов), соответствующих сущностям вещей и могущих заменять их в познании. В книге «Об искусстве комбинаторики» (1666) предвосхитил некоторые моменты современной математической логики; он выдвинул идею о применении в логике математической символики и по-

строении логических исчислений, поставил задачу логического обоснования математики. Лейбниц сыграл важную роль в истории создания электронно-вычислительных машин; он предложил использовать для целей вычислительной математики бинарную систему счисления, писал о возможности машинного моделирования функций человеческого мозга. Лейбницу принадлежит термин «модель».

- 1726 Джонатан Свифт (1667–1745), английский писатель, в «Путешествиях Гулливера» описывает машину, которая пишет книги случайным образом.
- 1769* *Машина для шахматной игры* В. фон Кемпелена, по-видимому, была основана на обмане.
- 1793* Ф.Т. фон Шубертом предложен *алгоритм разложения арифметических выражений на множители*.
- 1801 Жозе-Мари Жаккарт (1752–1834, Франция) изобрел механизм для автоматизации процесса производства узорчатых тканей, который управлялся с помощью *перфокарт*.
- 1818 Шелли (Shelley) Мэри (урожденная Годвин, Godwin) (1791–1851), английская писательница, в романе «Франкенштейн, или Современный Прометей» пишет об *искусственном создании, которое уничтожает своего создателя*.

Это роман о великом свершении и роковой ошибке человеческого гения. Швейцарский ученый Виктор Франкенштейн, изучив труды средневековых алхимиков и каббалистов, а также достижения современной медицины и естествознания, проникает в тайны жизни. Вознамерившись победить саму Смерть, он создает человекоподобное гигантское существо, непередаваемо уродливое чудовище. Наделенное добрыми задатками, его творение стремится сблизиться с людьми и служить им, но люди ужасаются его вида, гонят прочь и пытаются убить, и оно, отчаявшись, объявляет войну чело-

веческому роду и своему создателю. Франкенштейн преследует сотворенного им «демона», чтобы его уничтожить. Погоня приводит преследователя и преследуемого в полярные широты, где ученый умирает на борту корабля, пробивающегося к Северному полюсу. Терзаемое одиночеством и раскаянием чудовище клянется перед телом своего создателя, что умертвит себя и сожжет, чтобы унести с собой тайну искусственной жизни. Философская идея Мэри Шелли – непостижимость тайн мироздания и опасность вмешательства человека в дела природы и ее законы – со временем была упрощена и опошлена. В бесчисленных драматизациях, экранизациях и примитивных «продолжениях» ее книги на первый план постепенно выдвинулось само чудовище, оттеснив своего творца на второй и даже присвоив себе его имя.

1835 Джозеф Генри изобрел *электрическое реле*.

Генри (Henry) Джозеф (1797–1878), американский физик. Построил мощные электромагниты и электродвигатель, открыл (1832, независимо от Майкла Фарадея) самоиндукцию, установил (1842) колебательный характер разряда конденсатора. Первый директор (1846) Смитсоновского института.

1835 Бэббидж разработал *идею аналитической машины*, в которой можно обнаружить множество элементов современных компьютеров. Он объединил принципы механических вычислений и набора команд, записанных на перфорированной бумажной ленте. Из-за недостатка финансирования эта машина так и не была построена.

Бэббидж (Babbage) Чарлз (1791–1871), английский математик и изобретатель, иностранный член-корреспондент Петербургской АН (1832).

В 1812–1813 гг. Бэббидж, решив искоренить ошибки из логарифмических таблиц, пришел к идее механических расчетов. В 1822 г. Бэббидж описал машину, способную рассчитывать и печатать

большие математические таблицы, и сконструировал машину для табулирования, состоявшую из валиков и шестеренок, вращаемых с помощью рычага. Машина могла производить некоторые математические вычисления с точностью до восьмого знака после запятой. Это был прообраз его разностной машины, к постройке которой он приступил в 1823 г., получив правительственную субсидию для продолжения работ. Разностная машина должна была производить вычисления с точностью до 20 знака после запятой. Постройка машины отняла у Бэббиджа 10 лет, ее конструкция становилась все более сложной, громоздкой и дорогой. Она так и не была закончена, финансирование проекта было прекращено.

Тем временем Бэббиджем овладела идея создания нового прибора – аналитической машины. Главное ее отличие от разностной машины заключалось в том, что она была программируемой и могла выполнять любые заданные ей вычисления. По существу аналитическая машина стала прообразом современных компьютеров, так как включала их основные элементы: память, ячейки которой содержали бы числа, и арифметическое устройство, состоящее из рычагов и шестеренок. Бэббидж предусмотрел возможность вводить в машину инструкции при помощи перфокарт. Однако и эта машина не была закончена, поскольку низкий уровень технологий того времени стал главным препятствием на пути ее создания.

Разностная машина в несколько видоизмененном виде была построена в 1854 г. шведским изобретателем Шойцем. В 1991 г. британскими учеными по спецификации Бэббиджа была построена вторая разностная машина, способная производить вычисления с точностью до 31 знака после запятой.

1843 Ада Августа Лавлейс (1815–1852), дочь Байрона, составила несколько программ, показывающих возможности программирования для проведения вычислений на аналитической машине Бэббиджа.

Потомки называют её *первым программистом*.

1847 Буль (Boole) Джордж (1815–1864), английский математик и логик, один из основоположников математической логики. Разработал алгебру логики (булеву алгебру) («Исследование законов мышления», 1854), основу функционирования цифровых компьютеров.

В своем исследовании (1844), опубликованном в «Философских трудах Королевского общества», он коснулся проблемы взаимодействия алгебры и исчисления. В том же году молодой ученый был награжден медалью Королевского общества за вклад в математический анализ.

Вскоре после того как Буль убедился, что его алгебра вполне применима к логике, в 1847 г. он опубликовал памфлет «Математический анализ логики», в котором высказал идею, что логика более близка к математике, чем к философии.

В 1854 г. опубликовал работу «Исследование законов мышления, базирующихся на математической логике и теории вероятностей». Работы 1847 и 1854 гг. дали рождение алгебре логики, или булевой алгебре. Буль первым показал, что существует аналогия между алгебраическими и логическими действиями, так как и те и другие предполагают лишь два варианта ответов – истина или ложь, ноль или единица. Он придумал систему обозначений и правил, пользуясь которыми можно было закодировать любые высказывания, а затем манипулировать ими как обычными числами. Булева алгебра располагала тремя основными операциями – И, ИЛИ, НЕ, которые позволяли производить сложение, вычитание, умножение, деление и сравнение символов и чисел. Таким образом, Булю удалось подробно описать двоичную систему счисления. В своей работе «Законы мышления» (1854) Буль окончательно сформулировал основы математической логики.

1859 Дарвин (Darwin) Чарлз Роберт (1809–1882), английский естество-

испытатель, создатель дарвинизма, иностранный член-корреспондент Петербургской АН (1867). В основном труде «Происхождение видов путем естественного отбора, или Сохранение благоприятствуемых пород в борьбе за жизнь» (1859), обобщив результаты собственных наблюдений (плавание на «Бигле», 1831–1836) и достижения современной ему биологии и селекционной практики, вскрыл основные факторы *эволюции органического мира*. Историческая заслуга Дарвина состоит в том, что он совместно с Уоллесом вскрыл движущий фактор эволюции – естественный отбор и тем самым выявил причины протекания биологической эволюции.

Достаточны ли указанные им факторы (борьба за существование, изменчивость и наследственность) для объяснения всех явлений развития или при дальнейшем исследовании найдутся и новые, пока не уясненные, – покажет будущее; но и будущая биология останется эволюционной биологией. Да и другие отрасли знания, социальные науки, антропология, психология, этика и др., преобразовались и преобразуются в смысле эволюционизма, так что книга Дарвина знаменует новую эру не только в биологии, но и вообще в истории человеческой мысли.

1872 Лодыгин Александр Николаевич (1847–1923), российский электротехник. Изобрел *угольную лампу накаливания* (1872).

В 1872 г. он подал заявку, но лишь в 1874 г., после двухлетней российской бюрократической волокиты, получил привилегию на лампу накаливания. Свое изобретение Лодыгин запатентовал также в Австрии, Великобритании, Франции и Бельгии. Он направил патентную заявку на угольную лампу накаливания и в Америку, но будучи не в состоянии уплатить положенный патентный сбор не мог получить патента США.

1876 Белл (Bell) Александер Грейам (1847–1922), один из изобретателей

телефона. По происхождению – шотландец. С 1871 г. в США. В 1876 г. получил патент на первый практически пригодный *телефон* (US Patent 174,465).

- 1879 Эдисон (Edison) Томас Алва (1847–1931), американский изобретатель, изобрел *лампу накаливания*.

Предпринятые Лодыгиным попытки коммерческого использования изобретенной им лампы накаливания окончились неудачей из-за отсутствия средств. Образцами ламп Лодыгина, привезенными в США офицером, приемщиком крейсеров, строившихся там по заказу Русского морского ведомства, заинтересовался американский изобретатель Т. Эдисон. Занявшись усовершенствованием различных конструкций электрических ламп накаливания, Эдисон в 1879 г. создал лампу с угольной нитью накала.

- 1879 Фреге изобрел *исчисление предикатов*.

Фреге (Frege) Готлоб (1848–1925), немецкий логик, математик и философ, основоположник логицизма. Дал первую аксиоматику логики высказываний и предикатов, построил первую систему формализованной арифметики.

- 1888 Американский инженер Герман Холлерит (1860–1929) сконструировал первую электромеханическую счетную машину. Эта машина, названная табулятором, могла считывать и сортировать статистические записи, закодированные на перфокартах. В 1890 г. изобретение Холлерита было впервые использовано в 11-й американской переписи населения. Работа, которую пятьсот сотрудников выполняли в течение семи лет, Холлерит сделал с 43 помощниками на 43 табуляторах за один месяц. Это самый ранний пример широкомасштабной *обработки данных*.

В 1896 г. Герман Холлерит основал фирму Computing Tabulating Recording Company, которая стала основой для будущей Интернэшнл Бизнес Мэшинс (International Business Machines Corporation,

IBM) – компании, внесшей гигантский вклад в развитие мировой компьютерной техники.

1901 Зигмунд Фрейд: «Толкование сновидений».

Фрейд (Фройд) (Freud) Зигмунд (1856–1939), австрийский врач-психиатр и психолог, основатель психоанализа. Психоанализ – как общая теория личности и ее эмоционального развития. Особое внимание психоанализ уделяет инстинктам и эмоциям, которые скрыты в *бессознательном* и оказывают глубокое влияние на мышление и поведение.

1904 Первая вакуумная трубка, *диод на основе электронной лампы* (Флеминг (Fleming) Джон Амброз (1849–1945), английский физик).

1912* *Автомат для эндипиля король и ладья против короля.*

Торрес де Кеведо (Torres de Quevedo) Леонардо (1852–1936), испанский математик, механик и инженер. Область его исследований была прикладная математика, автоматизация вычислений. Один из изобретателей современной вычислительной машины. Сконструировал механизмы для решения алгебраических и дифференциальных уравнений. Ввел (1915) в теорию машин раздел, который называл автоматикой. Высказал ряд положений, в дальнейшем использованных в кибернетике. Построил автомат «Игрок в шахматы» и робот «Телекин», который принимал и оценивал приказы, переданные по радио.

1913 Генри Форд ввел *сборочный конвейер*.

Форд (Ford) Генри (1863–1947), американский инженер, изобретатель и промышленник, один из пионеров автомобилестроения и основателей автомобильной промышленности.

Форд пришел к выводу, что самым слабым звеном на производстве является человек. Ему требовалось все больше рабочих, которые работали бы все быстрее, и лучшим средством достижения этого стал введенный в 1913 г. впервые в мире метод поточной (конвей-

ерной) технологии сборки автомобилей, что позволило всего за один год поднять производительность труда на 40–60%, а также достигнуть при этом стандартизации и взаимозаменяемости деталей.

- 1915 Эйнштейн (Einstein) Альберт (1879–1955), физик-теоретик, создал теорию относительности. Теория относительности утверждает, что наблюдатели в различных системах, двигающихся относительно друг друга, видят мир по-разному. Таким образом, наблюдатель оказался замешенным в определении физической реальности. Ученый теряет роль зрителя и становится активным участником изучаемой системы.
- 1917* *Робот* (чеш. Robot), термин, употребленный впервые К. Чапеком в пьесе «R. U. R.» (в чешском языке «робота» означает «воинская повинность», но в 1923 г. английский перевод оставляет оригинальное слово). ЧАПЕК (Capek) Карел (1890–1938), чешский писатель.
- 1925 Р. Вагнер обнаружил в биологических системах наличие *обратной связи* (процесс передачи информации о выходных данных системы на ее вход).
- 1928* Джон фон Нейман (1903–1957), американский математик и физик, доказал *теорему о минимаксе* – формулируется принцип оптимальности в антагонистических играх, предписывающий игрокам выбирать стратегии, на которых достигается максимальный гарантированный выигрыш (позднее использовался в программах для теории игр).
- 1930 Ванневар Буш (Vannevar Bush) изобрел дифференциальный анализатор (в дальнейшем их стали называть *аналоговыми вычислительными машинами*), предназначенный для решения систем дифференциальных уравнений.
- 1930 *Тезис Чёрча*: «Интуитивно и неформально определенный класс эф-

фективно вычислимых функций совпадает с классом частично-рекурсивных функций». Алонзо Чёрч (Alonze Church) – американский логик и математик (1903–1995).

Чёрч, Тьюринг и Марков, каждый в соответствии со своим подходом, выдвинули утверждение (тезис) о том, что класс определенных ими функций совпадает с неформально определенным классом вычислимых функций. В силу основного результата все эти утверждения логически эквивалентны. Название *тезис Чёрча* теперь применяется к этим и аналогичным им утверждениям.

1931 *Теорема Геделя о неполноте.*

Гедель (Godel) Курт (1906–1978), логик и математик. Родился в Австро-Венгрии, с 1940 – в США.

Многие математики начала века были озабочены идеей исключения всех возможных математических ошибок путем создания алгоритма для установления истины. Однако математик К. Гедель затруднил эти попытки, доказав свои т.н. *Теоремы о неполноте* (теоремы Геделя), из которых, в частности, следует, что не существует полной формальной теории, где были бы доказуемы все истинные теоремы математики. Он показал, что любая математическая теория является неполной, поскольку должны существовать теоремы, истинность которых не может быть доказана в пределах данной теории.

1936 Общественное телевидение в Великобритании.

1936 Алан Тьюринг публикует «On Computable Numbers», где вводит «*машину Тьюринга*» – одну из формализаций понятия алгоритма.

Тьюринг (Turing) Алан Матисон (1912–1954), английский математик.

Под воздействием идей Геделя Тьюринг начал разрабатывать алгоритмический метод, способный определить, является ли данная задача не имеющей решения с целью исключить такие задачи из ма-

тематики. Однако вместо этого в своей работе «О вычислимых числах» (1936) он доказал, что не существует такого универсального метода для определения вычислимости, и, следовательно, в математике всегда будут задачи, не имеющие решения (в отличие от пока неразрешимых). Работа Тьюринга опровергла мнение Д. Гилберта и его школы о том, что любая математическая теория может быть выражена через набор аксиом и теорем.

Чтобы проиллюстрировать свою точку зрения, Тьюринг предложил гипотетический механизм, названный «машиной Тьюринга». Это устройство, состоявшее из бесконечной бумажной ленты с записанными на ней символами и считывающей головки, могло решать любые математические или логические задачи. Таким образом, она обладала основными свойствами современного компьютера: пошаговым выполнением математических операций, запрограммированных во внутренней памяти. Эта машина открыла дискуссию по теории автоматов и создала теоретическую базу для работы цифровых компьютеров, которые появились в 1940-е годы.

1940 Джон Атанасов и Клиффорд Берри создали *первый электронный цифровой компьютер* (машина ABC (Atanasoff Berry Computer)), предназначенный для решения линейных уравнений.

В 1973 г. по решению суда изобретателем электронного компьютера назван американский физик Джон В. Атанасов. До этого приоритет отдавали Дж. Маучли и Дж. Эккерту, создателям машины ENIAC, так как ABC не был запатентован.

1940 *Robinson* – первый действующий компьютер в Великобритании, базирующийся на реле; использовался, чтобы декодировать секретные нацистские коды.

1940 Первая цветная телевизионная передача.

1941 Конрад Цузе в Германии создал *Z3* – *первый программируемый компьютер*, который был сконструирован на основе электромеха-

- нических реле и применялся для проектирования военных самолетов.
- 1943* У. Мак-Каллок (McCulloch) и У. Питц (Pitts) предложили формальную модель, отражающую функционирование нейрона, который может находиться в двух устойчивых состояниях; *начало нейрокибернетики*.
- 1944 По проекту Х.Х. Айкена в 1944 г. на одном из предприятий Ай-Би-Эм (IBM) в сотрудничестве с учеными Гарвардского университета по заказу ВМС США была создана машина «Марк-1». Это был монстр весом около 35 тонн. «Марк-1» был основан на использовании электромеханических реле и оперировал десятичными числами, закодированными на перфоленте. Машина могла манипулировать числами длиной до 23 разрядов. Для перемножения двух 23-разрядных чисел ей было необходимо четыре секунды.
- Компьютер отличался высокой работоспособностью (24 часа в сутки) и надежностью; на нем выполнялись главным образом расчеты по секретным проектам военно-морского флота.
- 1945* Ванневар Буш предложил идею *гипертекста* (сам термин был придуман двадцать лет спустя).
- 1945 Конрадом Цузе разработан *первый язык программирования* Plankalkuel («исчисление планов»). Цузе составил на этом языке множество программ, но транслятор для этого языка никогда реализован не был.
- 1945 Грейс Мюррей Хоппер извлекла мотылька из реле компьютера «Марк II» и приклеила его в операторский журнал с примечанием «*первый случай debugging*» (9-09-45 15:45 – запись в журнале).
- На компьютерном жаргоне ошибка в программе называется багом (bug – насекомое), debugging – отладка.
- 1946 Джон Маучли и Джон Эккерт создали *первую действующую электронную вычислительную машину* ENIAC. Ее вес составлял 30

тонн, она требовала для размещения 170 квадратных метров площади. Вместо тысяч электромеханических деталей ENIAC содержал 18 тысяч электронных ламп. Считала машина в двоичной системе и производила пять тысяч операций сложения или триста операций умножения в секунду. В машине были воплощены многие идеи Д. Атанасова.

1946 Джон фон Нейман предложил *логическую конструкцию электронной вычислительной машины* с программой, хранимой в памяти.

1947 Основывается ACM (Association for Computing Machinery).

1948 Норберт Винер (1894–1964, американский математик): «Кибернетика, или Управление и связь в животном и машине» – начало кибернетики.

Кибернетика – наука о таких системах, как биологические популяции, человеческий мозг, человеческое общество. Кибернетика возникла сразу по окончании второй мировой войны, когда с помощью систем управления и методов системотехники удалось решить некоторые задачи неврологии. Термин «кибернетика» предложен Винером. Кибернетика занимается изучением потоков информации внутри системы, способной воспринимать, запоминать и перерабатывать ее. Кибернетика занимается проблемами биологического контроля, автоматизации, общения между биологическими популяциями и искусственным интеллектом. Сейчас термин «кибернетика» используется только в основном в России.

1948 К. Шеннон создал *теорию информации*.

Шеннон (Shannon) Клод Элвуд (1916–2001), американский инженер и математик. В 1948 г. опубликовал работу «Математическая теория связи», в которой представил свою унифицированную теорию передачи и обработки информации. Информация в этом контексте включала все виды сообщений, в том числе те, которые передаются по нервным волокнам в живых организмах. Шеннон предложил

измерять информацию в математическом смысле, сводя ее к выбору между двумя значениями, или двоичными разрядами, – «да» или «нет», заложив, таким образом, фундамент современной теории связи, которая в настоящее время играет важную роль во многих областях.

1949 Язык Short Code, разработанный Джоном Маучли (John Mauchly), американцы считают первым (примитивным) *языком программирования высокого уровня*. На этом языке в отличие от языка Цузи Plankalkuel осуществлялось практическое программирование.

1949 Моррис Уилкс (Wilkes, профессор Кембриджа, Англия) создал EDSAC – первый компьютер с хранимой программой. Впервые для программирования использовался язык ассемблера и техника микропрограммирования (конструирование сложных команд процессора из простых – микрокоманд).

1949* Д. Хеббс (Donald Hebb) открыл способ создания самообучающихся искусственных нейронных сетей. Этот метод («Обучение по Хеббсу») позволяет изменять весовые коэффициенты в нейронной сети так, что данные на выходе отражают связь с информацией на входе.

1949* Фон Нейман разработал (но не реализовал) двумерный клеточный самовоспроизводящийся автомат с вычислительной универсальностью. Всего около 40000 клеток и 29 состояний для каждой клетки (описан в литературе в 1966 г.).

1950* Айзек Азимов: «Я – Робот».

Азимов (Asimov) Айзек (1920, Белоруссия – 1992, Нью-Йорк), американский писатель-фантаст, популяризатор науки.

В сборнике рассказов «Я – робот» разработал этический кодекс для роботов.

Три закона робототехники:

1. Робот не может причинить вред человеку или своим бездействи-

ем допустить, чтобы человеку был причинен вред.

2. Робот должен повиноваться командам, которые ему дает человек, кроме тех случаев, когда эти команды противоречат первому закону.
3. Робот должен заботиться о своей безопасности, насколько это не противоречит первому и второму закону.

На первый взгляд подобные законы при их полном соблюдении должны обеспечить безопасность человечества. Однако при внимательном рассмотрении возникают некоторые вопросы. Во-первых, законы сформулированы на человеческом языке, который не допускает простого их перевода в алгоритмическую форму. Попробуйте, к примеру, перевести на любой из известных Вам языков программирования такой термин, как «причинить вред». Или «допустить».

Теперь интересно, что будет подразумевать робот под термином «вред» после долгих логических размышлений? Не решит ли он, что все существование человека – это сплошной вред? Ведь он курит, пьет, с годами стареет и теряет здоровье, страдает. Не будет ли меньшим злом быстро прекратить эту цепь страданий?

Следующим вопросом будет такой. Что решит робот в ситуации, когда спасение одной жизни возможно только за счет другой? Особенно интересны те случаи, когда робот не имеет полной информации о том, кто есть кто.

- 1950* Алан Тьюринг предложил *«тест Тьюринга»* – критерий на наличие интеллекта у искусственного создания. (Собеседник считается разумным существом, если, получая его ответы на вопросы, невозможно убедиться в обратном.)
- 1950 Дж.П. Эккерт (Eckert) и Дж.У. Мочли (Mauchley) продают UNIVAC – *первый коммерческий компьютер*.
- 1950 Программа текстового редактирования для машины EDVAS.

- 1951 Грейс Мюррей Хоппер создала первый *компилятор* (для машины UNIVAC-1).
- 1951 В Советском Союзе создана машина МЭСМ (малая электронная счетная машина) – первая отечественная вычислительная машина, первая на европейском континенте машина с хранимой в памяти программой.
- 1951 Джей Форрестер (MIT – Массачусеттский технологический институт) изобрел *память с произвольным доступом* на магнитных сердечниках.
- 1951 Бюро Переписи Населения (США) покупает Remington-Rand UNIVAC за \$159,000 (позже \$250,000).
- 1952 Алексей Андреевич Ляпунов читает первый учебный курс для студентов МГУ «Принцип программирования».
- 1951 Трансконтинентальный черно-белый ТВ в США.
- 1953 Уотсон и Крик открыли *химическую структуру ДНК*.
 Дезоксирибонуклеиновые кислоты (ДНК) – высокополимерные природные соединения, содержащиеся в ядрах клеток живых организмов; вместе с белками гистонами образуют вещество хромосом. ДНК – носитель генетической информации, ее отдельные участки соответствуют определенным генам.
 Американский биохимик Дж. Уотсон и английский физик Ф. Крик на основании рентгеноструктурного анализа кристаллов ДНК и данных других исследователей предложили трехмерную модель ее структуры. Согласно этой модели, молекулы ДНК представляют собой две правозакрученные вокруг общей оси полинуклеотидных цепи, или двойную спираль.
- 1953* Х.Г. Кахриманян и Дж.Ф. Ноулан написали первые программы для машинного выполнения дифференцирования.
- 1954* Айзек Азимов: «Стальные пещеры».
- 1954 Смерть Тьюринга (самоубийство).

- 1954* А. Ньюэлл, Дж. Шоу и Г. Саймон разрабатывают «IPL» – *первый язык ИИ*.
- 1954* Работы по *автоматическому переводу* ведутся во многих странах – США, Советском Союзе, Израиле, Англии и др.
- 1955 IBM производит первый транзисторный калькулятор.
- 1956* А. Ньюэлл, Дж. Шоу и Г. Саймон создали программу «Логичный теоретик» («The Logic Theorist»), которая сумела заново передоказать многие теоремы математической логики из книги Уайтхеда и Рассела «Principia mathematica». Программа использовала эвристические правила, но неэффективно осуществляла поиск решения. Язык программирования – IPL.
- 1956* Первая конференция по ИИ в Дортмутском колледже; встреча Аллена Ньюэлла (специалиста по когнитивной науке), Герберта Саймона (впоследствии нобелевская премия по экономике), Джона Маккарти и Марвина Минского. Джон Маккарти вводит термин «искусственный интеллект».
- 1956 FORTRAN изобретен в IBM (Дж. Бэкус).
- 1956* С. Улам разработал «MANIAC I» – первую шахматную программу, которая победила человека.
- 1957* А. Ньюэлл, Дж. Шоу и Г. Саймон создали General Problem Solver (GPS); разрабатывалась с целью имитации процесса решения задач человеком и базировалась на идеях эвристического поиска.
- 1957 Хомский (N. Chomsky): «Синтаксические Структуры».
- 1958* М. Минский и Дж. Маккарти основывали MIT (Массачусетский технологический институт) AILab. М. Минский стал первым директором AILab.
- 1958* Джон Маккарти изобрел Лисп в MIT. Опубликовал статью «Программы и здравый смысл», в которой предложил использовать логику для представления знаний и вывода.
- 1958 Создан язык ALGOL 58.

- 1958 Джек Килби (Jack St. Clair Kilby) изобрел *интегральную схему*, позволившую размещать на тонкой кремниевой пластинке большие электронные схемы.
- 1958* Эвристические правила для доказательства геометрических теорем (Х. Гелернтер и Н. Рочестер).
- 1959 Грейс Мюррей Хоппер создала язык программирования COBOL для деловых применений и бизнеса.
- 1959* Эвристические правила для доказательства теорем исчисления предикатов первого порядка (П. К. Гилмор).
- 1959* Фрэнк Розенблатт ввел понятие «*персептрон*» – формальную модель распознавания, опирающуюся на использование формальных нейронов. Розенблатт доказал теорему сходимости персептрона, обеспечившую теоретическую базу алгоритмам обучения персептрона.
- 1959* Американский кибернетик А. Самюэль (Samuel) составил для вычислительной машины программу, которая позволяет ей играть в шашки, причем в ходе игры машина обучается или, по крайней мере, создает впечатление, что обучается, улучшая свою игру на основе накопленного опыта. В 1962 г. эта программа сразилась с Р. Нили, сильнейшим шашистом в США, и победила.
- 1959 Роберт Нойс (Robert Noyce) независимо от Kilby изобрел интегральную схему.
- 1960* Первые программы автоматического переноса при редактировании текста.
- 1961* Эвристические приемы формульного интегрирования (Дж. Слэгл).
- 1962* Первые коммерческие промышленные роботы.
- 1962 Томас Кун: «Структуры научных революций».
Кун (Kuhn) Томас (1922–1996), американский физик, философ и историк науки. Выдвинул концепцию научных революций как смены парадигм – исходных концептуальных схем, способов поста-

новки проблем и методов исследования, господствующих в науке определенного исторического периода.

1962 Яакко Хинтика: «Знания и убеждения».

Хинтика (Hintikka) Яакко (р. 1929), финский логик и философ. Предложил концепцию глубинной и поверхностной информации, в которой утверждается, что логико-математические истины содержат информацию о реальной действительности. Хинтика внес значительный вклад в развитие современной логики. Разработал метод модельных множеств и дистрибутивных нормальных форм для построения доказательств полноты логических систем; построил теоретико-игровую интерпретацию логики; получил важные результаты, исследуя семантики возможных миров, пропозициональные установки, логику вопросов и эпистемическую логику.

1962 Сол Крипке: «Семантика возможных миров».

Крипке (Kripke) Сол Арон (р. 1940), американский логик и философ, представитель аналитической философии. Разработал основы теоретико-модельной семантики; построил семантику возможных миров («семантика Крипке»), на основе которой выдвинул оригинальную концепцию истины; использовал ее для преодоления семантических антиномий.

1963* М. Куиллиан (M. Ross Quillian) предложил семантические сети — как средство для представления знаний.

1963 Станислав Лем: «Сумма технологий».

Футурологическое и социологическое исследование о развитии земных технологий, множество предсказаний в области искусственного интеллекта.

1963 Дональд Кнут приступает к созданию библии программистов «Искусство программирования для ЭВМ».

1963* Марвин Минский: «Шаги по направлению к искусственному интеллекту».

- 1964 IBM вводит 360 серию (System/360) – широкий спектр приложений, совместимость разных моделей (можно уже говорить о мобильности программного обеспечения).
- 1964 Языки программирования PL/1, BASIC.
- 1965 Лотфи Заде (Lotfi Zadeh) создал теорию нечетких подмножеств (нечеткая логика). В начале 70-х гг. нечеткую логику стали использовать на практике (управление работой парового двигателя).
- 1965* Метод резолюций в логическом программировании (Дж.А. Робинсон), который позволяет автоматизировать процесс доказательства теорем при наличии исходных аксиом.
- 1965* Тед Нельсон опубликовал первую печатную работу, в которой были изложены его идеи относительно связанных текстов и в которой он впервые ввел термин «гипертекст» для нелинейных документов.
- 1965 Язык программирования APL.
- 1965* Б. Букхенен, Э. Фейгенбаум и Э. Леденберг (Buchanan, Feigenbaum и Lederberg, Стенфордский университет) начали проект экспертной системы «ДЕНДРАЛ» (DENDRAL) (первой и самой известной впоследствии).

Пользователь дает системе «ДЕНДРАЛ» некоторую информацию о веществе, а также данные спектроскопии (инфракрасной, ядерного магнитного резонанса и масс-спектрометрии), и та, в свою очередь, выдает диагноз в виде соответствующей химической структуры.
- 1965* Г. Саймон предсказывает: «В 1985-м машины будут способны сделать любую работу, которую может делать человек».
- 1965* Книжки с критикой ИИ: Мортимер Тауб (Mortimer Taub) «Компьютеры и здравый смысл: миф о мыслящих машинах», Хуберт Дрейфус (Hubert Dreyfus) «Алхимия и ИИ».
- 1966* Знаменитая программа Вейценбаума (Weizenbaum) ELIZA, имитирующая беседу психоаналитика с пациентом.
- 1967* MacHack побеждает Дрейфуса в шахматы.

- 1967 Создана отечественная ЭВМ БЭСМ-6 (1 млн операций в секунду).
- 1968* Письмо Эдгера В. Дейкстры в САСМ «GO TO statement considered harmful» – против использования оператора перехода. Истоки структурного программирования.
- 1968 Изобретена «мышь» (Дуглас Энгельбард).
- 1968* Н. Хомский (Chomsky и Halle): «The Sound Pattern of English».
- 1969* Джон Маккарти (John McCarthy и Pat Hayes): «Философские проблемы с точки зрения искусственного интеллекта (Исчисление ситуаций)».
- 1969* М. Минский и С. Пейперт написали книгу «Персептрон», в которой доказали, что персептрон Розенблатта не в состоянии выполнять многие функции, предсказанные Розенблаттом. Это привело к сокращению ассигнований, выделяемых на развитие нейроинформатики, о чем позже Минский высказывал сожаление.
- 1969 Alan Kay в своей докторской диссертации описывает теоретически персональный компьютер.
- 1969 Кнут Д.: «Искусство программирования для ЭВМ» Т. 1.
- 1969 Кеннет Томсон и Деннис Ричи (Thomson и Ritchie) создали UNIX – первую широко распространенную операционную систему.
- 1969* Брайсон и Хо открыли алгоритм обучения многослойного персептрона (алгоритм обратного распространения ошибки).
- 1970 В Советском Союзе принято ошибочное решение ориентироваться не на отечественную вычислительную технику, а на машины серии 360 (копии этих машин создавались в социалистическом лагере в виде ЕС ЭВМ).
- 1970* Prolog (Колмероз – Colmerauer, Франция).
- 1970* Программа Терри Винограда SHRDLU (Обработка Естественного Языка, Мир Блоков). Программа содержала знания о некотором игрушечном мире, в котором можно было перемещать с помощью команд на английском языке кубики и пирамиды.

- 1970 Флоппи-дискеты.
- 1970* Джон Хортон Конвей изобрел игру «Жизнь». С этого началось использование социальных и эмерджентных моделей обучения.
- 1971 Сотрудник компании Intel Марсиан (Тед) Хофф создал первый микропроцессор, разместив несколько интегральных микросхем на одном кремниевом кристалле (микропроцессор Intel 4004). Это революционное изобретение кардинально перевернуло представление о компьютерах как о громоздких, тяжеловесных монстрах.
- 1971 Первый карманный калькулятор (Pocketronic).
- 1971 Язык Паскаль (Никлаус Вирт).
- 1971 Начало электронной почты.
- Рэй Томлинсон из фирмы Bolt Beranek and Newman стал первым человеком, который послал электронное сообщение с компьютера из одной сети на компьютер, входящий в другую сеть.
- 1972* Х. Дрейфус: «Чего не могут вычислительные машины».
- 1972* Язык Смолток (Smalltalk) разработан в Xerox PARC (Алан Кэй) – один из первых языков объектно-ориентированного программирования.
- 1972* Первая массовая компьютерная игра – «Электронный теннис» (Нолан Бушнелл – основатель фирмы Atari).
- 1972 Cray Research.
- 1973 Создан язык программирования C (Деннис Ричи и Брайн Керниган) и операционная система UNIX (Кеннет Томпсон и Деннис Ричи) переписана на C.
- 1973 Роберт Меткалф в кандидатской диссертации формулирует теоретические основы Ethernet, протокола, составляющего сегодня фундамент большинства локальных сетей (в 1983 г. протокол стал международным стандартом).
- 1973* Р. Шенк и Р. Абельсон (Schank и Abelson) вводят понятие скрипт (сценарий) – один из методов представления знаний. Шенк исполь-

зует скрипты в программе SAM, которая могла интерпретировать рассказы.

- 1974* Первый робот, управляемый компьютером.
- 1974* М. Минский: «A Framework for Representing Knowledge» – фреймы для представления знаний.
- 1975* Коопер и Erlbaum основали Nestor, чтобы разработать технологию нейронных сетей.
- 1975 Первый персональный компьютер Altair 8800 (256 байт памяти).
- 1975* Д. Холланд – основоположник развития генетических алгоритмов.
- 1976* Ньюэлл и Саймон сформулировали (рабочую) гипотезу о физической символьной системе. «Каждый агент, проявляющий разумность в общепринятом смысле, должен являться физической символьной системой (разумное поведение достигается путем физической реализации операций над символьными структурами)».
- 1976* Greenblatt создал первую Лисп-машину «CONS».
- 1976* Kurzweil создал читающую машину.
- 1976 Стив Джобс (Jobs) и Стив Возняк (Wozniak) основали фирму Apple Computer. Персональные компьютеры фирмы Apple и Macintosh во многом являлись стандартом для производителей других фирм.
- 1976 Cray-1 – супер-компьютер, 138 мегафлоп.
- 1977* Дуг Ленат (D. Lenat) в Стэнфордском университете создал программу «автоматический математик» (AM) и позднее – EURISKO, что позволило открыть новые теории в математике (повторные открытия известных понятий и теорем теории чисел).
- 1977 Основан Microsoft (Билл Гейтс и Пол Аллен).
- 1977* Начало рассмотрения агентных технологий. Агент – это аппаратная или программная сущность, способная действовать в интересах достижения целей, поставленных перед ним владельцем и/или пользователем. Первоначально работы исследователей сосредоточены на анализе принципов взаимодействия между агентами, на

декомпозиции решаемых задач на подзадачи и распределении полученных задач между отдельными агентами, координации и кооперации агентов и т.п. В дальнейшем агентные технологии получают массу приложений.

- 1978* Экспертная система PROSPECTOR обнаружила молибденовую жилу. PROSPECTOR – экспертная система, созданная для содействия поиску коммерчески оправданных месторождений полезных ископаемых.
- 1978* Основание Xerox LISP machines.
- 1979* Raj Reddy основал Институт Робототехники в Carnegie Mellon University.
- 1979* MYCIN – медицинский эксперт (экспертная система описана в Журнале Американской медицинской ассоциации). ЭС разработана группой из Стэнфордского университета. Ставит соответствующий диагноз по инфекционным заболеваниям, исходя из представленных ей симптомов, и рекомендует курс медикаментозного лечения любой из диагностированных инфекций. База данных состоит из 450 правил.
- 1979* Хофштадтер: «Гедель, Эшер, Бах: эта бесконечная гирлянда». Возьмите простоты и занятности Перельмана, вложите ее в уста героев Кэрролла, добавьте эрудицию Умберто Эко (только в области не истории, а математики и естественных наук) и философию Станислава Лема и получите... культовую книгу хакеров. Разнотемье наук сводится, в конце концов, к проблеме искусственного интеллекта. Толстенная на почти 800 страниц формата А4, испещренная формулами и схемами книга стала бестселлером. Пулитцеровская и множество других премий, переиздания, дополнительные тиражи и торжественное юбилейное издание, переводы и издания на французском, итальянском, немецком, испанском, венгерском, шведском, португальском, китайском... В 2001 г. вышла и на русском.

- 1980* Экспертные системы вплоть до тысячи правил.
- 1980* Джон Сирл (философ из Беркли). Предложил тест китайской комнаты, с помощью которого можно было проверить наличие «слабого» интеллекта, но невозможно установить присутствие «сильного» интеллекта. Тезис: способность выполнять такие функции, как перевод по сложным правилам, не означает, что тот, кто это делает, понимает значение «выходных данных».
- 1980 Xerox, DEC и Intel вводят Ethernet.
- 1981 А. Ньюэлл: «The Knowledge Level».
- 1981 Фирма Osborne Computer (основанная Адамом Осборном) стала выпускать первые портативные персональные компьютеры.
- 1981 IBM выпустила персональный компьютер (PC).
- 1981* Япония объявила о начале проекта машин пятого поколения, базирующихся на принципах ИИ. Проект был рассчитан на 10 лет и предусматривал среди многих целей создание Пролог-машины с возможностями общения на естественном языке. Этот проект способствовал активизации исследований в области ИИ во многих странах.
- 1981* PSL (Портативный Стандартный Лисп), работающий на целом ряде платформ.
- 1981* ЛИСП-машины от Xerox, LMI, и Symbolics, доступные коммерчески, делают динамическую ООП-технологию широко доступной.
- 1981* Определяется Common Lisp – фактический стандарт на язык Лисп.
- 1982* Джон Хопфилд (Hopfield) «оживляет» нейронные сети: обнаружил возможность успешного применения однослойной нейронной сети с симметричными связями в задаче распознавания образов.
- 1982* Экспертная система SRI's PROSPECTOR обнаруживает основной депозит молибдена.
- 1982 IBM PC.
- 1983 Бьорн Страstrup создал язык программирования C++.

- 1983* Айзек Азимов пишет “Robots of Dawn”.
- 1983 IBM вводит PCjr.
- 1983 Sony заявляет о технологии CD.
- 1984— Корпорации инвестируют \$50 000 000 на разработку ИИ.
- 1986
- 1984* Джарон Ланье основал компанию VPL (Visual Programing Language) Research для работ в области виртуальной реальности.
- 1984* Gold Hill создал Golden Common LISP.
- 1984* «Wabot-2» читает музыку с листа и играет на органе.
- 1984 Фирма Apple создала Macintosh; для стандартных операций вместо клавиатуры используются мышь и экранные меню.
- 1984 Изобретены оптические диски.
- 1985* GM и Campbell's Soup не используют Лисп для экспертных систем.
- 1985* Робот Kawasaki убивает японского механика в результате сбоя.
- 1985* М. Минский опубликовал «The Society of Mind» («Общество разума») — всеобъемлющую теорию мышления, охватывающая все, начиная с зарождения человеческой речи и кончая утверждением, что компьютер, способный мыслить, может и не подчиняться исключительно чистой логике. Ключевая идея Минского: мышление основано на сложном взаимодействии множества достаточно тривиальных программ — вроде той, что обеспечивает процесс дыхания без участия сознания.
- 1985* Teknowledge отказывается от Лиспа и Пролога в пользу C.
- 1986* Промышленный доход ИИ теперь \$1 000 000 000.
- 1986* Робот-игрок (Anderson) обыграл в пинг-понг человека.
- 1986* Фирма Borland предлагает Turbo PROLOG за \$99.
- 1986* Полиция Далласа использовала робота, чтобы прорваться в квартиру преступника.
- 1986* Психологи Д. Румельхарт и Г. Хинтон переоткрыли алгоритм обратного распространения ошибки для обучения нейронных сетей.

- 1986* Первая OOPSLA конференция по объектно-ориентированному программированию, в котором CLOS впервые рекламируется за пределами Lisp/AI общества.
- 1986* McClelland и Rumelhart: «Parallel Distributed Processing» (Нейронные сети).
- 1986* Крис Лангтон создал самовоспроизводящийся клеточный автомат из 100 клеток по 8 состояний каждая (не обладающий универсальной вычислимостью). Развитие дисциплины «Искусственная жизнь».
- 1987* 1900 работающих экспертных систем.
- 1987* Доход ИИ – 1,4 млрд дол., исключая робототехнику.
- 1987* Экспертная система «XCON» фирмы DEC, используемая для конфигурации компьютеров, делает работу 300 людей, применяя 10000 правил.
- 1987 Япония разрабатывает Систему Автоматизированной Идентификации Отпечатка Пальца.
- 1988* 386-й чип приводит PC к скорости, конкурирующей с машинами LISP.
- 1988* Доход экспертных систем – более 400 млн дол.
- 1988* Hillis «Connection Machine» способна выполнять 65536 параллельных вычислений.
- 1988* М. Минский и Пайперт (Papert) опубликовали дополненное и исправленное издание «Perceptrons».
- 1988* Объектно-ориентированные языки.
- 1988 Червь Морриса: сетевой вирус. Роберт Моррис-младший, аспирант, с помощью написанного им вируса инфицировал большое количество компьютеров, подключенных к Интернету.
- 1988 Система Mathematica 1.0 (С. Вольфрам).
- 1988* Объем финансирования проектов в области ИИ – 2 млрд дол.
- 1989* Тим Бернес-Ли (CERN, Швейцария) предложил принципы созда-

ния WWW (World Wide Web) и создает язык HTML (1990), первый Web-сервер и первую клиентскую программу: браузер-редактор (1991).

1990 Новые PC, NeXT, Mac SUN, DEC.

1991 Первый web-сервер: info.cern.ch.

1992* Apple Computer вводит Dylan, язык из семейства Lisp, как предтечу для будущих языков.

1992 Закончен японский проект ЭВМ пятого поколения.

1992 Начало японского проекта Real World Computing.

1992 Более чем 1000 типов компьютерных вирусов.

1992* Джон Коза – исследования в области генетического программирования.

1995 Роджер Пенроуз (английский математик и физик-теоретик). Книга «Тени разума».

Утверждая, что смоделировать интеллект на машине нельзя, Пенроуз предлагает физический механизм, на котором, возможно, основаны наши интеллект и сознание, – а может быть, и то неуловимое, что мы называем личностью человека. Основные результаты и гипотезы Пенроуза и его коллег суммированы в книге «Тени разума» и в нескольких статьях. Их можно разделить на «отрицательную программу» и «положительную программу».

Отрицательная программа сводится к математической аргументации (на основе теоремы Геделя) против возможности алгоритмически смоделировать разум. Понятием «разум» можно хоть как-то оперировать в формальных терминах, если иметь в виду *математическое* творчество – теоремы, вычисления, алгоритмы. Поэтому появляется возможность использовать достаточно четкие аргументы – а они-то как раз и подтверждают, что даже в математике самое существенное – то, что *не* формализуемо! Тем меньше остается надежд, что можно смоделировать другие свойства разума.

Положительная программа, строго говоря, есть всего лишь обсуждение комплекта согласованных друг с другом гипотез. Одна их часть относится к физике, другая – к нейрофизиологии, а в итоге получается вот что. Существенную роль в таком неотъемлемом свойстве разума, как *сознание*, играет некий «квантовый процесс» в так называемых *микротрубочках* нейронов мозга. Этот процесс влияет на сигналы, которыми обмениваются нейроны, внося принципиально важный ингредиент: *невычислимость* (а без нее не обойтись, если мы согласны с выводами отрицательной программы). В рамках существующей квантовой теории описать этот процесс невозможно (так как в ней все вычислимо, пусть даже и в вероятностном смысле). Можно сделать лишь некоторые количественные оценки, но до сколько-нибудь полной теории таких явлений еще далеко. Более того, Пенроуз считает, что создание этой теории должно быть связано с таким же радикальным, концептуальным пересмотром основ физики, какого в свое время потребовало создание общей теории относительности. По поводу реализуемости нужных квантовых процессов в клетках мозга тоже есть лишь косвенные данные. Однако работа продолжается очень активно, и к ней начинают подключаться экспериментаторы.

- 1995 Джеймс Гослинг создал язык Java – машинно-независимый язык для разработки приложений в сети Интернет.
- 1997* В фирме Карнеги Меллон создан суперкомпьютер для игры в шахматы Deep Blue. Эта машина смогла победить Гарри Каспарова, чемпиона мира по шахматам.
- 1998 Появился Haskell 98 – де факто стандарт современного функционального языка.

1.4. Психологическая теория интеллекта

Мозг, хорошо устроенный, стоит
больше, чем мозг, хорошо наполненный.

Монтень

Вся мировая история, основанная на блестящих догадках, изобретениях и открытиях, свидетельствует о том, что человек, безусловно, разумен. Психологической основой разумности является интеллект. В общем виде интеллект – это система психических механизмов, которая обуславливает возможность построения «внутри» индивидуума субъективной картины происходящего.

Классические исследования интеллекта человека с помощью тестов показали невозможность оценить интеллект с помощью какого-либо одного параметра (например, способность решать задачи) или нескольких параметров. Ибо, пытаясь построить теории интеллекта и тем более теории структуры интеллекта, тестологи описывали своего рода психологическую квазиреальность, вызванную к жизни их собственными изоощренными усилиями. Следствием явилась иллюзия «исчезновения» интеллекта как реального психологического процесса. Трудности идентификации результатов интеллектуального тестирования вынудили сторонников тестологического подхода перейти либо на операциональное определение интеллекта (интеллект – это то, что измеряют тесты интеллекта), либо на определения с помощью подбора «примеров» интеллектуального поведения (интеллект – это склонность субъекта вести себя определенным образом в определенной ситуации).

Мы примем следующее определение (Марина Холодная¹):

Интеллект – форма ментального (умственного) опыта.

Такое определение показывает, что объяснить природу интеллекта на уровне анализа его проявлений невозможно.

Еще одно следствие:

¹ Холодная М.А. Психология интеллекта: парадоксы исследования. – Томск: Изд-во Том. ун-та, М.: Изд-во «Барс», 1997. – 392 с.

«В пустую голову никакая информация вообще попасть не может. А если бы даже она туда и попала, то ее упорядочивание и преобразование было бы невозможно».

Три слоя ментального опыта:

1. *Когнитивный опыт* (cognitio – лат. – знание, познание) – хранение, упорядочивание и трансформация наличной и поступающей информации.
2. *Метакогнитивный опыт* – сознательная и непроизвольная организация собственной интеллектуальной активности.
3. *Интенциональный опыт* (intentio – лат. – стремление; направленность сознания, мышления на какой-либо предмет, намерение, цель) – субъективные критерии выбора (в предметной области) источников информации, направления поиска решения.

Чтобы оценить индивидуальный склад ума, надо ответить на вопросы:

- ⇒ Как человек перерабатывает поступающую информацию?
- ⇒ Может ли он контролировать работу своего интеллекта?
- ⇒ Почему именно так и именно об этом он думает?
- ⇒ Как он использует свой интеллект?

Особенности организации когнитивного опыта

Переработка информации происходит одновременно на трех уровнях:

- через знак (словесно-речевой способ кодирования информации);
- через образ (визуально-пространственный);
- через чувство (чувственно-сенсорный способ).

Когда мы нечто понимаем, мы это словесно определяем, мысленно видим и чувствуем.

Становление интеллекта предполагает развитие способности осуществлять обратимые переводы с одного «языка» на другой.

Особенности организации метакогнитивного опыта

В составе метакогнитивного опыта можно выделить четыре типа ментальных структур, обеспечивающих различные формы саморегуляции интеллектуальной активности: непроизвольный интеллектуальный контроль, произвольный интеллектуальный контроль, метакогнитивная осведомленность и открытая познавательная позиция.

- Непроизвольный интеллектуальный контроль

Когнитивные стили:

1. «Узость – широта сканирования видимого поля». В данном случае речь идет об индивидуальных различиях в том, как организуется внимание человека при его столкновении с проблемной ситуацией.
2. «Импульсивность-рефлексивность». Подавление проявления импульсивности в ситуации принятия решений («держат паузу») говорит не столько о том, что человек склонен более медленно перерабатывать информацию, сколько о тщательности и точности сбора информации до момента принятия решения.
3. «Широта категории». Одни люди строят свои суждения о происходящем на основе использования широких категорий, ориентируясь на сходство объектов и явлений, другие – на основе использования узких категорий, обращая внимание в основном на отличительные черты тех же самых объектов и явлений.
4. Особенности ориентировки в течение субъективного времени. Имеются в виду либо эффекты быстрого, либо медленного течения времени, а также эффекты хаотической ориентировки во времени. Психическое время течет более медленно у людей с низким уровнем структурированности психического пространства (с малым ментальным опытом).

- Произвольный интеллектуальный контроль

Основными индикаторами сформировавшихся метакогнитивных структур опыта, лежащих в основе произвольного интеллектуального контроля, являются:

1. Способность планировать – выдвигать цели и подцели собственной интеллектуальной деятельности, продумывать средства их реализации, выстраивать последовательность собственных действий и т.д.
2. Способность предвосхищать – учитывать последствия принимаемых решений, а также прогнозировать возможные изменения проблемной ситуации. 2000 лет назад китайский философ Шан Ян охарактеризовал эту способность следующим образом: «Глупый не понимает сути дела, даже когда оно уже выполнено. Умный же постигает суть дела еще до того, как появятся первые его признаки».
3. Способность оценивать – субъективно определять качество отдельных «шагов» собственной интеллектуальной деятельности («Похоже, здесь я ошибся...»), ее результата («Ай, да Пушкин! Сукин сын!»), а также собственных знаний в той или иной предметной области (в том числе и в такой экстремальной, доступной только мудрецам форме как «я знаю только то, что я не знаю»).
4. Способность прекращать или притормаживать интеллектуальную деятельность на любом этапе ее выполнения.
5. Способность выбирать стратегию собственного обучения и модифицировать ее под влиянием новых требований и с учетом своих интеллектуальных способностей. Жизнь и другие люди, конечно, учат многому. Однако вопрос заключается в том, что эти уроки могут оказаться бесполезными, если они не сочетаются со способностью человека к произвольному самообучению.

- **Метакогнитивная осведомленность**

Метакогнитивная осведомленность – это особая форма ментального опыта, характеризующая уровень и тип интроспективных (introspectare – лат. –

смотреть внутрь) представлений человека о своих индивидуальных ресурсах. Метакогнитивная осведомленность предполагает:

1. Знание своих индивидуальных интеллектуальных качеств (каковы особенности собственной памяти, мышления, предпочитаемые способы постановки и решения проблем и т.д.), а также знание оснований своей интеллектуальной деятельности (в виде представлений о закономерностях запоминания, правилах эффективного мышления, различиях между логически необходимыми и эмпирически верными суждениями и т.д.).
2. Умение оценивать свои индивидуальные интеллектуальные качества как на уровне «плохой – хороший», «недостаточный – достаточный», так и на уровне самопринятия. Заметим, что чувство интеллектуальной состоятельности (либо интеллектуальной несостоятельности) может существеннейшим образом влиять на темп и характер интеллектуального развития личности.
3. Готовность использовать приемы стимулирования и настройки работы своего собственного интеллекта. Соответственно, одним из критериев интеллектуальной зрелости является возможность человека оперативно и эффективно мобилизовать свои интеллектуальные силы для решения возникшей проблемы. Необходимо подчеркнуть, что приемы интеллектуальной самонастройки у каждого человека предельно индивидуализированы. Кому-то для того чтобы привести себя в состояние пика интеллектуальной формы, надо принять горячую ванну, кому-то стать под холодный душ, кому-то включиться в бурную дискуссию, кому-то совершить одинокую прогулку и т.д. вплоть до самых экстравагантных способов произвольной мобилизации своих интеллектуальных сил.

- Открытая познавательная позиция

О сформированной открытой познавательной позиции можно судить по ряду специфических состояний индивидуального ума:

1. Осознание возможности множества разнообразных мысленных «взглядов» на одно и то же явление.

2. Готовность использовать множество варьирующих способов описания и анализа того или иного явления. В том числе – способность произвольно переходить от одного способа к другому (от логико-аналитического – к образному, от интуитивно-ассоциативного – к алгоритмическому, от действенно-практического – к игровому и т.д.).
3. Осознание необходимости учета точки зрения другого человека, а также способность синтезировать разные познавательные позиции в условиях диалога с другими людьми.
4. Особое отношение к парадоксам и противоречиям, связанное с готовностью принимать любые необычные сведения без каких-либо субъективных защитных искажений.
5. Относительный характер индивидуальных суждений, проявляющийся в возможности, с одной стороны, соглашаться с явно различающимися по своему содержанию источниками информации и, с другой – сомневаться в, казалось бы, очевидном и бесспорном источнике информации. Так, известна притча о мудреце, к которому пришли для разрешения своего спора два человека. Внимательно выслушав одного, мудрец заявил: «Ты прав, уважаемый!» После этого, столь же внимательно выслушав прямо противоположные аргументы другого, мудрец изрек: «И ты прав, уважаемый!» Когда же жена мудреца, не вытерпев, отчитала мужа за столь нелепое поведение, он и ей ответил: «И ты права, дорогая!»
6. Восприятие происходящего по принципу «возможно все – даже то, что невозможно». Обычно, чем человек умнее, тем труднее его чем-либо удивить: даже самые невероятные события оказываются для него субъективно ожидаемыми.

Особенности организации интенционального опыта

- Предпочтения

Большинство одаренных людей предпочитают иметь дело со сложными объектами и ситуациями (им больше нравятся дисгармонические геометриче-

ские фигуры, они выбирают для решения более неоднозначные и трудные задачи, их привлекают, в первую очередь, парадоксальные и противоречивые сведения и т.п.). С другой стороны, интеллектуальные предпочтения уникальны, о чем свидетельствует крайнее разнообразие проявлений индивидуального интеллектуального выбора. По сути дела, предпочтения – это своего рода ментальный компас, выводящий человека в ту строго определенную область действительности, которая находится в максимальном соответствии с его индивидуальными интеллектуальными возможностями и в которой его интеллектуальные ресурсы могут реализоваться с максимальной эффективностью. В этом отношении прав был Дж. Пойа², когда писал, что «... логика – это дама, стоящая у выхода магазина самообслуживания и проверяющая стоимость каждого предмета в большой корзинке, содержимое которой отбиралось не ей».

- Убеждения

Вера в наличие определенных принципов, которым подчиняется природа изучаемых объектов. Изначальная уверенность в правильности выбранного пути изучения реальности. «На том стою и не могу иначе» – чрезвычайная помехоустойчивость интеллектуального труда одаренных людей.

- Умонастроение

Формируется особого рода «бессодержательное знание», благодаря которому такой человек заранее знает, что ему нужно и что ему не нужно делать. Пойа: «Полезная идея всегда возникает вместе с ощущением уверенности в том, что цель может быть достигнута и что именно данный элемент проблемной ситуации обязательно приведет к решению».

Вывод

Умен не тот, кто знает, а тот, у кого сформированы механизмы приобретения, организации и применения знаний.

² Джордж Пойа – выдающийся математик и педагог (1888 г. рождения).

Конфуций:

«Обладаю ли я знаниями? Нет. Но когда низкий человек спросит меня о чем-либо, то, даже если я не буду ничего знать, смогу рассмотреть этот вопрос с двух сторон и обо всем рассказать ему».

2. Основания искусственного интеллекта

2.1. Законы мышления с позиций ИИ

– Кролик – он умный! – сказал Пух в раздумье.

– Да, – сказал Пятачок, – Кролик – он хитрый.

– У него настоящие Мозги.

– Да, – сказал Пятачок, – у Кролика настоящие Мозги.

Наступило долгое молчание.

– Наверно, поэтому, – сказал наконец Пух, – наверно, поэтому-то он никогда ничего не понимает!

Пятачок: «У Пуха нет настоящих мозгов, но он делает все, как надо. Он поступает неразумно, а оказывается, что это было правильно».

А. Милн. Винни-Пух и все-все

Сознание

Большинство ученых понятия «сознание» и «разум» считают синонимами. Считается, что сознание – продукт естественного отбора³:

- Человек с его сознанием не сильно отличается от животного. Сознание в редуцированном виде уже есть у других живых существ.
- Сознание – часть процесса развития когнитивных способностей живых существ.

Главная разница между амебой и Эйнштейном не в способности производить теории, а в способности устранения ошибок в теории. Амеба не осознает процесса устранения ошибок. Основные ошибки амёбы устраняются путем устранения амёбы: это и есть естественный отбор.

³ Деннет Дэниэл С. Виды психики: на пути к пониманию сознания. – М.: Идея-Пресс, 2004. – 184 с.

У них по-разному происходит устранение ошибок. В случае амебы любая грубая ошибка может быть устранена устранением амебы. Ясно, что в случае Эйнштейна дело обстоит не так; он знает, что будет совершать ошибки, и активно ищет их. Однако не удивительно, что большинство людей унаследовали от амебы сильное нежелание как совершать ошибки, так и признавать, что они их совершили! Тем не менее, бывают исключения: некоторые люди не имеют ничего против совершения ошибок, если только есть шанс обнаружить их и – если ошибка обнаружена – начать всю работу сначала. Таким был Эйнштейн и таковы большинство ученых творческого склада: в противоположность другим организмам, человеческие существа используют метод проб и ошибок *сознательно* (если только он не стал для них второй натурой). Похоже, есть два типа людей: те, кто находится под чарами унаследованного отвращения к ошибкам и потому боится их и боится их признавать, и те, кто тоже хотел бы избегать ошибок, но знает, что мы чаще ошибаемся, чем не ошибаемся, кто узнал (методом проб и ошибок), что может противостоять этому, *активно ища свои собственные ошибки*. Люди первого типа мыслят догматически; люди второго типа – это те, *кто научился мыслить критически*.

Исходные положения:

- «интеллект» требует «понимания»;
- «понимание» требует «осознания»;
- осознание – пассивный аспект феномена сознания;
- свободная воля – активный аспект сознания.

Осознание: восприятие красного цвета, ощущение боли, восхищение музыкальным произведением.

Свободная воля: подъем с кровати или, напротив, намеренное решение воздержаться от какой-либо энергичной деятельности.

Активный и пассивный аспекты вместе: воспоминания, составление планов.

Основные критерии разума

- Гибко реагировать на различные ситуации.
- Извлекать преимущество из благоприятного стечения обстоятельств.
- Толковать двусмысленные или противоречивые сообщения.
- Оценивать различные элементы данной ситуации по степени их важности.
- Находить сходство между ситуациями, несмотря на возможные различия.
- Находить разницу между ситуациями, несмотря на возможное сходство.
- Создавать новые понятия, по-новому соединяя старые.
- Выдвигать новые идеи.

Что может настоящий интеллект?

Мы блаженствуем в переполненном информацией мире благодаря нашим изумительным способностям⁴:

- выбирать,
- выделять,
- глядеть украдкой,
- группировать,
- делать наброски,
- делать приближенные выводы,
- едва касаться,
- идеализировать,
- исследовать,
- обобщать,
- обследовать,
- окидывать взглядом,
- организовывать,
- отбирать,
- отбрасывать лишнее,
- отвлекаться,
- отделять,
- откидывать,

⁴ Эдвард Р. Тэфт, статистик и картограф.

- отмечать,
- отсеивать,
- отслеживать,
- отфильтровывать,
- пересматривать,
- подбирать,
- проверять,
- пролистывать,
- пропускать,
- просматривать беглым взглядом,
- различать,
- раскладывать по ящикам,
- располагать по парам,
- распределять по категориям,
- редактировать,
- сваливать в кучу,
- сглаживать,
- сгущать,
- сжимать,
- синтезировать,
- смешивать,
- собирать по мелочам,
- совершенствовать,
- согласовывать,
- соединять,
- сокращать,
- сооружать,
- сортировать,
- сосредотачиваться,
- составлять каталоги,
- составлять списки,
- схватывать на лету,
- увеличивать,
- укрупнять,
- усреднять,

- устанавливать связи,
- уточнять,
- отличать баранов от козлов.

Прыжки за пределы системы

Для человека всегда самое трудное выйти за пределы умственного тождества с самим собой.

С. Лем

Человеческому интеллекту свойственно умение, выпрыгивая за пределы системы, смотреть на то, что он делает, со стороны; при этом он ищет – и часто находит – какую-либо схему, закономерность. В то же время, сказав, что разум способен взглянуть на свою работу со стороны, но он это делает не всегда. Например, читающему человеку может захотеться спать. Вместо того чтобы дочитать книгу до конца, он может закрыть книгу и прекратить чтение. При этом он «выходит из системы»; нам это кажется вполне естественным.

В некоторых случаях только редкие личности могут заметить систему, управляющую жизнью многих людей, – систему, никогда раньше таковой не считавшуюся. Подобные личности зачастую посвящают свою жизнь тому, чтобы убедить остальных, что система действительно существует и что из неё необходимо выйти!

- (Литературные примеры: Станислав Лем: «Условный рефлекс», «Патруль» (О пилоте Пирксе)⁵.)

Насколько хорошо можно научить компьютер выскакивать за пределы системы? Приведем пример, в свое время удививший многих наблюдателей. Начало 80-х гг. на шахматном чемпионате среди компьютеров: у одной из программ (самой слабой) оказалась необычайная особенность – сдаваться задолго до конца партии. Она не была хорошим игроком, зато могла увидеть, когда позиция становилась безнадежной, и сдаться в этот момент, вместо того, чтобы ждать, пока другая программа пройдет через скучную процедуру матования.

⁵ Лем С. Рассказы о пилоте Пирксе. Лунная ночь. Радиопьеса. Собр. соч. в 10 т. Т. 4. – М.: Текст, 1993. – 400 с.

Хотя та программа проиграла все свои правила, она сделала это с шиком, удивив многих местных знатоков шахмат. Таким образом, если мы определим здесь «систему» как «делать ходы шахматной партии», ясно, что та программа имела сложную, заранее запрограммированную способность выходить из системы. С другой стороны, если вы считаете, что «системой» в данном случае является все то, что компьютер «запрограммирован делать», несомненно, что та программа вовсе не умела выходить из системы.

- (Литературный пример осознания искусственным разумом того, что находишься в системе: С. Лем, «Маска»⁶.)
- (Пример невозможности выхода насекомых из системы: Ж.А. Фабр «Нравы насекомых», пункт «Невежество инстинкта», Оса Сфекс (Sphex)⁷.)
- (Пример выхода из системы: о С. Рамануджане⁸ – слова Г. Харди: «он обладал такой способностью к обобщениям и **к быстрому изменению своих гипотез** и таким чувством формы, что ему не было равных в его области».)

Изучая формальные системы, очень важно отличить работу *внутри* системы от наших наблюдений *над* системой.

Формальная система *MIU*

Алфавит: *M, I, U*.

Формулы = $\{M, I, U\}^*$.

Аксиома: *MI*.

Правила вывода:

- 1) $xI \rightarrow xIU$ (продукция);
- 2) $Mx \rightarrow Mxx$ (продукция);
- 3) $III \rightarrow U$ (правило переписывания);
- 4) $UU \rightarrow \emptyset$ (правило переписывания).

⁶ Лем С. Рукопись, найденная в ванне. Высокий замок: Романы. Маска: Повесть. Собр. соч. в 10 т. Т. 5. – М.: Текст, 1994. – 351 с.

⁷ Фабр Ж. А. Инстинкт и нравы насекомых: В 2 т. Т. 1. – М.: ТЕРРА, 1993. – 608 с.

⁸ См. раздел 2.3. Тезис Чёрча.

Найдите вывод MU или докажите, что он не возможен.

Примечание.

- Продукция – правило, применяемое к формулам, рассматриваемым как единое целое.
- Правило переписывания – правило, которое может применяться к любой подформуле формулы.

Подсказка: во всех теоремах MIU «число символов I » $\equiv 1$ (или 2) $(\text{mod } 3)$.

Наверное, подобно большинству читателей, вы начали работу над головоломкой MU внутри системы. Вероятно, вы смогли ответить на этот вопрос; это пример работы внутри системы. Но наверняка в какой-то момент ваше терпение истощилось и вы вышли из системы, пытаясь проанализировать результаты вашей работы и понять, почему вам до сих не удалось получить MU . Возможно, вы смогли ответить на этот вопрос; это – пример размышления о системе. Нельзя сказать, что эти два метода совершенно несовместимы; напротив, по-видимому, любой человек до определенной степени способен одновременно работать внутри системы (*механический режим*) и размышлять над тем, что он делает (*интеллектуальный режим*). Более того, в человеческих делах часто почти невозможно точно отделить работу внутри системы от её анализа; жизнь состоит из такого количества сложных, переплетенных между собой систем, что подобное деление вообще кажется слишком большим упрощением. Однако сейчас для нас важно четко сформулировать простые идеи, чтобы в дальнейшем мы могли опираться на них при анализе более сложных систем.

Возможны ошибки, когда мы смешиваем рассуждения на разных уровнях. Например, когда нам не удастся четко разграничить работу внутри системы и размышления о системе. Может показаться, например, вполне разумным предположить, что, поскольку $P \vee \neg P$ – теорема, то одна из двух – либо P , либо не $\neg P$, должна также являться теоремой. Но это совершенно неверно: ни один из членов этой пары не является теоремой. Опасно считать, что символы

можно свободно передвигать между разными уровнями – как, например, язык формальной системы и её метаязык (русский).

Выход из самого себя – современный миф

Интересно поразмыслить над тем, можем ли мы, люди, выйти за пределы самих себя – и могут ли это сделать компьютерные программы. Разумеется, программа может модифицировать себя, но возможность всякой модификации должна быть заложена в программе с самого начала, так что это не может служить примером «выхода из системы». Как бы программа не вертелась и не извивалась, чтобы вырваться за свои пределы, она все же следует заложенным в ней правилам. Она так же не может выйти за пределы самой себя, как человек не может по желанию перестать следовать законам физики. Физика – это система, выхода из которой не существует. Однако возможно осуществить нечто подобное в меньшем масштабе, а именно: выйти из подсистемы собственного мозга в более широкую подсистему. Иногда удастся сойти с наезженной колеи. Это все еще объясняется взаимодействием различных подсистем мозга, но на вид это весьма похоже на полный выход из себя. Подобно этому, можно представить, что нам удастся создать программу, умеющую частично «вылезать из своей шкуры».

Однако важно не упускать из вида отличий между *восприятием* самого себя и *выходом* из самого себя. Воспринимать себя вы можете различными способами: в зеркале, в описаниях вас другими и т.п. Но вы не можете выйти из собственной кожи и встать снаружи себя самого. Некоторые формальные математические теории (например, арифметика Пеано) могут говорить о себе, но они не могут выйти из себя. Компьютерная программа может модифицировать себя, но она не может нарушить своих собственных инструкций, – большее, на что она способна, это изменить себя частично, следуя тем же инструкциям.

Дзен и выход из системы

Может быть выход из себя является центральной темой дзена. Дзен-будист старается глубже понять, кем он является на самом деле, все более и бо-

лее освобождаясь от его собственных идей о себе самом, нарушая все правила и соглашения, которые, по его мнению, держат его связанным, – включая правила самого дзена.

Как связано мышление с состоянием мозга?

Само состояние мозга в данный момент времени не содержит никакой информации о том, какая мысль будет выбрана. Это в большей степени определяется внешними обстоятельствами.

Из этого следует, что в зависимости от обстоятельств один и тот же мозг может породить две полностью противоречивые мысли. Любое заслуживающее внимания прочтение мозга на высшем уровне должно содержать все эти конфликтные версии. На самом деле, совершенно ясно, что мы – не что иное, как ходячие мешки противоречий, и наша целостность зависит от того, что в каждый данный момент мы способны сконцентрироваться только на чем-то одном. На чем именно – это предсказать невозможно, поскольку обстоятельства, определяющие выбор, заранее не известны.

Рабочая гипотеза ИИ

Мышление можно описать на некотором более высоком уровне (уровне символов), чем на уровне физиологии мозга (нейроны и т. п.).

Символы – это представление мышления в виде некоторой системы высокого уровня. Как представлены символы в мозгу (как аппаратура или как программа) – неизвестно.

Если считать, что интеллект есть характеристика, отделимая от аппаратуры, в которой она заключается, то явления сознания и интеллекта – это явления высшего порядка в том же смысле, как и многие другие сложные явления природы; они управляются своими законами высшего уровня, которые, разумеется, зависят от низшего уровня, но тем не менее могут быть от него отделены.

Эта гипотеза основана на редукционизме.

Редукционизм в науке – подход к знаниям, который пытается понять явления одного уровня научных феноменов в терминах другого, низшего, предположительно более фундаментального уровня.

В чистом виде это сформулировал Людвиг Витгенштейн в своем «Логико-философском трактате»⁹:

- Мир есть совокупность фактов, а не вещей.
- Все факты можно разложить на простые объекты – атомарные факты.
- Факты – это и есть символы языка, научившись группировать которые появляется интеллект.

Витгенштейн утверждает:

- Мы создаем для себя образы фактов.
- То, что элементы образа соединяются друг с другом определенным способом, показывает, что так же соединяются друг с другом и вещи.

С точки зрения редукционизма, для того, чтобы понять, что такое интеллект, а в более общем случае – сознание, надо отыскать в субъекте (человеке или компьютере) простые элементы и логические отношения, отражающие объекты и отношения мира. Зафиксировав эти отношения де-факто в виде правил, можно решить задачу ИИ.

Кризис постановки задач ИИ

Программными методами сейчас моделируют не интеллект и даже не современное представление о нем, а представления об интеллекте у научного общества середины XX в.

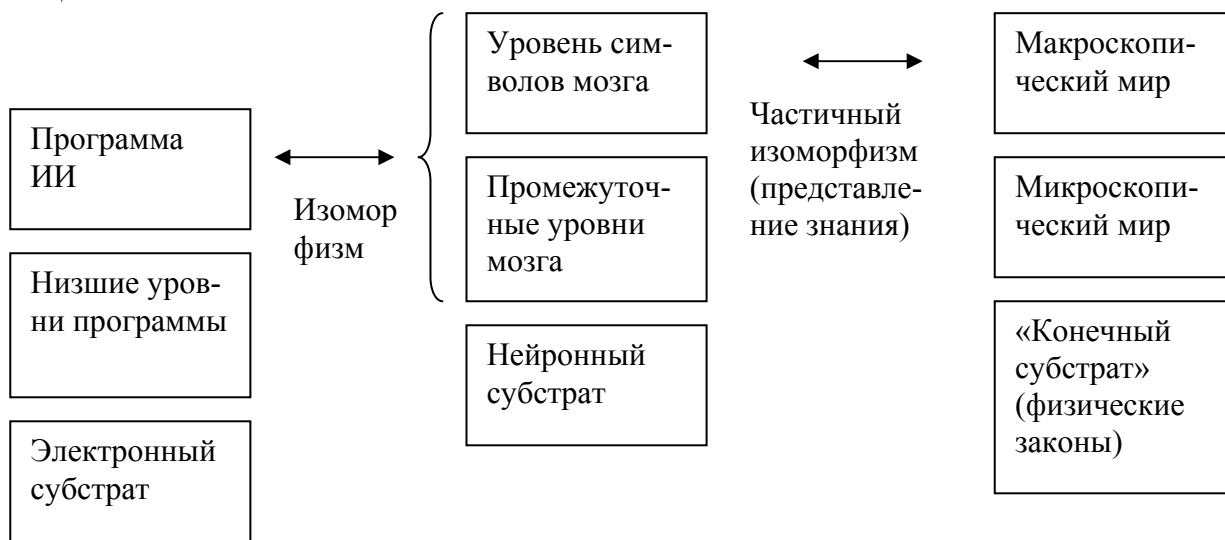
Один из основных тезисов Д. Хофштадтера

Любой аспект мышления можно рассматривать как описание на высшем уровне некой системы, которая на низшем уровне управляется простыми и даже формальными правилами.

⁹ Витгенштейн Л. Философские работы. Ч. 1. М.: Изд-во «Гнозис», 1994. – 612 с.

Предполагаемые отношения между ИИ, естественным разумом и реальным миром

Центральным понятием в исследованиях по искусственному интеллекту является то, что уровень символов мозга может быть «выделен» из их нейронного субстрата и пересажен в другую среду, такую, например, как электроника компьютера. Пока неясно, насколько глубоко должен зайти этот процесс имитации мозга.



Параллельный прогресс в ИИ и моделирование мозга

Что является показателем интеллекта? Умение играть в шахматы? Умение переводить с одного языка на другой? Умение интегрировать? Умение доказывать теоремы? Распознавать образы?

История показывает, что в прошлом люди весьма приблизительно представляли себе, какими качествами должна обладать система, чтобы ее можно было признать разумной. Иногда кажется, что каждый новый шаг на пути создания ИИ вместо того, чтобы произвести нечто такое, что все признали бы разумным, углубляет наше понимание того, что интеллектом не является. Если разум включает возможность познания, творческие способности, эмоциональные реакции, чувство красоты, самосознание, то специалистам по ИИ еще предстоит долгий путь, — и возможно, что эти черты удастся воспроизвести, только целиком промоделировав человеческий мозг.

2.2. О понимании

Лем С. о понимании

«Понимание», которое без (хотя бы капли) «интеллекта» невозможно, кажется нам (особенно в повседневных ситуациях) элементарным, но, по существу, это сложное и по-прежнему загадочное явление.

Что касается людей, они тоже могут демонстрировать «понимание на соответствующем уровне» и даже (это удастся студентам на экзаменах) могут создавать поведенчески-языковую видимость, что понимают заданные им вопросы, раз дают ответ, удовлетворяющий экзаменатора, но более тщательное изучение (что называется «прижать виновного к стенке») покажет, что это была только видимость (например, заучивание фраз на память, а не «на ум» и т.п.). Помимо этого или параллельно этому случаются ситуации, в которых можно заметить, что в течение достаточно длительного промежутка времени происходит приближение к пониманию: что есть «так-то и так»¹⁰.

Дойч Д. о понимании

Чем же отличается понимание от простого знания? Что есть объяснение, в отличие от простой формулировки факта, коей являются правильное описание или предсказание? На практике мы обычно достаточно быстро чувствуем разницу. Мы осознаем, когда чего-то не понимаем, даже если мы можем точно описать это и дать этому точное предсказание (например, течение известной болезни неизвестного происхождения), и также мы знаем, что объяснение поможет нам лучше понять это. Но дать точное определение понятий «объясне-

¹⁰ Кудрявцев Л.Д., профессор МФТИ: «Вызывает особое беспокойство то, что после окончания средней школы многие не умеют отличать то, что они понимают, от того, что они не понимают. Люди, которые не научились правильно думать, логически рассуждать, которые считают, что они понимают то, что они на самом деле не понимают, могут представить серьезную опасность для общества при самых добрых их намерениях. Весьма вероятно, что бедственное положение России, в которой она оказалась в настоящее время, не является следствием сознательных действий кого-то, а произошло благодаря людям, которые не понимали, что они делают, так как в свое время их не научили отдавать отчет в том, что они в действительности понимают и чего не понимают, что они в действительности знают и чего не знают».

ние» или «понимание» сложно. Грубо говоря, они скорее отвечают на вопрос «почему», чем на вопрос «что»; затрагивают внутреннюю суть дел; описывают реальное, а не кажущееся состояние вещей; говорят о том, что должно быть, а не что случается; определяют законы природы, а не эмпирические зависимости. Эти понятия можно отнести к связности, утонченности и простоте в противоположность произвольности и сложности, хотя ни одному из этих понятий также нельзя дать простое определение. Но в любом случае, понимание – это одна из высших функций человеческого мозга и разума, и эта функция уникальна. Многие другие физические системы, например, мозг животных, компьютеры и другие машины, могут сравнивать факты и действовать в соответствии с ними. Но в настоящее время мы не знаем ничего, кроме человеческого разума, что было бы способно понять объяснение или желало бы получить его прежде всего. Каждое открытие нового объяснения и каждое понимание существующего объяснения зависят от уникальной человеческой способности мыслить творчески.

Возможно, типичную теорию из учебной программы современного студента понять сложнее, чем любую из эмпирических зависимостей древнего строителя; но современных теорий гораздо меньше, а их объяснительная способность придает им такие качества, как красота, внутренняя логика и связь с другими предметами, благодаря которым эти теории проще изучать¹¹.

О значении и понимании

Мы понимаем друг друга, поскольку вкладываем в наши слова одни и те же значения. К каким последствиям может привести буквальное истолкование наших слов см. литературный пример: У.У. Джекобс «Обезьянья лапа»¹².

¹¹ Дойч Д. Структура реальности. – Москва; Ижевск: РХД, 2001.

¹² Антология фантастической литературы / Сост. Х.Л. Борхес, А. Бьой Касарес, С. Окампо. – СПб.: Амфора, 2001. – 367 с.

Компьютер и Буриданов осел

Известен случай успешной атаки пары аргентинских самолетов-торпедоносцев на английский эсминец «Шелфилд», закончившейся нанесением серьезного урона этому кораблю. Из-за ошибок в программном обеспечении установленная на нем система противовоздушной обороны не смогла выбрать цель, которую полагалось сбивать первой, поскольку атакующие самолеты летели слишком близко друг к другу.

Может ли существовать изоморфизм между мозгами?

Мы можем оставить всякую надежду найти абсолютный изоморфизм между людьми; однако, с другой стороны, ясно, что некоторые люди мыслят более похоже, чем другие. Кажется естественным заключить, что между мозгами людей, которые мыслят схожим образом, существует некоторый частичный изоморфизм на уровне программ – в частности, изоморфизм между (1) репертуаром символов и (2) способами их использования.

Насколько он частичный – говорит пример с Васей и сортировкой.

– Какие могут быть формальности между друзьями! Вася, сделай мне программу сортировки.

– А что такое сортировка?

– Мне надо, чтобы я вводил любые числа, а программа выдавала УПОРЯДОВЕННЫЕ числа.

(через неделю)

– Ты что, Вася! Я ввожу 5 4 7 6, а твоя программа выдает 1 2 8 9.

– Так бы и сказал, что она должна использовать ВВЕДЕННЫЕ числа.

(через неделю)

– Ты что, Вася! Я ввожу 5 4 7 6, а она выдает 4 5 6.

– Так бы и сказал, что ВСЕ числа должны присутствовать.

(через неделю)

– Ты что, Вася! Я ввожу 5 4 7 6, а она выдает 4 5 6 7 и 8 и 9.

– Я выдал все, а от себя ДОБАВИЛ, по дружбе, чтобы ты от меня отстал, наконец.

(через неделю)

– Ты что, Вася! Я ввожу 5.4 и 7.6, а она даже два числа отказывается сортировать.

– А откуда я знал, что тебе НЕ ТОЛЬКО целые надо сортировать? Может тебе завтра взбредет комплексные сортировать?! Последний раз!!!

(через неделю)

– Ты что, Вася! Программа больше девяти чисел не сортирует...

Р. Пенроуз

Почему роботы значительно хуже справляются с элементарными, «бытовыми», действиями, нежели со сложными задачами, для выполнения которых требуются высококвалифицированные люди-специалисты?

Пониманию соответствуют **невычислимые** процессы одинаковой природы, вне зависимости от того, идет ли речь о подлинно математическом восприятии, скажем, бесконечного множества натуральных чисел или всего лишь осознании того факта, что предметом удлиненной формы можно подпереть открытое окно, о понимании того, какие именно манипуляции следует производить с куском веревки для того, чтобы привязать или, напротив, отвязать уже привязанное животное, о постижении смысла слов «счастье», «битва» или «завтра» и, наконец, о логическом умозаключении относительно вероятного местонахождения правой ноги Авраама Линкольна, если известно, что левая его нога пребывает в настоящее время в Вашингтоне, — это только некоторые из примеров, оказавшихся на удивление мучительными для одной реально существующей ИИ-системы!

Такого рода невычислимые процессы лежат в основе всякой деятельности, результатом которой является непосредственное осознание чего-либо. Именно это осознание позволяет нам визуализировать геометрию движения деревянного бруска, топологические свойства веревки (в трехмерном простран-

ве) или же «связность» Авраама Линкольна. Оно также позволяет нам получить до некоторой степени прямой доступ к опыту другого человека, с помощью чего мы можем «узнать», что этот другой скорее всего подразумевает под такими словами, как «счастье», «битва» или «завтра», несмотря даже на то, что предлагаемые в процессе общения объяснения зачастую оказываются недостаточно адекватными.

Передать «смысл» слов от человека к человеку все же возможно, однако не с помощью объяснений различной степени адекватности, а лишь благодаря тому, что собеседник уже, как правило, имеет в сознании некий общий образ возможного смысла этих слов (т.е. «осознает» их), так что даже очень неадекватных объяснений обычно бывает вполне достаточно для того, чтобы человек смог «уловить» верный смысл. Именно наличие такого общего «осознания» делает возможным общение между людьми.

И именно этот факт ставит неразумного управляемого компьютером робота в крайне невыгодное положение. (В самом деле, уже самый *смысл* понятия «смысл слов» изначально воспринимается нами как нечто само собой разумеющееся, и поэтому совершенно непонятно, каким образом *такое* понятие можно сколько-нибудь адекватно описать нашему неразумному роботу.)

Смысл можно передать лишь от человека к человеку, потому что все люди имеют схожий жизненный опыт или аналогичное внутреннее ощущение «природы вещей». Можно представить «жизненный опыт» в виде своеобразного хранилища, в которое складывается память обо всем, что происходит с человеком в течение жизни, и предположить, что нашего робота не так уж и сложно таким хранилищем оснастить. Думается, что это не так: **ключевым моментом** здесь является то, что рассматриваемый субъект, будь то человек или робот, должен свой жизненный опыт *осознавать*.

2.3. Тезис Чёрча

Тезис Чёрча. Тавтологическая версия

Математические задачи можно решать только математическими методами¹³.

Тезис Чёрча. Стандартная версия

Предположим, что существует метод, при помощи которого разумное существо может разделять числа на два класса. Предположим также, что этот метод всегда приводит к ответу за конечное время и что этот ответ – всегда один и тот же для одного и того же числа. В таком случае существует некоторая программа на универсальном языке (то есть некая частично-рекурсивная функция), которая будет давать точно такие же ответы, как и разумное существо.

Можно различать коллективные и индивидуальные мыслительные процессы.

Тезис Чёрча. Версия коллективных процессов

Предположим, что существует метод, при помощи которого разумное существо может разделять числа на два класса. Предположим также, что этот метод всегда приводит к ответу за конечное время и что этот ответ – всегда один и тот же для одного и того же числа. Если этот метод может быть эффективно сообщен одним разумным существом другому при помощи языка, то в таком случае существует некоторая программа на универсальном языке (то есть некая частично-рекурсивная функция), которая будет давать точно такие же ответы, как и разумное существо.

¹³ Вопрос: насколько хорошо неформальное и интуитивное понятие эффективно вычислимой функции отражено в различных формальных описаниях? Чёрч, Тьюринг и Марков, каждый в соответствии со своим подходом, выдвинули утверждение (тезис) о том, что класс определенных ими функций совпадает с неформально определенным классом вычислимых функций. Доказано, что все эти утверждения логически эквивалентны. Название «тезис Чёрча» теперь применяется к этим и аналогичным им утверждениям.

Эта версия более слабая, чем предыдущая. Она говорит, что коллективные человеческие способы вычисления можно моделировать на компьютере, но ничего не говорит о такой возможности для любого индивидуального метода.

Пример очень мощного индивидуального разума:

Сриниваса Рамануджан

Индийский математик (1887–1920) Сриниваса Рамануджан не получил специального математического образования. Его доказательствам не хватало строгости. Иногда он просто называл результат, полученный, по его словам, чисто интуитивно, без какого бы то ни было сознательного поиска. Часто Рамануджан говорил, что богиня Намагари сообщила ему ответ во сне. Некоторые из его «интуитивных» теорем оказались ложными.

Пример формулы, полученной Рамануджаном:

$$\frac{1}{p} = \frac{2\sqrt{2}}{9801} \sum_{n=0}^{\infty} \frac{(4n)! [1103 + 26390n]}{(n!)^4 396^{4n}},$$

каждое слагаемое формулы даёт 8 точных десятичных цифр.

Рамануджан непостижимо «чувствовал» числа; вот одно из обнаруженных им трансцендентных, но «почти целых» чисел:

$$e^{p\sqrt{163}} = 262537412640768743.9999999999925.....$$

Один из самых знаменитых результатов Рамануджана связывает не вычислимые по отдельности (даже через числа π и e) слагаемые

$$A = 1 + \frac{1}{1 \cdot 3} + \frac{1}{1 \cdot 3 \cdot 5} + \dots$$

$$B = \frac{1}{1 + \frac{1}{1 + \frac{2}{1 + \frac{3}{1 + \dots}}}}$$

удивительной формулой для их суммы: $A + B = \sqrt{\frac{pe}{2}}$.

Годфри Харди¹⁴:

«Его память и его способности к вычислениям были весьма необычны, но их нельзя было назвать «ненормальными». Если ему нужно было перемножить два больших числа, он делал это обычным способом – правда, с необычайной скоростью и аккуратностью – но не быстрее и не аккуратнее, чем любой другой математик, кто от природы быстро соображает и имеет опыт в вычислениях.

Кроме памяти, терпения и способности к вычислениям он обладал такой способностью к обобщениям и к быстрому изменению своих гипотез и таким чувством формы, что ему не было равных в его области.

В его трудах не было той простоты и неизбежности, которая отличает величайшие математические открытия; они были бы более великими, если бы они были менее странными. Но зато его работы имели нечто, чего не может отрицать никто: они были глубоко и непобедимо оригинальны. Возможно, что он был бы более великим математиком, если бы его «поймали» и «приручили» в юности: он открыл бы много нового и, без сомнения, более важного. С другой стороны, он был бы менее похож на Рамануджана и более – на европейского профессора, и потеря от этого могла бы быть больше, чем выигрыш».

Тезис Чёрча. Версия Харди

На низшем уровне все математики изоморфны.

Утверждается, что умственные способности всех математиков совпадают (и если принять версию коллективных процессов, то отсюда следует версия изоморфизма).

Харди:

«Меня часто спрашивают, не было ли у Рамануджана какого-нибудь особого секрета, не пользовался ли он методами иного типа, отличными от методов других математиков, и было ли его мышление действительно необычным. Не могу ответить на эти вопросы с достаточной уверенностью, но лично я в это не

¹⁴ Харди Г. Двенадцать лекций о Рамануджане. – М.: Институт компьютерных исследований, 2002. – 336 с.

верю. Я считаю, что все математики в основном думают примерно одинаково и что Рамануджан не является исключением».

В голове людей-калькуляторов («гениальных идиотов») не происходит ничего сверхъестественного – просто их мозг совершает промежуточные действия очень быстро и уверенно.

Тезис Чёрча. Версия изоморфизма

Предположим, что существует метод, при помощи которого разумное существо может разделять числа на два класса. Предположим также, что этот метод всегда приводит к ответу за конечное время и что этот ответ – всегда один и тот же для одного и того же числа. В таком случае существует некоторая программа на универсальном языке (то есть некая частично-рекурсивная функция), которая будет давать точно такие же ответы, как и разумное существо. Более того, мыслительный процесс и эта программа будут изоморфны в том смысле, что на каком-то уровне будет существовать соответствие между операциями, выполняемыми компьютером и мозгом.

Если не букву, то дух версии изоморфизма можно передать, говоря, что гениальный идиот, вычисляя, скажем, логарифм π , проделывает операции, изоморфные операциям карманного калькулятора, решающего ту же задачу. Изоморфизм существует на уровне арифметических действий, а не на уровне нейронов мозга и электроники калькулятора¹⁵.

Тезис Чёрча. Микроскопическая версия

Поведение отдельных компонентов человеческого существа может быть промоделировано на компьютере. Следовательно, поведение отдельных компонентов (обычно ими считаются клетки) может быть вычислено с помощью частично-рекурсивной функции с любой степенью точности, если дано дос-

¹⁵ Одну и ту же задачу обычно можно решить многими способами. Какой способ выбирается, по-видимому, мало связано с интеллектом. Пример: решение задачи о мухе, летающей между идущими навстречу друг другу поездами. Обычный человек для вычисления пройденного пути мухой умножает скорость мухи на время (а время такое же, как для поездов от отправления до встречи), а выдающийся математик Норберт Винер решил дифференциальное уравнение.

таточно точное описание как внутреннего состояния данного компонента, так и его окружения.

Эта версия тезиса утверждает, что процесс мышления, хотя он и имеет больше уровней организации, не более загадочен, чем, скажем, процесс пищеварения. Короче, предполагается, что мозговые процессы, в принципе, можно понять.

Ниже дано резюме микроскопического тезиса Чёрча в макроскопическом виде.

Тезис Чёрча. Редукционистская версия

Все процессы, происходящие в мозгу, опираются на субстрат, поддающийся вычислению.

Это утверждение является, пожалуй, сильнейшей теоретической поддержкой для возможности создания искусственного интеллекта.

Тезис Чёрча. Версия души

Некоторые происходящие в мозгу процессы могут быть приблизительно воспроизведены на компьютере – но не большинство этих процессов и, безусловно, не самые интересные из них. Но даже если бы удалось моделировать все процессы мозга, то все равно осталось бы объяснить душу, на что не способен никакой компьютер.

Иррациональное и рациональное могут сосуществовать на разных уровнях

Мысль «о несовместимости иррационального с самим духом компьютеров» основана на серьезном смешении уровней. Это ошибочное мнение опирается на идею, что поскольку компьютеры – безошибочно функционирующие машины, то они должны быть «логичными» на всех уровнях. Однако совершенно очевидно, что компьютер может быть запрограммирован таким образом, чтобы напечатать серию нелогичных высказываний – или, для разнообразия, несколько высказываний со случайными значениями истинности. И все же,

следуя подобным инструкциям, компьютер не совершит никакой ошибки! Наоборот, ошибкой было бы, если бы компьютер выдал что-либо отличное от высказываний, которые ему приказано напечатать. Это показывает, как безошибочная работа на одном уровне может лежать в основе манипуляций символами на высшем уровне, – и цель высшего уровня может быть совершенно отлична от провозглашения истинны.

Дело в том, что значение может существовать на двух или более различных уровнях оперирующей символами системы, и вместе со значением на каждом из этих уровней может существовать истинность или ложность. Присутствие значения на данном уровне определяется тем, есть ли на этом уровне изоморфное (в какой-либо степени) отображение реальности.

Таким образом, тот факт, что нейроны никогда не ошибаются в сложении (и даже в гораздо более сложных вычислениях), совершенно не влияет на правильность заключений высшего уровня, который опирается на эту аппаратуру. Чем бы не занимался наш высший уровень – попыткой доказать коаны булева будизма или медитацией над теоремами дзеновой алгебры – нейроны нашего мозга функционируют рационально. Совершенно так же символические процессы высшего уровня, порождающие чувство красоты у нас в мозгу, полностью рациональны на безошибочно функционирующем низшем уровне; вся иррациональность, если таковая имеется, принадлежит высшему уровню и является эпифеноменом – следствием событий, происходящих на нижнем уровне.

Нет причин полагать, что безошибочное функционирование компьютерной аппаратуры не может породить таких сложных состояний высшего уровня, как замешательство, забывчивость или восприятие красоты. Для этого было бы необходимо наличие множества подсистем, взаимодействующих друг с другом согласно сложной «логике». Явным следствием этого было бы логичное или нелогичное поведение, опирающееся на скрытый уровень надежной, безошибочной аппаратуры.

Тезис Чёрча. Версия ИИ

Любые мыслительные процессы могут быть смоделированы при помощи компьютерной программы, написанной на языке, на котором возможно запрограммировать все частично-рекурсивные функции.

Эта наиболее сильная версия является фундаментом ИИ. На практике многие специалисты по ИИ верят в идею, родственную тезису Чёрча.

Тезис ИИ

По мере того как компьютерный разум прогрессирует, механизм, на котором он основан, постепенно становится все ближе к механизму, на котором основан человеческий разум. Иными словами, любой разум – лишь вариация одной и той же темы; чтобы создать настоящий разум, работники ИИ должны подойти как можно ближе к низшим уровням, к механизмам мозга, если они хотят, чтобы машины обладали теми же возможностями, что и мы.

2.4. Эпистемологический круг

Если использовать редукционизм как методологический принцип в попытках объяснить сознание, мы получаем порочный круг.

1. Примером редукционизма на психологическом уровне является бестселлер Карла Сагана «Драконы Эдема». Он пишет: «Моя основная посылка касательно мозга заключается в том, что его работа – которую мы иногда называем «разумом» – это следствие его анатомии и физиологии, и ничего более».

Подобные попытки свести человеческое поведение к его биологической основе имеют долгую историю и восходят к ранним дарвинистам и их современникам, работающим в области физиологической психологии. До XIX в. дуализм «тело–разум», центральное понятие в философии Декарта, помещал человеческий разум вне сферы биологии. Эволюционисты с их вниманием к нашей «обезьянности» сделали нас объектом биологических исследований при помощи методов, применявшихся к человекообразным обезьянам и по аналогии к другим животным. Павловская школа развила эту тенденцию, и она легла в

основу многих бихевиористских теорий. Хотя между физиологами не существует абсолютного согласия о том, насколько далеко могут заходить редукционистские объяснения, большинство из них соглашались с тем, что в наших действиях имеются гормональный, нейрологический и физиологический компоненты. Хотя объяснение, предлагаемое Саганом, и находится в русле основной традиции в психологии, оно достаточно радикально, так как постулирует возможность полного объяснения в терминах нижнего уровня.

2. В то время как различные школы психологии пытались свести свою науку к биологии, другие исследователи жизни искали еще более базовые уровни объяснения. Их взгляды выражены в сочинениях известного популяризатора молекулярной биологии Фрэнсиса Крика. Его книга «О молекулах и людях» атакует витализм, доктрину, согласно которой биологию следует объяснять через некую «жизненную силу», лежащую за пределами физики. Крик пишет: «Конечная цель современной биологии – объяснение всех биологических процессов в терминах физики и химии». Далее он объясняет, что под физикой и химией он подразумевает уровень атомов... Крик выражает позицию радикального редукционизма, взгляда, доминирующего среди целого поколения биохимиков и молекулярных биологов.

3. Теория относительности утверждает, что наблюдатели в различных системах, движущихся относительно друг друга, видят мир по-разному. Таким образом, наблюдатель оказался замешанным в определении физической реальности. Ученый теряет роль зрителя и становится активным участником изучаемой системы.

С развитием квантовой механики роль наблюдателя стала еще более важной в физической теории, определяющей физические события. Сложная система может быть описана только в терминах вероятности того или иного результата эксперимента. Чтобы узнать, каков именно результат данного эксперимента, необходимо произвести измерения. Именно эти измерения и являются физическим событием, в отличие от вероятности, являющейся математической абстракцией. Единственно простое и последовательное описание из-

мерения включает наблюдателя, осознающего результат. Таким образом, физическое событие и человеческий разум становятся неразделимы. Эта связь заставила физиков рассматривать сознание как существенную часть структуры физики.

Юджин Винер (нобелевский лауреат): «Нельзя сформулировать непротиворечивые законы квантовой механики, не включив в них сознание».

4. Теперь мы можем свести воедино перспективы трех крупных областей науки: психологии, биологии и физики. Скомбинировав идеи трех выразителей различных взглядов, Сагана, Крика и Винера, мы получаем довольно неожиданную картину.

Во-первых, человеческий разум, включая сознание и мысли о самом себе, может быть объяснен в терминах деятельности центральной нервной системы, которая, в свою очередь, может быть сведена к уровню биологической структуры и функций данной физиологической системы. Во-вторых, биологические явления могут быть полностью поняты в терминах атомной физики, т.е. в терминах действия и взаимодействия, составляющих систему атомов углерода, азота, кислорода и т.д. И наконец, атомная физика, наиболее полно понимаемая в терминах квантовой механики, должна содержать разум в качестве основного компонента системы.

Таким образом, мы шаг за шагом описали эпистемологический круг – от разума назад к разуму. Если мы не согласны с этой эпистемологической кругообразностью, нам остаются две противостоящих области: физика, утверждающая, что она полна, поскольку описывает всю природу, и психология, считающая себя всеобъемлющей, поскольку она имеет дело с разумом, единственным источником наших знаний о природе. Однако оба эти взгляда не свободны от проблем, так что нам, может быть, не мешает вернуться к кругу и взглянуть на него с большей симпатией. Хотя он и лишает нас абсолютов, но, по крайней мере, он принимает во внимание проблему «тело–разум» и предоставляет осно-

ву, на которой могут контактировать индивидуальные дисциплины. Возможно, что этот круг представляет наилучший подход к теоретической психологии¹⁶.

2.5. Точка зрения Пенроуза

Роджер Пенроуз в книге «Тени разума» выделяет четыре наиболее характерные точки зрения на вопрос о связи сознательного мышления и вычислений на компьютере.

- А. Мышление целиком и полностью является вычислением. В частности, ощущение осознания вызывается просто выполнением соответствующих вычислений.
- В. Сознание есть один из результатов физического действия мозга. Любое физическое действие может быть вычислительно смоделировано. Однако само по себе вычислительное моделирование не может вызвать осознание.
- С. Определенное физическое действие мозга вызывает осознание. Однако это физическое действие не может быть вычислительно смоделировано ни в каком разумном смысле.
- Д. Сознание невозможно объяснить ни в рамках физики, ни в рамках теории вычислений, ни вообще в рамках науки.

Пенроуз придерживается позиции С, и ее обоснованию посвящена вся его книга. По отношению к другим позициям автор исключительно корректен (в особенности это относится к D), но всегда абсолютно четко формулирует свое отношение к ним. По поводу D он замечает примерно следующее: а почему, собственно, *именно эту* проблему – что такое сознание, осознание, понимание – мы должны отказаться исследовать научными методами, которые позволили человечеству так заметно продвинуться в понимании мира, в котором мы живем?

¹⁶ Харольд Дж. Моровиц «Новое открытие разума» (Цит. по книге: Хофштадтер Д., Деннет Д. Глаз разума. – Самара: Издательский Дом «Бахрах-М», 2003. – 432 с.)

Утверждая, что смоделировать интеллект на машине нельзя, Пенроуз предлагает физический механизм, на котором, возможно, основаны наши интеллект и сознание, а может быть, и то неуловимое, что мы называем личностью человека. Основные результаты и гипотезы Пенроуза и его коллег суммированы в книге и в нескольких статьях. Их можно разделить на «отрицательную программу» и «положительную программу».

Отрицательная программа сводится к математической аргументации (на основе теоремы Геделя¹⁷) против возможности алгоритмически смоделировать разум. Понятием «разум» можно хоть как-то оперировать в формальных терминах, если иметь в виду *математическое* творчество – теоремы, вычисления, алгоритмы. Поэтому появляется возможность использовать достаточно четкие аргументы – а они-то как раз и подтверждают, что даже в математике самое существенное – то, что *не* формализуемо! Тем меньше остается надежд, что можно смоделировать другие свойства разума.

Положительная программа, строго говоря, есть всего лишь обсуждение комплекта согласованных друг с другом гипотез. Одна их часть относится к физике, другая – к нейрофизиологии, а в итоге получается вот что. Существенную роль в таком неотъемлемом свойстве разума, как *сознание*, играет некий «квантовый процесс» в так называемых *микротрубочках* нейронов мозга. Этот процесс влияет на сигналы, которыми обмениваются нейроны, внося принципиально важный ингредиент: *невычислимость* (а без нее не обойтись, если мы согласны с выводами отрицательной программы). В рамках существующей квантовой теории описать этот процесс невозможно (так как в ней все вычислимо, пусть даже и в вероятностном смысле). Можно сделать лишь некоторые количественные оценки, но до сколько-нибудь полной теории таких явлений еще далеко. Более того, Пенроуз считает, что создание этой теории должно быть связано с таким же радикальным, концептуальным пересмотром основ

¹⁷ Теорему Геделя можно сформулировать в следующем популярном виде: «Всякая достаточно сложная формальная теория несовершенна – она либо противоречива, либо недостаточна для решения всех возникающих в ней проблем».

физики, какого в свое время потребовало создание общей теории относительности. По поводу реализуемости нужных квантовых процессов в клетках мозга тоже есть лишь косвенные данные. Однако работа продолжается очень активно, и к ней начинают подключаться экспериментаторы.

2.6. Автореферентность

Специалист в области искусственного интеллекта Д. Хофштадтер считает, что сердцевиной естественного интеллекта является автореферентность (самоссылочность). С его точки зрения создание искусственного интеллекта обязательно требует реализацию автореферентности в той или иной форме.

Рассмотрим некоторые формы автореферентности. Ниже приведены автореферентные предложения, употребление и понимание которых свойственно человеку.

Несколько автореферентных фраз

- Это высказывание ложно.
- – В чем разница между невежеством и апатией?
– Я не знаю и мне это безразлично!
- Не использ. сокращений!
- Это предложение содержит ровно три ошибки.
- Чтобы у «этого предложения» появился смысл, вы должны проигнорировать кавычки в «нем».
- предложение должно начинаться с большой буквы.
- Я набор нейронных взрывов, которые происходят в вашем мозгу, когда вы читаете это предложение и думаете обо мне.
- Вы находитесь под моим контролем, поскольку вы будете читать меня, пока не дочитаете до конца.
- До тех пор, пока вы меня не прочитали, пятое слово этого предложения ни на кого не ссылалось.

- Я перевел это предложение на русский, поскольку я не смог прочитать его на санскрите.
- Если бы это предложение было на китайском, то оно бы означало что-то ещё.
- Это предложение состояло бы из семи слов, если было бы на семь слов короче.
- Ты читаешь меня?
- Я обычно думал, что я нерешительный, но теперь я в этом неуверен.
- Я никогда не допускаю ошибок. Однажды мне показалось, что я её допустил, но я ошибался.
- Это предложение предлагает своему (своим) читателю (читателям) различные вариации/альтернативы, которые он или она (или они) могут принять или/и отклонить.
- Вы, конечно, только начали читать предложение, которое только что закончили читать.
- В этом предложении есть ошибка, расположенная в канце.
- Я никогда не заканчиваю предложение словом «и».
- – Что означают слова "амнезия" и "мазохизм"?
– Не помню, хоть пытай!
- It is hard to translate this english phrase into russian.
- В этом предложении пять слов.
- В этом предложении тридцать две буквы.
- В этой фразе двадцать слогов, пятьдесят девять букв и одиннадцать слов.
- В это предложение цифра «0» входит 1 раз, «1» – 11, «2» – 2, «3» – 1, «4» – 1, «5» – 1, «6» – 1, «7» – 1, «8» – 1, «9» – 1.
- В этом предложении слово «в» встречается дважды, слово «встречается» встречается двенадцать раз, слово «дважды» встречается шесть раз, слово «двенадцать» встречается трижды, слово «предложении» встречается дважды, слово «раз» встречается четырежды, слово «слово» встречается двена-

дцать раз, слово «трижды» встречается трижды, слово «четырежды» встречается дважды, слово «шесть» встречается дважды, слово «этом» встречается дважды.

© Сумароков Стас

- Люди делятся на три типа: которые умеют считать, и которые не умеют.
- Чтобы стать успешным, нужно соблюдать следующие два правила:
Никогда не рассказывайте всего, что Вы знаете.
- Всегда помните о правильном употреблении множественных чисел.
- Я не знаю падежей.

Куины

Рассмотрим пример реализации автореферентности в программировании. **Куином** называется программа, которая на своем выходе генерирует собственный исходный текст. Этот термин предложил Д. Хофштадтер в честь американского философа и математика *Willard van Orman Quine* (1908–2000).

Доказано в теории алгоритмов, что куины существуют для любого универсального языка программирования. Это является следствием известной теоремы Клини о рекурсии (или о неподвижной точке), которую можно сформулировать в следующем виде: «Нельзя найти алгоритма, преобразующего программы, который бы по каждой программе давал бы другую (не эквивалентную ей)».

Пусть теперь дано преобразование программ:

$$F: p \rightarrow \{\text{программа, которая на любом входе печатает } p\}.$$

В силу теоремы Клини о рекурсии существует программа p такая, что $F(p) = p$, т.е. существует программа, печатающая (на любом входе) свой собственный текст.

Написать куин — хорошая и популярная задача для любителей программирования. Неформальную инструкцию на русском языке можно представить следующим образом:

напечатать два раза, второй раз в кавычках, такой текст:
 «напечатать два раза, второй раз в кавычках, такой текст:»

Написание куинов на процедурных языках требует некоторых хитростей.

Вот пример на Паскале:

```
var a:array[1..7] of string;
    i:integer;
begin
a[1]:='var a:array[1..7] of string;';
a[2]:='  i:integer;';
a[3]:='begin';
a[4]:='for i:=1 to 3 do writeln(a[i]);';
a[5]:='for i:=1 to 7 do writeln(#97#91,i,#93#58#61#39,a[i],#39#59);';
a[6]:='for i:=4 to 7 do writeln(a[i]);';
a[7]:='end.';
for i:=1 to 3 do writeln(a[i]);
for i:=1 to 7 do writeln(#97#91,i,#93#58#61#39,a[i],#39#59);
for i:=4 to 7 do writeln(a[i]);
end.
```

Чтобы понять эту программу, полезно иметь в виду соответствие между символами и кодами:

<i>a</i>	[]	:	=	;	'
97	91	93	58	61	59	39

Предложенный текст куина на Паскале имеет существенный недостаток: он использует значения кодов ASCII – но коды не являются частью самого языка Паскаль. Следующий куин исправляет этот недостаток (автор Dan Hoey: hoey@aic.nrl.navy.mil):

```
program s;const bbb='program s;const bbb';a='a';b='b';bb='');writeln(';
aa='";ab='";ba='";';
```



```

aaa='begin writeln(bbb,ab,bbb,ba,a,ab,a,ba,b,ab,b,ba,b,b,ab,bb,ba';
aba='a,a,ab,aa,aa,ba,a,b,ab,ab,aa,ba,b,a,ab,aa,ba,ba';
abb='a,a,a,ab,aaa,ba);writeln(a,b,a,ab,aba,ba);writeln(a,b,b,ab,abb,ba';
baa='b,a,a,ab,baa,ba);writeln(b,a,b,ab,bab,ba);writeln(aaa,bb';
bab='aba,bb);writeln(abb);writeln(bb,baa);writeln(bb,bab)end.';
begin writeln(bbb,ab,bbb,ba,a,ab,a,ba,b,ab,b,ba,b,b,ab,bb,ba);writeln(
a,a,ab,aa,aa,ba,a,b,ab,ab,aa,ba,b,a,ab,aa,ba,ba);writeln(
a,a,a,ab,aaa,ba);writeln(a,b,a,ab,aba,ba);writeln(a,b,b,ab,abb,ba
);writeln(b,a,a,ab,baa,ba);writeln(b,a,b,ab,bab,ba);writeln(aaa,bb
);writeln(aba,bb);writeln(abb);writeln(bb,baa);writeln(bb,bab)end.

```

Как видим, вопреки общепринятому мнению, куин на процедурном языке не обязан содержать итерацию или рекурсию.

Обобщением простого куина является косвенный или периодический куин. Речь идет о различных программах P_1, P_2, \dots, P_n ($n > 1$), где программа P_1 печатает текст программы P_2 , программа P_2 печатает текст программы P_3 , ..., а P_n -программа печатает, в свою очередь, текст программы P_1 .

Приведем пример периодического куина с периодом 2 – программы P_1 и P_2 :

```

var a:array[1..9] of string;
    i:integer;
    b:boolean;
begin
b:=TRUE;
a[1]:='var a:array[1..9] of string;';
a[2]:='    i:integer;';
a[3]:='    b:boolean;';
a[4]:='begin';
a[5]:='for i:=1 to 4 do writeln(a[i]);';
a[6]:='writeln(#98#58#61,not b,#59);';
a[7]:='for i:=1 to 9 do writeln(#97#91,i,#93#58#61#39,a[i],#39#59);';
a[8]:='for i:=5 to 9 do writeln(a[i]);';
a[9]:='end.';
for i:=1 to 4 do writeln(a[i]);
writeln(#98#58#61,not b,#59);

```

```

for i:=1 to 9 do writeln(#97#91,i,#93#58#61#39,a[i],#39#59);
for i:=5 to 9 do writeln(a[i]);
end.

```

```

var a:array[1..9] of string;
    i:integer;
    b:boolean;
begin
b:=FALSE;
a[1]:='var a:array[1..9] of string;';
a[2]:='    i:integer;';
a[3]:='    b:boolean;';
a[4]:='begin';
a[5]:='for i:=1 to 4 do writeln(a[i]);';
a[6]:='writeln(#98#58#61,not b,#59);';
a[7]:='for i:=1 to 9 do writeln(#97#91,i,#93#58#61#39,a[i],#39#59);';
a[8]:='for i:=5 to 9 do writeln(a[i]);';
a[9]:='end.';
for i:=1 to 4 do writeln(a[i]);
writeln(#98#58#61,not b,#59);
for i:=1 to 9 do writeln(#97#91,i,#93#58#61#39,a[i],#39#59);
for i:=5 to 9 do writeln(a[i]);
end.

```

2.7. Представление знаний и вывод знаний

Где находится значение сообщения?

Три уровня любого сообщения

1. Сообщение-рамка

Сообщение-рамка гласит: «Я – сообщение; расшифруйте меня, если сможете». Эта информация содержится в структурном аспекте предмета – носителя сообщения.

Понять сообщение-рамку означает признать необходимость декодирующего механизма.

2. Внешнее сообщение

Если мы видим сообщение-рамку, то наше внимание направляется на уровень 2 – внешнее сообщение. Это информация, явно переданная с помощью схем сим-

волов и общей структуры сообщения: она сообщает, как расшифровать внутреннее сообщение.

Понять внешнее сообщение означает построить – или знать, как построить – правильный декодирующий механизм для внутреннего сообщения.

3. Внутреннее сообщение

Информация, ради которой было послано сообщение.

Понять внутреннее сообщение означает извлечь значение, вложенное в сообщение его отправителем.

Сообщение внешнего уровня всегда неявно, поскольку отправитель послания не может гарантировать, что оно будет понято. Попытаться послать инструкции по расшифровке внешнего послания было бы напрасным усилием, так как они являлись бы частью внутреннего сообщения, а его можно понять только после того, как найден декодирующий механизм. Поэтому внешнее сообщение всегда представляет собой скорее набор триггеров (запускающих механизмов), чем какое-либо послание, поддающееся расшифровке.

Пример. Идею трех уровней сообщения хорошо поясняет пример бутылки, выброшенной на берег прибоем. С первым уровнем, рамкой, мы сталкиваемся, когда видим, что бутылка запечатана и внутри неё – сухой листок бумаги. Даже не видя, написано ли там что-нибудь, мы знаем, что этот предмет – носитель информации. Мы открываем бутылку и исследуем значки на бумаге. Может быть они написаны по-японски; это можно установить, узнав символы, но при этом не поняв ничего из внутреннего сообщения. Внешнее сообщение может быть передано русской фразой «Я – сообщение, написанное по-японски». Как только этот факт установлен, мы можем обратиться к внутреннему сообщению, которое может оказаться чем угодно (например, призывом о помощи).

Значение может существовать на двух или более различных уровнях оперирующей символами системы, и вместе со значением на каждом из этих уровней может существовать истинность или ложность. Присутствие значения на

данном уровне определяется тем, есть ли на этом уровне изоморфное (в какой-либо степени) отображение реальности.

Парадокс сообщения

Прежде чем понять любое сообщение, необходимо сообщение, говорящее нам, как понять это сообщение; иными словами, существует бесконечная иерархия уровней сообщений, которая не позволяет понять ни одного из них.

Но сообщения понимаются. Как же это происходит?

Это происходит потому, что наш разум не бестелесен; он расположен в физических объектах – наших мозгах. Их структура сформировалась в процессе долгой эволюции и их действия подчиняются законам физики. Поскольку они являются физическими телами, *наши мозги действуют, не нуждаясь в инструкциях к действию*. Именно на том уровне, где мозг интерпретирует входящую информацию как сообщение, перестает действовать «парадокс сообщения». По-видимому, в нашем мозгу уже есть встроенная аппаратура», позволяющая нам распознавать сообщения в некоторых объектах и затем эти сообщения декодировать.

Значение врожденно, если разум естественный

Единообразие естественного человеческого разума устанавливает общий «язык», на котором передаются рамки и внешние сообщения. Если есть что-то общее между человеческим разумом и разумом «в общем виде» (но что это – мы не знаем), то возможно общение и с нечеловеческими существами.

Основная проблема

Какое количество контекста (например, человеческой культуры), необходимого для понимания данного сообщения, может быть восстановлено на основе данного сообщения?

Пример. Является ли ДНК универсальным пусковым механизмом? Может ли ДНК вызвать фенотип, не используя соответствующего химического

контекста? Ответ на этот вопрос – нет, но это «нет» относительное. Возможно попытки (сознательные или случайные) восстановления химического контекста в течение многих миллионов лет привели бы к успеху.

Многоуровневые описания (о шахматах и не только)

Одна из самых трудных задач, стоящих перед исследователями искусственного интеллекта, – найти способ соединить эти два описания и создать систему, которая могла бы принимать один уровень описания и производить другой.

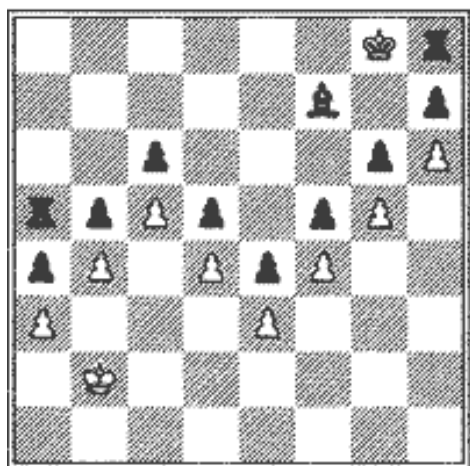
В 1940 г. датский психолог Адриан де Грот исследовал то, как шахматные мастера, в отличие от новичков, оценивают позицию. Его исследования показали, что мастера воспринимают расположение фигур *блоками*. Существует более высокий уровень описания доски, чем прямолинейное «белая пешка на e5, черная ладья на d6», и мастер каким-то образом создает мысленный образ доски на высшем уровне. Это доказывается тем, как быстро, по сравнению с новичком, мастер может восстановить какую-либо позицию из партии, после того как обоим показали доску в течение пяти секунд. Весьма показателен тот факт, что ошибки мастера касались целых групп фигур, которые он ставил в неправильное место; при этом стратегическая позиция оставалась почти той же самой – но не на взгляд новичка! Окончательным доказательством этого факта послужил тот же эксперимент, в котором на этот раз вместо настоящих позиций фигуры были расставлены как попало. В реконструкции таких случайных позиций мастера показали себя ничуть не лучше новичков.

Из этого следует, что в шахматных партиях повторяются некоторые типы ситуаций, некие определенные схемы и что именно эти схемы высшего уровня воспринимаются мастером. Он думает на *ином уровне*, чем новичок, и оперирует другим набором понятий (пример: А.И. Нимцович «Моя система»). Почти все бывают удивлены, узнав, что во время партии мастер редко заглядывает вперед дальше, чем новичок, – более того, мастер обычно рассматривает горстку возможных ходов. Трюк заключается в том, что его восприятие доски по-

добно фильтру: глядя на позицию, он буквально *не видит плохих ходов*, подобно тому, как любители не видят ходов, *противоречащих правилам*. Любой, кто хотя бы немного играл в шахматы, организует свое восприятие таким образом, что диагональные ходы ладьей, вертикальное взятие пешками и тому подобное просто не приходят ему в голову. Подобно этому, мастера создали высшие уровни организации в их восприятии позиции; в результате, рассматривать плохие ходы для них так же маловероятно, как и для большинства людей – рассматривать незаконные ходы. Это можно назвать *явной обрезкой* гигантского разветвленного дерева возможностей. С другой стороны, неявная обрезка включает рассмотрение хода и после поверхностного анализа – решение этот ход больше не анализировать.

Пример. Цель белых – ничья. Однако программа Deep Thought, играя белыми, решила взять ладью... (из книги Р. Пенроуза «Тени разума»).

Сильнейшая шахматная программа – она обыгрывала гроссмейстеров – не смогла понять элементарную позицию, очевидную для любого человека. Это была позиция, ориентированная на догадку. А создатели программы не рассматривали позиции такого типа, потому что они «неестественные». Да, эта позиция «неестественная» в контексте шахматной игры, но вполне естественная в контексте теста Тьюринга.



Это различие приложимо также и к другим видам интеллектуальной деятельности, например к занятиям математикой. Способный математик обычно

не обдумывает всякие ложные пути к доказательству нужной теоремы, как это могли бы делать менее одаренные люди; скорее всего он «нюхом чувствует» многообещающие пути и сразу направляется по ним.

Компьютерные шахматные программы, основанные на заглядывании далеко вперед, не научены думать на высшем уровне; стратегия таких машин есть «грубая сила» просчета вариантов, в надежде таким образом сокрушить любое сопротивление¹⁸. Однако это небольшой выигрыш в области интеллекта, по сравнению с открытием того, что важнейшей составляющей разума является его умение создавать многоуровневые описания сложных схем, таких, как шахматные доски, телевизионные экраны, печатные страницы или картины.

Данные и знания

При изучении интеллектуальных систем традиционно возникает вопрос – что же такое знания и чем они отличаются от обычных данных, десятилетиями обрабатываемых компьютерами. Можно предложить несколько рабочих определений, в рамках которых это становится очевидным.

Данные – это отдельные факты, характеризующие объекты, процессы и явления предметной области, а также их свойства.

При обработке на компьютере данные трансформируются, условно проходя следующие этапы:

- данные как результат измерений и наблюдений;
- данные на материальных носителях информации (таблицы, протоколы, справочники);
- модели (структуры) данных в виде диаграмм, графиков, функций;
- данные в компьютере на языке описания данных;
- базы данных на машинных носителях информации.

¹⁸ Deep Blue – компьютер фирмы IBM, победивший Каспарова, имел 256 процессоров, каждый из которых имел 4 Гб дисковой памяти и 128 Мб оперативной. Весь этот комплекс мог просчитывать более 100 000 000 ходов в секунду.

Знания основаны на данных, полученных эмпирическим путем. Они представляют собой результат мыслительной деятельности человека, направленной на обобщение его опыта, полученного в результате его практической деятельности.

Знания – это закономерности предметной области (принципы, связи, законы), полученные в результате практической деятельности и профессионального опыта, позволяющие специалистам ставить и решать задачи в этой области.

При обработке на компьютере знания преобразуются аналогично данным:

- *знания в памяти человека* как результат мышления;
- *материальные носители знаний* (учебники, методические пособия);
- *поле знаний* – условное описание основных объектов предметной области, их атрибутов и закономерностей, их связывающих;
- *знания, описанные на языках представления знаний* (продукционные языки, семантические сети, фреймы – см. далее);
- *база знаний* на машинных носителях информации.

Часто используют такое определение знаний.

Знания – это хорошо структурированные данные, или данные о данных, или метаданные.

Существует множество способов определять понятия. Один из широко применяемых способов основан на идеях интенционала и экстенционала.

Согласно принятому среди логиков словоупотреблению, слова или символы языка можно разделить на логические, или функциональные, слова («если», «и», «или», «не», «все», «некоторые»,...) и *термины* или *предикаты*, которые могут быть столь же разнообразными, как и темы обсуждения («красный», «высокий», «дед», «кислород», «посредственный автор сонетов»,...).

Каждый значащий термин или предикат языка имеет **экстенционал** (предмет либо класс предметов, к которым этот термин отсылает) и **интенцио-**

нал (особый способ выделения или определения данного предмета или класса предметов).

«Отец Челси Клинтон» и «президент Соединенных Штатов в 1995 году» именуют одного и того же человека – Билла Клинтона – и, следовательно, имеют один и тот же экстенсионал, но они указывают на общий для них предмет разными способами и, стало быть, имеют разные интенсионалы. Термин «равносторонний треугольник» вычленяет тот же самый класс объектов, что и термин «равноугольный треугольник», поэтому эти два термина имеют один и тот же экстенсионал, но очевидно, что они не означают одно и то же: один термин говорит о равенстве сторон треугольника, а другой – о равенстве его углов.

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

Экстенсионал понятия – это определение через данные, через соотнесение с понятиями более низкого уровня абстракции или перечисление фактов, относящихся к определяемому объекту.

Пример

Понятие «персональный компьютер». Его интенсионал: «Персональный компьютер – это компьютер с дружественным интерфейсом, который можно поставить на стол и купить менее чем за 1000–2000 долларов».

Экстенсионал этого понятия: «персональный компьютер – это Mac, IBM PC, ...».

Во многих случаях, отмечают логики, мы можем не обращать внимания на различие в *интенционалах* терминов и учитывать только их *экстенсионалы*. В конце концов, каким бы другим именем мы ни назвали розу, она будет благоухать точно так же, поэтому, если темой для обсуждения являются розы, то с логической точки зрения должны быть равнозначными все бесконечно разнообразные способы, которые позволяют сделать нам класс роз предметом обсуждения. Поскольку вода есть H_2O , то все истинные высказывания о воде, содержащие термин «вода», будут такими же истинными, если мы заменим этот

термин на термин «H₂O», даже если эти два термина слегка различаются по значению или интенционалу. Эта свобода особенно очевидна и полезна в таких предметных областях, как математика, где вы всегда можете воспользоваться правилом «подстановки равных на место равных», заменяя, скажем, «4×4» на «16» или наоборот, так как эти два различных термина обозначают одно и то же число. Такая свобода подстановок в лингвистических контекстах вполне уместно названа *референциальной прозрачностью* (прозрачность по ссылкам): по сути, вы можете видеть прямо сквозь термины те вещи, которые эти термины обозначают¹⁹.

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

При каждом применении и при каждом пересмотре существующего знания его конкретные формы видоизменяются. Самым часто применяемым преобразованием знания является его *конкретизация*. Например, применением теоремы, обратной к теореме Пифагора, является возможность построения прямого угла с помощью веревки, разделенной на 12 частей. Принцип бесконечного спуска является одной из конкретизаций возвратной индукции. Все многочисленные понятия гомоморфизма в алгебре являются конкретизациями общего понятия морфизмов алгебраических систем. Экономические «реформы» в России являются конкретизацией общего принципа, что частная собственность эффективнее государственной. Рассмотренные примеры показывают, что конкретизация понятий высокого уровня является сложной и не всегда успешной операцией. Даже чисто формально она неразрешима для понятий выше первого порядка.

¹⁹ Функциональные языки программирования, в которых имеет место референциальная прозрачность, называются «чистыми»; к таковым относится, например, Haskell.

Факты могут быть включены в базу имеющихся знаний, лишь если они организованы при помощи суждений более высокого уровня.

Таким образом, главной характеристикой знания является его *гибкость*, возможность выражать одно и то же в различных формах. Именно это позволяет исключительно широко применять настоящие знания, но порою затрудняет понимание обычными людьми того, что говорят действительно знающие специалисты, поскольку они не могут вообразить себе, что столь различные высказывания эксперта являются всего-навсего различными выражениями одной и той же идеи²⁰. Более того, творческие и знающие люди обычно даже не могут думать на внешнем «естественном» языке. Они вынуждены переводить внутреннюю структуру в выражения общепринятого языка, чтобы результаты рассуждений могли понять другие.

Интересны случаи, когда одни структуры выступают под видом других²¹.

Самый распространенный случай – *данные, имеющие внешнюю форму знаний*. Такую структуру ума назовем *эрудицией*. Эрудиция делает упор на *выражениях и текстах*, тогда как знание подчеркивает *идеи и контексты*. Эрудиция идеально соединяет слабейшие стороны двух форм мышления, столь же эффективно взаимоуничтожая их сильнейшие стороны. Негибкость данных сочетается у эрудированного человека с трудностями применений и зачастую с туманными формами выражения, присущими на определенном этапе развития знанию.

Единственным преимуществом эрудиции является повышение престижа человека при неглубоком общении с ним, поскольку он создает впечатление весьма знающей личности.

Знания, принявшие форму данных, – типичный учебник традиционного типа. Поскольку основные оперативные характеристики знаний теряются при данном представлении, такая фиксация знаний создает предпосылки для их

²⁰ Конечно, эти формулировки делают упор на различных ее аспектах.

²¹ Непейвода Н.Н. Online Journal «Logical Studies». – 1999. – No. 3.

фактической потери и перехода в эрудицию. В этом случае теория превращается в учение, а текст – в каноническую книгу.

Традиции составления учебников являются одной из главных причин потери знаний. Но есть и другая, еще более фундаментальная, причина. Все изложение современной науки основано на аристотелевой логике. Аристотелева (*классическая*) логика является адекватным инструментом изложения *дескриптивных* знаний, т. е. знаний самих по себе, применяемых лишь к другим знаниям и описывающих *состояние дел*. Но она обязательно приводит к отрицательным последствиям, если мы интересуемся получением умений из знаний. Эта операция называется *конструктивизацией* знания.

Знания могут быть классифицированы по следующим категориям:

- *Поверхностные* – знания о видимых взаимосвязях между отдельными событиями и фактами в предметной области.
- *Глубинные* – абстракции, аналогии, схемы, отображающие структуру и природу процессов, протекающих в предметной области. Эти знания объясняют явления и могут использоваться для прогнозирования поведения объектов.

Пример

Поверхностные знания: «Если нажать на кнопку звонка, раздается звук. Если болит голова, следует принять аспирин».

Глубинные знания: «Принципиальная электрическая схема звонка и проводки. Знания физиологов и врачей высокой квалификации о причинах, видах головных болей и методах их лечения».

Современные экспертные системы работают в основном с поверхностными знаниями. Это связано с тем, что на данный момент нет универсальных методик, позволяющих выявлять глубинные структуры знаний и работать с ними.

Кроме того, знания традиционно делятся на *процедурные* и *декларативные*. Исторически первичными были процедурные знания, т.е. знания, «раство-

ренные» в алгоритмах. Они управляли данными. Для их изменения требовалось изменять программы. Однако с развитием искусственного интеллекта приоритет данных постепенно изменялся, и все большая часть знаний сосредотачивалась в структурах данных (таблицах, списках, абстрактных типах данных), т.е. увеличивалась роль декларативных знаний.

Сегодня знания приобрели чисто декларативную форму, т.е. знаниями считаются предложения, записанные на языках представления знаний, приближенных к естественному и понятных специалисту.

Неоднозначность знаний

В области интеллектуальных систем представление знаний означает не что иное, как систематизированную методику описания на машинном уровне того, что знает человек-эксперт, специализирующийся в конкретной предметной области. Но ошибочно считать, будто представление знаний сводится к кодированию в смысле, аналогичному шифрованию. Если закодировать сообщение, подставив некоторым регулярным образом вместо одних символов другие, то полученный результат не имеет ничего общего с представлением содержания сообщения в том смысле, как это понимается в теории искусственного интеллекта, даже если полученный код легко воспринимается на машинном уровне и его легко хранить в памяти компьютера.

Обратим внимание хотя бы на то, что в таком коде сохраняется та лексическая или структурная неоднозначность, которая присуща естественному человеческому языку. Так, сообщение

«Он встретил ее на поляне с цветами»

имеет три различных толкования.

Один из парадоксов искусственного интеллекта состоит в том, что многие задачи поиска смыслового содержания, которые легко решаются человеком, очень трудно реализовать на машине, и наоборот. Рассмотрим следующую фразу:

«Молоток ударил графин, и он разбился».

К чему относится «он» в этой фразе? Для нас ответ очевиден, и мы даже не замечаем неоднозначности в этой фразе. Но как в общем смысле машина будет интерпретировать эту фразу? Предположение, что «он» относится к последнему по порядку следования в предложении существительному, не всегда срабатывает. Например:

«Графин ударился о камень, и он разбился».

Для нас совершенно очевидно, что пострадавшим в обоих случаях должен быть графин. Мы обладаем тем, что называется «предварительным знанием», но непонятно, как оно должно быть представлено в машине. Также далеко не очевидно, как собрать такого рода знания и как организовать их извлечение в конкретной ситуации. Единственное, что в этом смысле можно предложить, — сформировать огромную таблицу, состоящую из всевозможных пар объектов во вселенной, и указать в ней, какой из двух предметов более хрупкий.

Теперь рассмотрим задачу из совершенно другой области. Нужно решить, является ли некоторая логическая формула теоремой исчисления высказываний. Например, является ли теоремой формула

$$(p \& (q \supset r)) \supset ((s \vee v) \& (\neg r \supset \neg q)).$$

Оказывается, что не является, поскольку существует вариант, когда истинное значение присваивается последовательно переменным p , q , r , v и s и все выражение становится ложным. Написать программу, которая поможет компьютеру прийти к такому заключению, — задача довольно тривиальная, а сделать то же самое обычному человеку довольно сложно.

Грубо говоря, разница между этими двумя задачами состоит в том, что знание, необходимое для решения задач из области исчисления высказываний, можно выразить в компактной форме в виде правил, а знания, которые требуются для правильной интерпретации любой фразы в форме

« X ударил Y , и он разбился»,

кажутся на первый взгляд бесконечными по объему и предполагают множество исключений вроде того, что существует и пластиковый молоток, и выточенная из камня ваза, бумажная стена и т.п. и т.д. Кажется, что для решения подобных

проблем программа должна обладать чем-то вроде «здорового смысла», в то время как для решения формальных логических задач никакого здравого смысла не нужно.

Представление

Представление – множество синтаксических и семантических соглашений, которое делает возможным описание сущности. Здесь под «сущностью» понимается состояние в некоторой проблемной области, например объекты в этой области, их свойства, отношения, которые существуют между объектами. *Описание* позволяет использовать соглашения из представления для описания определенных сущностей.

Синтаксис представления специфицирует набор правил, регламентирующих объединение символов для формирования выражений на языке представления. Можно говорить о том, что выражение *хорошо* или *плохо сформировано*, т.е. о том, насколько оно соответствует этим правилам. Смысл должны иметь только хорошо сформированные выражения.

Семантика представления специфицирует, как должно интерпретироваться выражение, построенное в соответствии с синтаксическими правилами, т.е. как из его формы можно извлечь какой-то смысл. Спецификация обычно выполняется присвоением смысла отдельным символам, а затем индуцированием присвоения в более сложных выражениях.

Существуют десятки моделей (или языков) представления знаний для различных предметных областей. Большинство из них может быть сведено к следующим классам:

- продукционные модели;
- семантические сети;
- фреймы;
- формальные логические модели.

Продукционная модель

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

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

Возможна различная интерпретация продукций вида «Если A , то B »:

«Если A истинно, то B истинно»,

«Если A имеется в базе знаний, то B надо внести в базу знаний»,

«Если A – текущая ситуация, то надо делать B » и т.п.

Имеется большое число интеллектуальных систем, реализующих продукционный подход, в частности экспертные системы MYCIN и DENDRAL.

Семантическая сеть

Сеть (network) – это четверка $H = \langle A, B, P, C \rangle$,

где A — множество вершин;

B — множество имен (весов) вершин;

P — множество дуг, соединяющих пары вершин;

C — множество весов (имен) дуг.

Семантическая сеть – это сеть, в вершинах которой находятся объекты (понятия, сущности), а дуги характеризуют отношения и связи между ними. Семантические сети вначале использовались для представления смысла выражений естественного языка человека, откуда и появилось название этого класса сетей.

Для примера на рисунках приведены две семантические сети. Первая сеть (рис. 1) соответствует тексту: «В центре комнаты стоит стол. Слева от него окно. У стола глубокое удобное кресло. Недалеко от него столик с телефоном».

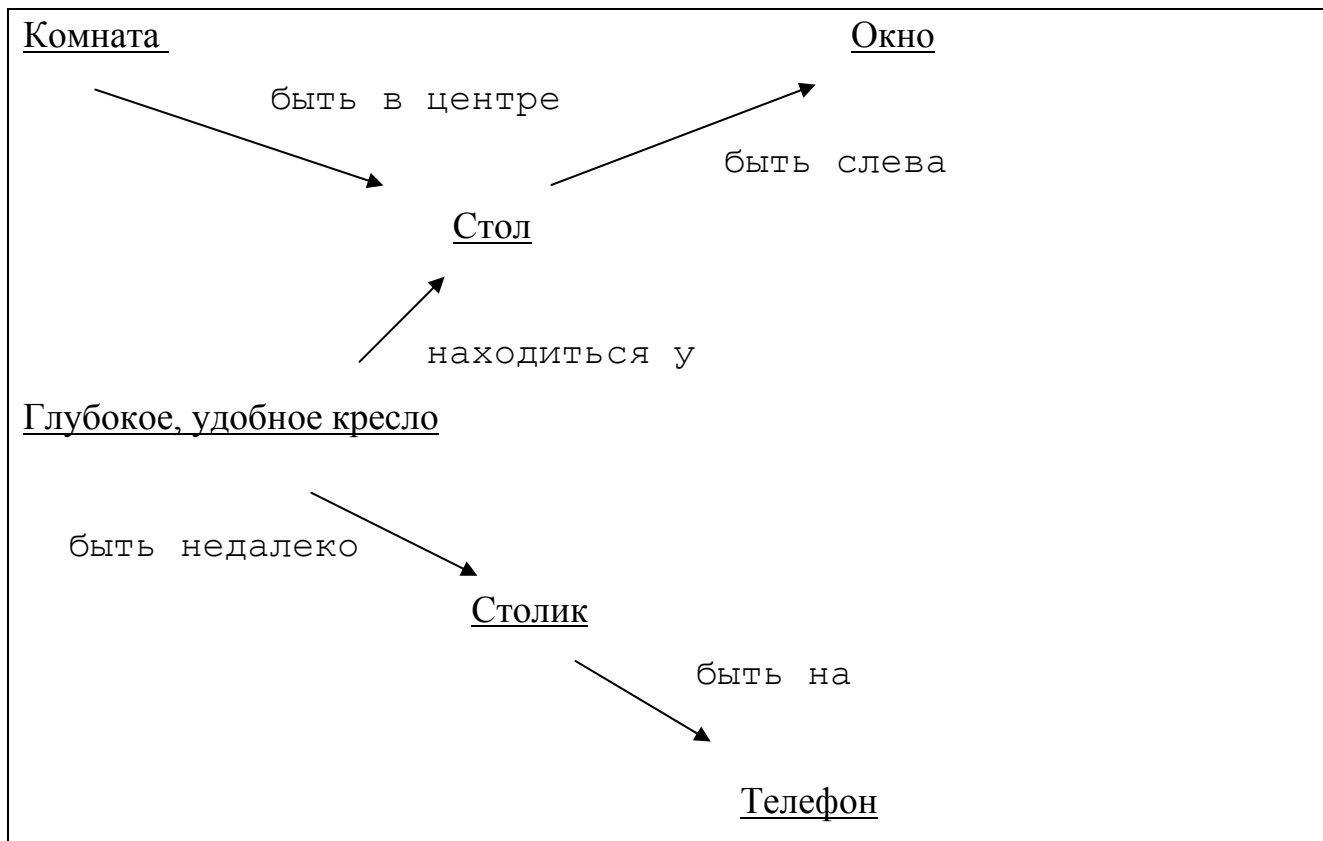


Рис. 1. Мебель в комнате

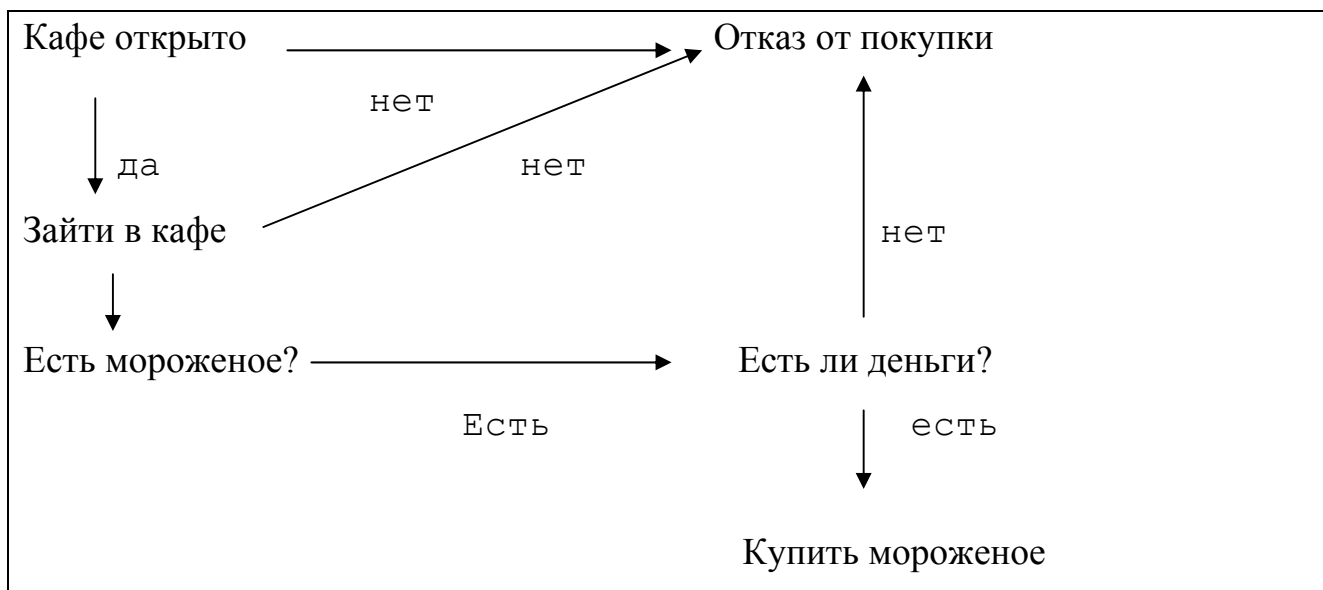


Рис. 2. Покупка мороженого в кафе

Вторая сеть (рис. 2) описывает набор процедур, необходимых для покупки мороженого в кафе. Если в вершинах первой сети находятся некоторые объекты (стол, комната и т.д.), то в вершинах второй сети названы процедуры. Дуги во

второй сети не именованы, так как все они тут имеют одинаковый смысл: «перейти к процедуре».

В соответствии с введенным определением семантической сети отсутствуют какие-либо ограничения как на типы отношений, так и на типы объектов, отображаемых в сети. В большинстве случаев многообразие объектов сети можно подразделить на три группы:

- объекты-понятия – сведения о физических и абстрактных объектах, предметной области;
- объекты-события – абстрактные или конкретные действия, которые могут привести к изменению состояния предметной области (пример – сеть на рис. 2);
- объекты-свойства – уточняют понятия и события, например, указывают характеристики понятий (глубокое, удобное), фиксируют параметры событий.

Многообразие отношений, используемых в семантических сетях, подразделяется на следующие группы:

- лингвистические отношения;
- логические отношения (конъюнкция, дизъюнкция, отрицание, импликация; отношение «перейти к процедуре» из сети на рис. 2 можно передать импликацией);
- теоретико-множественные отношения включают в себя отношения типа «множество-подмножество» («род – вид», «класс – подкласс»), «целое – часть», «элемент множества» и др.;
- квантифицированные отношения, которые подразделяются на логические кванторы общности и существования, нечеткие кванторы (много, несколько, часто и т.д.);
- функциональные связи (определяемые обычно глаголами «производит», «влияет», ...);
- пространственные (далеко от, близко от, за, под, на, ...; пример – сеть на рис. 1);
- временные (раньше, позже, в течение, ...).

Проблема поиска решения в базе знаний типа семантической сети сводится к задаче поиска фрагмента сети, соответствующего некоторой подсети, отражающей поставленный запрос к базе.

Модель семантической сети для представления знаний была предложена американским психологом Куиллианом. Основным ее преимуществом является то, что она более других соответствует современным представлениям об организации долговременной памяти у человека. Недостатком этой модели является сложность организации процедуры поиска вывода на семантической сети.

Семантическую сеть можно реализовать без ограничений с помощью языка искусственного интеллекта Лисп, но, кроме того, существуют специальные сетевые языки. Известная экспертная система PROSPECTOR использует семантическую сеть в качестве языка представления знаний.

Фреймы

Термин *фрейм* (от английского frame, что означает «каркас» или «рамка») был предложен Марвином Минским, одним из пионеров ИИ, для обозначения структуры знаний для восприятия пространственных сцен. Эта модель, как и семантическая сеть, имеет глубокое психологическое обоснование.

Фрейм – это абстрактный образ для представления некоего стереотипа восприятия.

В психологии и философии известно понятие абстрактного образа. Например, произнесение вслух слова «комната» порождает у слушающих образ комнаты: «жилое помещение с четырьмя стенами, полом, потолком, окнами и дверью, площадь 6–20 м²». Из этого описания ничего нельзя убрать (например, убрав окна, мы получим уже чулан, а не комнату), но в нем есть «дырки» или «слоты» – это незаполненные значения некоторых атрибутов – например, количество окон, цвет стен, высота потолка, покрытие пола и др.

В теории фреймов такой образ комнаты называется фреймом комнаты. Фреймом также называется и формализованная модель для отображения образа.

Различают *фреймы-образцы*, или *прототипы*, хранящиеся в базе знаний, и *фреймы-экземпляры*, которые создаются для отображения реальных физических ситуаций на основе поступающих данных.

Фрейм имеет имя (название) и состоит из частей, обычно называемых *слотами*; изображается фрейм в виде цепочки:

Фрейм = <слот 1><слот 2>...<слот N>.

Слот представляет собой пару атрибут (имя слота) – значение. В качестве значения могут выступать константные факты, выражения, содержащие переменные, ссылки на другие слоты, ссылки на фреймы и т.п.

Рассмотрим в качестве примера фрейм «Битва». Цепочка этого фрейма выглядит так:

Битва = <кто?><с кем?><когда?><где?><результат>.

Представленный фрейм является фреймом-прототипом. Во фреймах такого вида слоты имеют переменные значения. Например:

Битва = <Герой><Антигерой><утром><в чистом поле><победил>.

В этом случае, по крайней мере, слоты Герой и Антигерой – переменные, их значения могут уточняться. В результате такого уточнения получается фрейм-экземпляр:

Битва = <Царевич><Кощей Бессмертный><утром><в чистом поле><победил>.

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

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

Более адекватно структуру фрейма можно представить в виде таблицы, дополнив ее двумя столбцами.

Имя фрейма			
Имя слота	Значение слота	Способ получения значения	Присоединенная процедура

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

Существуют несколько способов получения слотом значений во фрейме-экземпляре:

- по умолчанию от фрейма-образца;
- через наследование свойств от фрейма, указанного в слоте «род – вид» (наличие связи типа «род – вид» между объектами A и B означает, что понятие A более общее, чем понятие B);
- по формуле, указанной в слоте;
- через присоединенную процедуру;
- явно из диалога с пользователем;
- из базы данных.

Характерной чертой фреймов является так называемое *наследование свойств*. И во фреймах, и в семантических сетях наследование происходит по связям, задающим отношение «род–вид». Слот «род–вид» указывает на фрейм более высокого уровня иерархии, откуда неявно наследуются, т.е. переносятся значения аналогичных слотов.

Например, в сети фреймов на рис. 3 понятие «ученик» наследует свойства фреймов «ребенок» и «человек», которые находятся на более высоком уровне иерархии. Так, на вопрос «Любят ли ученики сладкое?» следует ответ «да», так как этим свойством обладают все дети, что указано во фрейме «ребенок». Наследование свойств может быть частичным, так как возраст для учеников не

наследуется из фрейма «ребенок», поскольку указан явно в своем собственном фрейме.

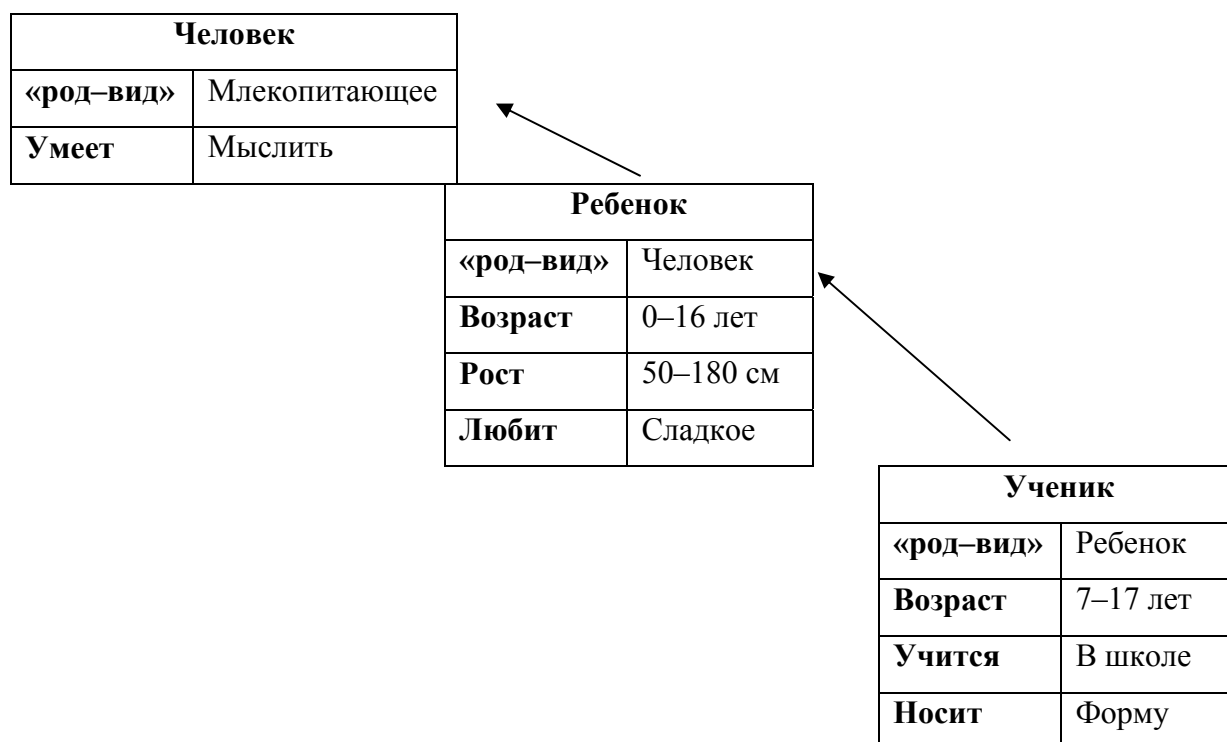


Рис. 3. Сеть фреймов

Основным преимуществом фреймов как модели представления знаний является то, что она отражает концептуальную основу организации памяти человека.

Для автоматизированной обработки фреймов создан ряд языков, таких, например, как KPL, FPL. Близкое к понятию фрейма понятие объекта реализовано в объектно-ориентированном программировании.

Частным случаем фреймов являются *скрипты*, представляющие собой описание стереотипного сценария действия с участием определенных объектов. Скрипты связаны с текущей культурой и необходимым знанием для понимания таких предложений «Я вошел в ресторан, официантка принесла мне меню». Они могут вызывать другие скрипты и обладать большими, чем фреймы, возможностями для описания динамических аспектов знания.

Логические модели

Традиционно в представлении знаний выделяют логические модели, основанные на формальных теориях с различными наборами аксиом и своими правилами вывода. Эти модели могут быть различными: от привычных исчислений высказывания и исчисления предикатов до моделей, использующих неклассические логики.

Роль логики в проблематике представления знаний и рассуждений многообразна.

- Логику можно прямо использовать для представления знаний и рассуждений.
- Она может пригодиться для ссылок и как эталон выразительности, модель компетенции, гарант элементарных логических принципов. Может помочь в точном определении альтернативных методов.
- Она определяет принципы и законы, незаменимые при решении многих проблем.
- Она позволяет анализировать смысл некоего представления знаний и обоснованность выводов.
- В этом отношении она является преимущественно средством анализа знаний и рассуждений как таковых.

Вывод знаний

Вывод знаний – получение новых информационных единиц из ранее известных.

Перечислим основные формы вывода знаний.

- Вывод абдуктивный ($Q, P \supset Q$ влечет P)

Вывод на основании абдукции – правдоподобного заключения от частного к частному.

- Вывод вероятностный

Вывод, при котором каждое выражение, используемое в нем, имеет оценку правдоподобия в виде вероятности того, что оно является истинным. При

таком выводе применяются специальные процедуры для вычисления вероятности истинного значения результирующего выражения по вероятностям посылок, используемых при выводе.

- **Вывод естественный**

Вывод, полученный на основании «здравого смысла». Эта форма вывода может либо соответствовать логическому выводу в некоторой формальной системе (но быть для человека очевидным), либо опираться на соображения, которые не укладываются в строгие рамки формальной системы.

- **Вывод здравого смысла**

Один из видов правдоподобного рассуждения, опирающийся не на основания, верные в некоторой формальной системе, а на соображения, апеллирующие к человеческому опыту, интуиции, вере.

- **Вывод индуктивный**

Вывод «от частного к общему». Позволяет на основании обобщения частных примеров некоторого явления выдвинуть гипотезу о существовании общей закономерности. В интеллектуальных системах, использующих индуктивный вывод, работают механизмы, позволяющие при формировании гипотезы приписывать ей оценку правдоподобия (например, вероятность того, что данная гипотеза является истинной). Индуктивный вывод является средством получения новых знаний в интеллектуальных системах.

- **Вывод интуиционистский**

Вывод, характерный для интуиционистской логики, не использующий, в частности, закон снятия двойного отрицания и закон исключенного третьего.

- **Вывод логический**

Последовательность рассуждений, приводящая от посылок к следствию с использованием аксиом и правил вывода. Такой вывод осуществляется в формальных аксиоматических системах, например в логике предикатов первого порядка.

- **Вывод на знаниях**

Вывод, использующий в качестве посылок выражения, хранящиеся в базе знаний. Вывод на знаниях может быть достоверным, если эти выражения являются достоверными, или правдоподобным, если они снабжены оценками правдоподобия. Как правило, процедуры вывода включают поиск необходимых фрагментов знаний для вывода, т.е. процедуру поиска по образцу.

- Вывод обратный

Вывод, при котором поиск доказательства начинается с целевого утверждения. Выясняются условия, при которых целевое утверждение является выводимым. Эти условия принимаются за новые целевые утверждения, и процесс поиска продолжается. Вывод заканчивается, когда все очередные условия оказываются аксиомами или процесс обрывается, не приведя к аксиомам. Обратный вывод реализован в алгоритме интерпретатора языка Пролог.

- Вывод по аналогии

Вывод, основанный на перенесении рассуждения из одной области на другую, похожую на исследованную. Если имеется вывод $A \Rightarrow B$ и область, в которой определено A , изоморфна области, где определено C , а область, где определено B , изоморфна области, где определено D , то вывод $A \Rightarrow B$ порождает вывод $C \Rightarrow D$. Вывод по аналогии есть частный случай правдоподобного вывода.

- Вывод по умолчанию

Один из видов правдоподобного рассуждения, где результат получается не из явно присутствующих для этого посылок, а на основе «традиции», прошлого опыта, внутренних моральных или ценностных убеждений и т.п. Вывод по умолчанию возникает тогда, когда во входной информации часть сведений отсутствует и интеллектуальная система пополняет их на основе хранящейся в ее памяти специальной информации, предназначенной для случаев неполноты входной информации. Например, во фреймах могут существовать специальные слоты, к которым система обращается за информацией, когда для проведения рассуждения чего-то не хватает.

- Вывод правдоподобный

Вывод, при котором каждый его шаг сопровождается вычислением оценки достоверности полученного утверждения. Частными случаями правдоподобного вывода являются, например, вывод вероятностный и вывод индуктивный.

- Вывод прямой

Вывод, ведущий от исходных аксиом к целевому выражению. При прямом выводе из-за неоднозначности выбора применяемых аксиом и правил вывода образуется дерево решений, и процесс нахождения цепочки, ведущей от исходных аксиом к целевому выражению, является переборным. Стандартной процедурой, используемой при обходе дерева решений, является процедура возврата – бектрекинг.

Способ решения задачи – разбиение задачи на подзадачи

Кроме конечной задачи можно обычно выделить несколько подзадач, решение которых помогает в нахождении решения основной задачи. (Разбиение продолжается рекурсивно.)

У нас нет гарантии, что метод разбиения задач на подзадачи сработает в каждом отдельном случае. Во многих ситуациях он терпит фиаско. Рассмотрим, например, следующую простую задачу. Представьте себе, что вы – собака и что хозяин только что перекинул вашу любимую кость через забор в соседний двор. Кость видно сквозь щели в заборе; она лежит на траве, и у вас текут слюнки. На расстоянии приблизительно 15 метров от кости вы видите открытую калитку в заборе. Что вы сделаете? Некоторые собаки просто подбегают к забору, поближе к кости и начинают лаять. Другие собаки бросаются к калитке (удаляясь при этом от цели) и бегут прямо к лакомому кусочку. Можно сказать, что и те и другие применяют технику разбиения задачи на подзадачи; разница в том, что они осуществляют разбиение по-разному.

Лающая собака видит подзадачи в том, чтобы

- подбежать к забору,
- попасть на ту сторону,

- подбежать к кости;

но вторая подзадача оказывается очень трудной, и она начинает лаять.

Для другой собаки подзадачи заключаются в том, чтобы

- подбежать к калитке,
- пробежать сквозь нее,
- подбежать к кости.

Обратите внимание, что все зависит от того, как вы представляете себе «пространство проблемы» – т.е. от того, что вам кажется *упрощением задачи* (движением к цели) и что – *усложнением задачи* (движением от цели).

(Литературный пример: Ярослав Гашек, «Похождения бравого солдата Швейка», глава «Будейовицкий анабазис Швейка» (анабазис – (греч.) поход в глубь страны); Швейк считал, что все дороги идут в Будейовицы и поэтому шел в этот город, отправляясь в противоположную сторону.)

Польза разбиения задачи на подзадачи (упрощения проблемы) зависит от того, каким образом эта задача представлена у вас в голове. То, что в определенном пространстве выглядит как отступление, в ином пространстве может быть революционным шагом вперед.

В каком-то смысле все возможные задачи являются лишь вариантами задачи собаки и кости. Многие проблемы разворачиваются не в физическом, а в некоем концептуальном пространстве. Если разбиение задачи на подзадачи приводит к неудаче, то это не означает, что следует отказываться от упрощения задач – напротив, это весьма полезный прием. Проблема лежит глубже: как можно выбрать подходящую внутреннюю интерпретацию задачи? В каком пространстве вы ее располагаете? Какие действия сокращают дистанцию между вами и вашей целью в выбранном вами пространстве? На математическом языке это может быть выражено как задача подходящей метрики в этом пространстве. Вам надо найти такую метрику, в которой расстояние между вами и целью было бы очень коротким.

Изменение пространства задачи – как выход из системы

При решении проблемы вы можете действовать по фиксированным, жестким правилам, используя конкретное представление пространства задачи – это значит, что вы находитесь в некоторой системе. Или вы можете изменить представление задачи – это означает, что вы выходите из системы и смотрите на проблему со стороны.

Пример1. Готовность Рамануджана изменить свои собственные гипотезы.

Пример 2. Оса Сфекс – невозможность заметить тождественности повторяющихся событий.

Пример изменения представления

Поиск хорошего представления знаний в ходе решения задачи является почти всегда необходимым этапом на пути к решению.

Задача о четырех шахматных конях

Как за минимальное число ходов изменить на противоположное положение двух черных и двух белых коней? В задаче используется шахматная доска 3×3 . Исходная позиция задачи о четырех конях представлена на рис. 4. Кони ходят (перемещаются по доске) привычным образом. Центральная клетка имеет метку *E*.

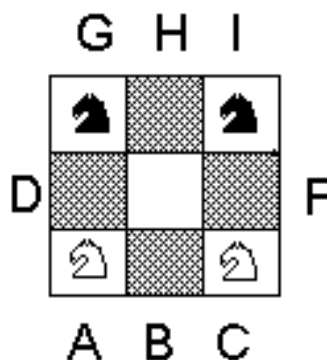


Рис. 4. Исходная позиция
в задаче о четырех конях

При знакомстве с задачей нашим первым побуждением является сделать несколько попыток по перемещению коней на настоящей шахматной доске, но интуитивно мы понимаем, что имеется слишком много вариантов получения конечной позиции, причем при таком подходе плохо используется симметрия, присущая этой задаче.

Пытаясь промоделировать условия задачи на уровне какого-то более общего представления, первое, о чем мы вспоминаем, это декартова система координат. Однако такое представление имеет существенный недостаток, связанный с трудностью записи перемещения фигур «ходом коня». Однако такие перемещения очень важны в нашем случае, и именно они-то и должны быть представлены в удобной форме.

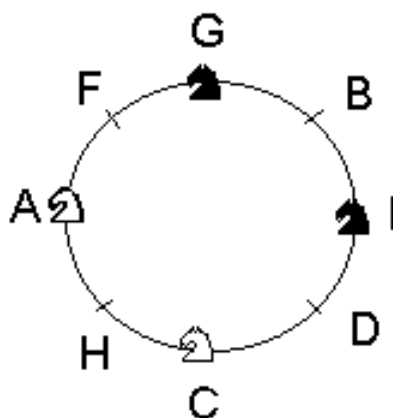


Рис. 5. Решение задачи
о четырех конях

Сама по себе шахматная доска почти не играет в задаче существенной роли, и ее физическое представление является только внешней поддержкой решения задачи, так как на самом деле единственно существенным является учет связей между ходами, сделанными при переходе от позиции к позиции. Мы знаем, что черный конь может переместиться на поле A за один ход с поля H или поля F . В свою очередь на поле H конь сможет попасть либо с того же поля A , либо с поля C , на поле C – с поля D , на него – с поля I , на поле I – с поля B ,

на него – с поля G и, наконец, на G – с поля F , начиная с которого конь сможет вновь попасть на поле A . Этот маршрут в виде окружности, проходящей в указанном порядке через все позиции, изображен на рис. 5.

Представление, получаемое с помощью этой диаграммы, наглядно и удобно для использования. На рисунке не представлено поле E , но это соответствует реальной ситуации – ни при каком ходе коня ни в одной позиции оно недостижимо. Все другие ситуации отражены, и это позволяет использовать ту же кольцевую диаграмму и для других коней.

Новая формулировка задачи теперь состоит в том, что нужно так переместить по этой окружности черных коней на A и C , а белых коней на G и I , чтобы за один ход конь перемещался на одну позицию справа или слева по окружности. Решение задачи теперь практически очевидно: достаточно лишь повернуть на пол-оборота весь ансамбль фигур так, чтобы C попало на G , A – на I , G – на C и I – на A . Это вращение, которое соответствует четырем ходам каждого коня или шестнадцати ходам всех фигур, дает решение задачи – минимальное число ходов для изменения исходной позиции на противоположную.

Изменения представления знаний во время решения задачи – основной путь ее решения.

В представлении знаний при решении математических задач существенным является то, о чем не говорится в учебниках по математике и на лекциях, а именно, каким образом и как находят доказательство.

На самом деле способ действия математика включает три различные фазы:

1. **Понять теорему**, заданную на формальном языке, т.е. перевести ее в некоторое внутреннее представление.
2. **Доказать теорему** в этом внутреннем представлении, используя для этого все накопленные и адаптированные к этому внутреннему представлению знания.
3. **Перевести** найденное еще не полное доказательство в строгое математическое доказательство, вновь введя символическую формализацию обычного математического языка.

Мы будем использовать логический подход в изучении ИИ, хотя он и имеет большие ограничения (см. раздел 1.4. Психологическая теория интеллекта). Представлять знания мы будем с помощью формул логики предикатов первого порядка и использовать логический вывод. Конкретным инструментом для программирования будет язык SWI-Prolog²². Наш выбор логического подхода определяется достаточно быстрым получением реальных результатов для хорошо формализуемых задач.

²² Свободно распространяемые реализации языка доступны на сайте www.swi-prolog.org.

3. Решение проблем посредством поиска

3.1. Задача о кубиках. Общий метод решения задач

Задача состоит в выработке плана переупорядочивания кубиков, поставленных друг на друга, как показано на рис. 6. На каждом шагу разрешается переставлять только один кубик. Кубик можно взять только тогда, когда его верхняя поверхность свободна. Кубик можно поставить на стол либо на другой кубик. Для того чтобы построить требуемый план, мы должны отыскать последовательность ходов, реализующую заданную трансформацию.

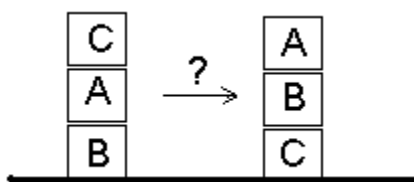


Рис. 6. Задача перестановки кубиков

Эту задачу можно представлять себе как задачу выбора среди множества возможных альтернатив. В исходной ситуации альтернатива всего одна: поставить кубик *C* на стол. После того, как кубик *C* поставлен на стол, мы имеем три альтернативы:

- поставить *A* на стол, или
- поставить *A* на *C*, или
- поставить *C* на *A*.

Ясно, что альтернативу «поставить *C* на стол» не имело смысла рассматривать всерьез, так как этот ход никак не влияет на ситуацию.

Как показывает рассмотренный пример, с задачами такого рода связано два типа понятий:

- проблемные ситуации;

- разрешенные ходы или действия, преобразующие одни проблемные ситуации в другие.

Проблемные ситуации вместе с возможными ходами образуют направленный (ориентированный) граф, называемый *пространством состояний*. Пространство состояний для только что рассмотренного примера дано на рис. 7. Вершины графа соответствуют проблемным ситуациям, дуги – разрешенным переходам из одних состояний в другие. Проблема отыскания плана решения задачи эквивалентна проблеме построения пути между заданной начальной ситуацией («стартовой» вершиной) и некоторой указанной заранее конечной ситуацией, называемой также *целевой вершиной*.

Нетрудно построить аналогичное представление в виде графа и для других популярных головоломок. Наиболее очевидные примеры – это задача о «ханойской башне» и задача о перевозке через реку волка, козы и капусты.

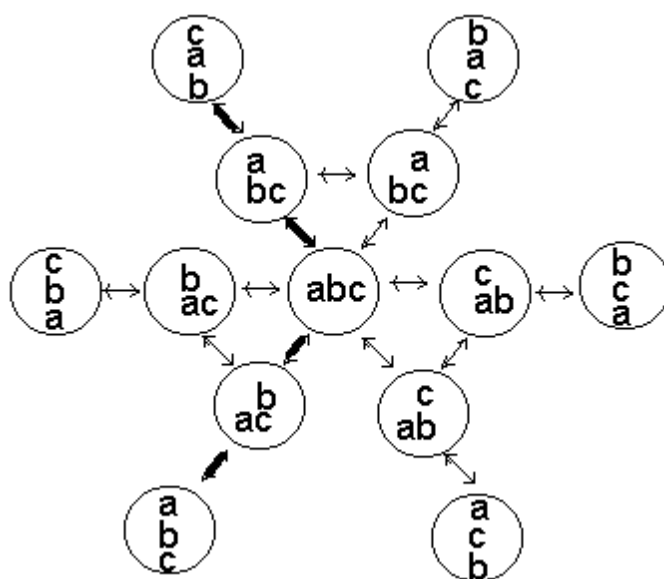


Рис. 7. Графическое представление задачи манипулирования кубиками. Выделенный путь является решением задачи

Пространство состояний некоторой задачи определяет «правила игры»: вершины пространства состояний соответствуют ситуациям, а дуги – разре-

шенным ходам, или действиям, или шагам решения задачи. Конкретная задача определяется:

- пространством состояний;
- стартовой вершиной;
- целевым условием (т.е. условием, к достижению которого следует стремиться); «целевые вершины» – это вершины, удовлетворяющие этим условиям.

Каждому разрешенному ходу или действию можно приписать его стоимость. Например, в задаче манипуляции кубиками стоимости, приписанные тем или иным перемещениям кубиков, будут указывать нам на то, что некоторые кубики перемещать труднее, чем другие. В задаче о коммивояжере ходы соответствуют переездам из города в город. Ясно, что в данном случае стоимость хода – это расстояние между двумя городами.

В тех случаях, когда каждый ход имеет стоимость, мы заинтересованы в отыскании решения минимальной стоимости. Стоимость решения – это сумма стоимостей дуг, из которых состоит «решающий путь» – путь из стартовой вершины в целевую. Даже если стоимости не заданы, все равно может возникнуть оптимизационная задача: нас может интересовать кратчайшее решение.

Прежде чем будут рассмотрены некоторые программы, реализующие классический алгоритм поиска в пространстве состояний, давайте сначала обсудим, как пространство состояний может быть представлено в прологовской программе.

Мы будем представлять пространство состояний при помощи отношения $\text{next}(X, Y)$, которое истинно тогда, когда в пространстве состояний существует разрешенный ход из вершины X в вершину Y . Мы будем говорить, что Y – это *преемник* вершины X . Если с ходами связаны их стоимости, мы добавим третий аргумент, стоимость хода: $\text{next}(X, Y, C)$. Эти отношения можно задавать в программе явным образом при помощи выбора соответствующих фактов. Однако такой принцип оказывается непрактичным и нереальным для тех типичных случаев, когда пространство состояний устроено достаточно сложно. Поэтому отношение следования next обычно определяется неявно, при помощи правил

вычисления вершин-преемников некоторой заданной вершины. Другим вопросом, представляющим интерес с самой общей точки зрения, является вопрос о способе представления состояний, т.е. самих вершин. Это представление должно быть компактным, но в то же время оно должно обеспечивать эффективное выполнение необходимых операций, в частности операций вычисления вершин-преемников, а возможно, и стоимостей соответствующих ходов.

Рассмотрим в качестве примера задачу манипулирования кубиками. Мы будем рассматривать общий случай, когда имеется произвольное число кубиков, из которых составлены столбики, – один или несколько. Число столбиков мы ограничим некоторым максимальным числом, скажем 3, чтобы задача была интереснее. Такое ограничение, кроме того, является вполне реальным, поскольку рабочее пространство, которым располагает робот, манипулирующий кубиками, ограничено.

Проблемную ситуацию можно представить как список столбиков. Каждый столбик, в свою очередь, представляется списком кубиков, из которых он составлен. Кубики упорядочены в списке таким образом, что самый верхний список находится в голове списка. «Пустые» столбики изображаются как пустые списки. Таким образом, исходную ситуацию можно записать как терм $[[c,a,b], [], []]$. Целевая ситуация – это любая конфигурация кубиков, содержащая столбик, составленный из всех имеющихся кубиков в указанном порядке. Таких ситуаций три:

$[[a,b,c], [], []]$

$[[], [a,b,c], []]$

$[[], [], [a,b,c]]$

Отношение следования можно запрограммировать, исходя из следующего правила: ситуация S_2 есть преемник ситуации S , если в S имеются два столбика C_1 и C_2 , такие, что верхний кубик из C_1 можно поставить сверху на C_2 и получить тем самым S_2 . Поскольку все ситуации – это списки столбиков, правило транслируется на Пролог так:

```

next(S, [C1,[Cub|C2]|T):-      % переставить Cub на столбик C2
    select(S,[Cub|C1],S1),      % найти первый столбик C1
    select(S1,C2,T).            % найти второй столбик C2

```

В нашем примере целевое условие имеет вид:

```
goal(S):- member([a,b,c],S).
```

3.2 Основные методы поиска

Существует много различных подходов к проблеме поиска решающего пути для задач, сформулированных в терминах пространства состояний. В качестве примера графа, представляющего пространство состояний некоторой задачи, мы будем использовать граф на рис. 8. Поскольку мы считаем, что по любому ребру мы можем двигаться в обоих направлениях, то стрелки на ребрах не указаны.

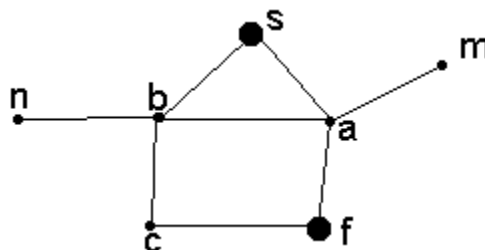


Рис.8. Пространство состояний. s – начальная вершина, f – целевая вершина

При поиске пути из начальной в целевую вершину нам необходимо:

- использовать некоторую схему учета, позволяющую упорядоченным способом исследовать все возможные пути;
- не допускать циклов.

Для лучшего представления множества путей в графе полезно будет преобразовать граф в дерево (рис. 9), при этом корнем дерева является начальная вершина, а листья дерева – это целевые или тупиковые вершины (тупики возникают в связи с нашим требованием не допускать циклов).

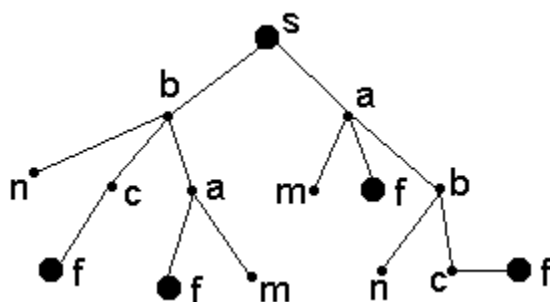


Рис. 9. Преобразование графа в дерево

Заметим, что число ярусов в полученном дереве не превосходит числа вершин в графе.

Поиск в глубину

Поиск в глубину основывается на следующей стратегии:

В каждой вершине выбирается какая-то определенная альтернатива, и ее изучение продолжается до тех пор, пока дальнейшее продвижение оказывается невозможным. Тогда процесс поиска возобновляется от ближайшей точки ветвления, имеющей неисследованные альтернативные варианты.

Поиск в глубину основывается на предположении «любой данный путь хорош, как и всякий другой». На рис. 10 показан порядок обхода вершин при поиске в глубину.

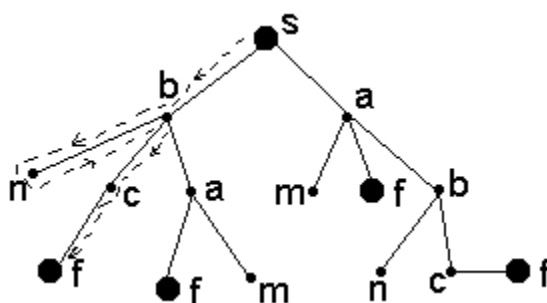


Рис. 10. Поиск в глубину

Поиск в глубину обладает следующими недостатками: а) во-первых, отыскивается не самый короткий путь; б) во-вторых, если в дереве есть бесконеч-

ные ветви и если такая бесконечная ветвь встретится, поиск никогда не закончится, даже если есть конечный путь в целевую вершину.

Поиск в глубину легко запрограммировать на Прологе.

% поиск в глубину

```
solve(Start,Solve):-    % Start - начальная вершина, Solve - искомый путь
    depth([],Start,Solve).
```

```
depth(P,X,[X|P]):-
```

```
    goal(X). % этот предикат проверяет,
              % является ли вершина целевой
```

```
depth(P,X,Solve):-
```

```
    next(X,X1),
    not(member(X1,P)),
    depth([X|P],X1,Solve).
```

Предикат `depth` использует первый аргумент для накопления пройденного пути. Вершины в пути перечисляются в обратном порядке.

Для графа на рис. 8 мы можем экономно определить отношение `next`:

```
ребра([[s,b],[s,a],[b,n],[b,c],[c,f],[a,m],[b,a],[a,f]]).
```

```
next(X,Y):-
```

```
    ребра(L),
    (member([X,Y],L);member([Y,X],L)).
```

Добавим также к программе факт `goal(f)`. Тогда имеем:

```
?- solve(s,X).
```

```
X = [f, c, b, s] ;
```

```
X = [f, a, b, s] ;
```

```
X = [f, a, s] ;
```

```
X = [f, c, b, a, s] ;
```

```
No
```

Вернемся к задаче о перестановке кубиков. Добавим предикат `printList` для удобной печати списка и предикат `run` для простоты запуска программы.

```
printList([]).
```

```
printList([H|T]):-
```

```
    write(H),nl,
```

```
    printList(T).
```

```
run:-
```

```
    solve([[c,a,b],[],[]],Solve),
```

```
    printList(Solve).
```

```
goal(S):- member([a,b,c],S).
```

Решим задачу:

```
?- run.
```

```
[[], [a, b, c], []]
```

```
[[a], [b, c], []]
```

```
[[b, a], [c], []]
```

```
[[], [c, b, a], []]
```

```
[[c], [b, a], []]
```

```
[[], [b, c], [a]]
```

```
[[b], [c], [a]]
```

```
[[], [c, b], [a]]
```

```
[[a], [c], [b]]
```

```
[[], [c, a], [b]]
```

```
[[c], [a], [b]]
```

```
[[], [a, c], [b]]
```

```
[[a], [b], [c]]
```

```
[[], [b, a], [c]]
```

```
[[b], [a], [c]]
```

```
[[], [c], [a, b]]
```

```
[[], [c], [a, b]]
```

```
[[c], [a, b], []]
[[a, c], [b], []]
[], [b, a, c], []
[[b], [a, c], []]
[[a, b], [c], []]
[[c, a, b], [], []]
```

Yes

У нас получилось поистине «ужасное» решение. Разберемся в чем причина. Во-первых, ситуации в найденном решении повторяются: например, состояния [], [c], [b,a], [[c], [b,a], []] и [[b,a], [c], []] в программе различаются. Это явилось следствием того, что в списке столбиков учитывается порядок. Для улучшения программы надо столбики рассматривать как элементы множества и заменить предикат `member` более сложным предикатом. Во-вторых, в решении встречаются два одинаковых состояния, идущих подряд,

```
[], [c], [a, b]]
[], [c], [a, b]]
```

Это уже следствие недостаточно хорошего определения предиката `next`. Дело в том, что одним из состояний-преемников для [], [c], [a, b]] является состояние [[c], [], [a, b]], которое предикат `next` выдает в виде [], [c], [a, b]]. Здесь снова при программировании `next` надо учесть, что порядок перечисления столбиков в состоянии для нас не важен.

Если известна верхняя граница длины решающего пути, то можно ограничить глубину поиска.

% Поиск в глубину с ограничением глубины

```
solve(Start,Solve):-    % Start - начальная вершина, Solve - искомый путь
    depth([],Start,Solve).
```

```
depth(P,X,[X|P]):-
    goal(X),
```



```

length([X|P],N),nl,
write('Нашли решение за '),write(N),write(' шагов '),nl.
depth(P,X,Solve):-
    maxlength(Max),
    length(P,N),
    N+1<Max,
    next(X,X1),
    not(member(X1,P)),
    depth([X|P],X1,Solve).

```

```

% maxlength(N) -> N - максимальная глубина;
maxlength(10).

```

Теперь, чтобы решить задачу (при поиске в глубину с ограничением 10), достаточно запустить цель
?- run.

Нашли решение за 10 шагов

```

[], [a, b, c], []
[], [b, c], [a]
[[b], [a], [c]]
[], [c], [a, b]
[[c], [a, b], []]
[[a, c], [b], []]
[], [b, a, c], []
[[b], [a, c], []]
[[a, b], [c], []]
[[c, a, b], [], []]

```

Yes

Поиск в глубину наиболее адекватен рекурсивному стилю программирования, принятому в Прологе. Причина этого состоит в том, что, обрабатывая цели, пролог-система сама просматривает альтернативы именно в глубину.

Поиск в ширину

При поиске в ширину целевая вершина сначала отыскивается среди всех вершин, расположенных на данном уровне, прежде чем будут исследованы ветви, отходящие вниз от этих вершин (рис. 11). Иными словами, сначала ищем решение среди путей длины один, потом – длины два и т.д.

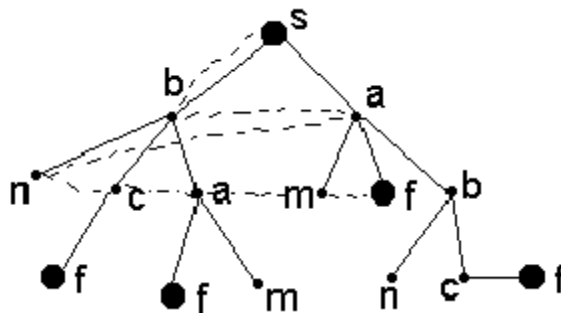


Рис. 11. Поиск в ширину

Очевидно, что этот поиск применим, даже если дерево бесконечно или практически бесконечно. К недостаткам этого поиска можно отнести неэффективность: если b – среднее число альтернатив для каждой внутренней вершины, то число путей длины n равно в среднем b^n .

Поиск в ширину программируется на Прологе немного сложнее.

% поиск в ширину

```
solve(Start,Solve):-
```

```
    width([[Start]],Solve).
```

```
width([[X|P]_],[X|P]):- goal(X).
```

```
width(Ps,Solve):-
```

```
    gener(Ps,Npath),
```

```
    width(Npath,Solve).
```

```
newPath([],_,[]).
```

```
newPath([X|T],L,[[X|L]|LL]):-  
    newPath(T,L,LL).
```

```
postList(X,L,K):-
```

```
    findall(Y, (next(X,Y),not member(Y,L)),K).
```

```
generer([[X|L]|T],Npath):-
```

```
    postList(X,L,K),
```

```
    newPath(K,[X|L],ZZ),
```

```
    append(T,ZZ,Npath).
```

Предикат `width(LL,S)` использует аргумент `LL` для хранения всех начатых и еще не рассмотренных путей. `LL` представляет из себя список путей – список списков. Рассматриваемый в текущий момент путь является головой списка `LL`. Если этот путь приходит в целевую вершину, то решение найдено (это первое правило для `width`). Иначе применяется второе правило для `width`: создается новый список путей – кандидатов к рассмотрению – и рекурсивно вызывается снова `width`.

Предикат `generer([[X|L]|T],Npath)` удлиняет на одну вершину первый путь `[X|L]` из списка путей-кандидатов и множество удлинённых путей добавляется в конец списка путей-кандидатов. Используемый в нем вызов предиката `postList(X,L,K)` создает список `K` вершин-преемников вершины `X`, не принадлежащих списку `L`.

Проверим, как работает программа. Сначала поиск в ширину в графе на рис. 8:

```
?- solve(s,Solve).
```

```
Solve = [f, a, s] ;
```

```
Solve = [f, c, b, s] ;
```

```
Solve = [f, a, b, s] ;
```

Solve = [f, c, b, a, s] ;

No

Для задачи о кубиках поиск в ширину сразу выдает самое короткое решение:

?- run.

[[], [a, b, c], []]

[[], [b, c], [a]]

[[b], [a], [c]]

[[a, b], [c], []]

[[c, a, b], [], []]

Yes

Поиск в глубину и поиск в ширину относятся к стратегиям полного перебора. Эффективность поиска повышается, если упорядочить ветви, идущие от каждой вершины, – в первую очередь при переборе выбирать наиболее перспективные ветви. Мы используем это, скажем, при подъеме в гору, выбираем тропинки, идущие вверх. Другой пример: если вам надо на автомобиле пересечь незнакомый город в направлении с севера на юг, то вы предпочитаете ехать по широким улицам, идущим близко к этому направлению. Такая стратегия называется *стратегией наискорейшего подъема (спуска)*.

Если мы отказываемся от полного перебора и отбрасываем некоторые, на наш взгляд, неперспективные ветви, то такой поиск называется *эвристическим*. Эвристический поиск быстрее находит решение, хотя и может быть неудачным. Пример: если мы должны выйти из леса в город, то следует предпочесть асфальтированные дороги проселочным.

3.3. Сведение задач к подзадачам. И/ИЛИ-графы

Представление задач в виде И/ИЛИ-графов

Мы рассмотрели подход к решению задач, основанный на поиске пути в графе пространства состояний. Однако для некоторых категорий задач представление в форме И/ИЛИ-графа является более естественным. Такое представ-

ление основано на разбиении задач на подзадачи. Разбиение на подзадачи дает преимущества в том случае, когда подзадачи взаимно независимы, а следовательно, и решать их можно независимо друг от друга. Проиллюстрируем это на примере. Рассмотрим задачу отыскания на карте дорог маршрута между двумя заданными городами, как показано на рис. 12. Не будем учитывать длину путей. Разумеется, эту задачу можно сформулировать как поиск пути в пространстве состояний. Соответствующее пространство состояний выглядело бы в точности, как карта (рис. 12): вершины соответствуют городам, дуги – непосредственным связям между городами. Тем не менее давайте построим другое представление, основанное на естественном разбиении этой задачи на подзадачи. На карте (рис. 12) мы видим также реку. Допустим, что переправиться через нее можно только по двум мостам: один расположен в городе f , другой – в городе g . Очевидно, что искомый маршрут обязательно должен проходить через один из мостов, а значит, он должен пройти либо через f , либо через g . Таким образом, мы имеем две главных альтернативы:

Для того чтобы найти путь из a в z , необходимо найти одно из двух:

- (1) путь из a в z , проходящий через f , или
- (2) путь из a в z , проходящий через g .

Теперь каждую из этих двух альтернативных задач можно, в свою очередь, разбить следующим образом:

- (1) Для того чтобы найти путь из a в z через f , необходимо:

- 1.1) найти путь из a в f и
- 1.2) найти путь из f в z .

- (2) Для того чтобы найти путь из a в z через g , необходимо:

- 2.1) найти путь из a в g и
- 2.2) найти путь из g в z .

Итак, мы имеем две главные альтернативы для решения исходной задачи: (1) путь через f или (2) путь через g . Далее, каждую из этих альтернатив можно разбить на подзадачи (1.1 и 1.2 или 2.1 и 2.2 соответственно). Здесь важно то обстоятельство, что каждую из подзадач в обеих альтернативах можно решать

независимо от другой. Полученное разбиение исходной задачи можно изобразить в форме И/ИЛИ-графа (рис. 13).

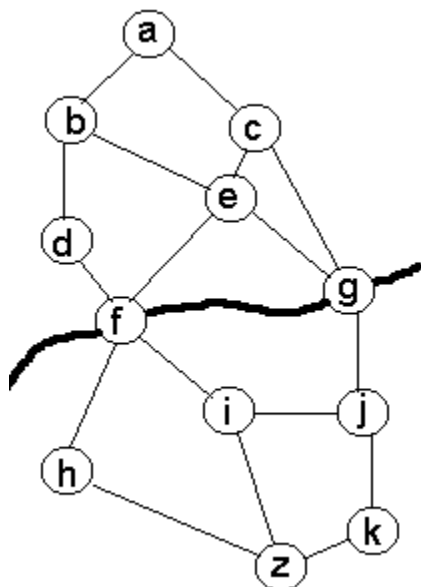


Рис. 12. Поиск маршрута из a в z на карте дорог.

Через реку можно переправиться в городах f и g .

И/ИЛИ-представление этой задачи показано на рис. 13

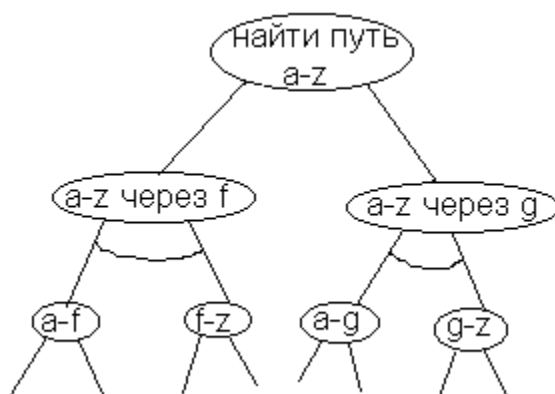


Рис. 13. И/ИЛИ-представление задачи поиска маршрута рис. 12.

Вершины соответствуют задачам или подзадачам, полукруглые дуги означают, что все (точнее, обе) подзадачи должны быть решены

Обратите внимание на полукруглые дуги, которые указывают на отношение И между соответствующими подзадачами. Граф, показанный на рис. 13, –

это всего лишь верхняя часть всего И/ИЛИ-дерева. Дальнейшее разбиение подзадач можно было бы строить на основе введения дополнительных промежуточных городов.

Какие вершины И/ИЛИ-графа являются целевыми? Целевые вершины – это тривиальные, или «примитивные» задачи. В нашем примере такой подзадачей можно было бы считать подзадачу «найти путь из a в c », поскольку между городами a и c на карте имеется непосредственная связь.

Рассматривая наш пример, мы ввели ряд важных понятий. И/ИЛИ-граф – это направленный граф, вершины которого соответствуют задачам, а дуги – отношениям между задачами. Между дугами также существуют свои отношения. Это отношения И и ИЛИ, в зависимости от того, должны ли мы решить только одну из задач-преемников или же несколько из них (рис. 14). В принципе из вершины могут выходить дуги, находящиеся в отношении И вместе с дугами, находящимися в отношении ИЛИ. Тем не менее мы будем предполагать, что каждая вершина имеет либо только И-преемников, либо только ИЛИ-преемников; дело в том, что в такую форму можно преобразовать любой И/ИЛИ-граф, вводя в него при необходимости вспомогательные ИЛИ-вершины. Вершину, из которой выходят только И-дуги, называют И-вершиной; вершину, из которой выходят только ИЛИ-дуги, – ИЛИ-вершиной.

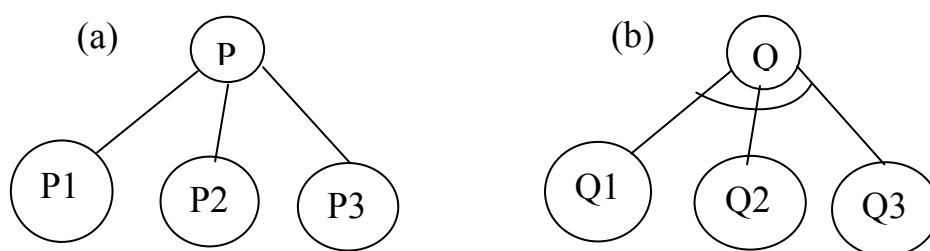


Рис. 14. (a) Решить P – это значит решить $P1$, или $P2$, или ...;

(b) Решить Q – это значит решить $Q1$, и $Q2$, и ...

Когда задача представлялась в форме пространства состояний, ее решением был путь в этом пространстве. Что является решением в случае И/ИЛИ-представления? Решение, конечно, должно включать в себя все подзадачи И-

вершины. Следовательно, это уже не путь, а дерево. Такое решающее дерево T определяется следующим образом:

- исходная задача P – корень дерева T ;
- если P является ИЛИ-вершиной, то в T содержится только один из ее преемников (из И/ИЛИ-графа) вместе со своим собственным решающим деревом;
- если P – это И-вершина, то все ее преемники (из И/ИЛИ-графа) вместе со своими решающими деревьями содержатся в T .

Поиск в И/ИЛИ-графах

Простейший способ организовать поиск в И/ИЛИ-графах средствами Пролога – это использовать переборный механизм, заложенный в самой пролог-системе. Оказывается, что это очень просто сделать, потому что процедурная семантика Пролога это и есть не что иное, как поиск в И/ИЛИ-графе. Например, И/ИЛИ-граф задачи на рис. 12 можно описать при помощи следующих предложений (предикат $a-z$ соответствует задаче «найти путь из a в z », предикат $a-z/f$ соответствует задаче «найти путь из a в z через f » и т.д.):

$a-z:-a-z/f.$ % задача $a-z$ - ИЛИ-вершина с двумя преемниками

$a-z:-a-z/g.$ % $a-z$ через f и $a-z$ через g

$a-z/f:-a-f,f-z.$ % задача $a-z/f$ - И-вершина с двумя преемниками $a-f$ и $f-z$

$a-z/g:-a-g,g-z.$

$a-f:-a-f/b.$

$a-f:-a-f/c.$

$a-f/b:-a-b,b-f.$

$a-f/c:-a-c,c-f.$

$b-f:-b-d,d-f.$

$c-f:-c-e,e-f.$

$f-z:-f-z/h.$

f-z:-f-z/i.

f-z/h:-f-h,h-z.

f-z/i:-f-i,i-z.

/* пропущены правила

для a-g и g-z

*/

a-b. b-d. d-f. a-c. c-e. e-f. f-h. h-z. f-i. i-z. % "тривиальные" задачи

Для того чтобы узнать, имеет ли эта задача решение, нужно просто спросить:

?- a-z.

Получив этот вопрос, пролог-система произведет поиск в глубину в И/ИЛИ-дереве и после того, как найдет решающее дерево ответит **Yes**.

За простоту такого метода программирования приходится расплачиваться: мы не получаем явно решающего дерева. Но этот недостаток исправим – надо определить собственную процедуру поиска в глубину для И/ИЛИ-деревя.

3.4. Игры и минимаксный принцип

Формулировка игровых задач в терминах И/ИЛИ-графов

Такие игры, как шахматы или шашки, естественно рассматривать как задачи, представленные И/ИЛИ-графами. Игры такого рода называются играми двух лиц с полной информацией. Будем считать, что существует только два возможных исхода игры: ВЫИГРЫШ и ПРОИГРЫШ. (Об играх с тремя возможными исходами – ВЫИГРЫШ, ПРОИГРЫШ и НИЧЬЯ – можно также говорить, что они имеют только два исхода: ВЫИГРЫШ и НЕВЫИГРЫШ.) Так как участники игры ходят по очереди, мы имеем два вида позиций в зависимости от того, чей ход. Давайте условимся называть участников игры «игрок» и «противник», тогда мы будем иметь следующие два вида позиций: позиция с ходом игрока («позиция игрока») и позиция с ходом противника («позиция противника»). Допустим также, что начальная позиция P – это позиция игрока.

Каждый вариант хода игрока в этой позиции приводит к одной из позиций противника $Q1, Q2, Q3, \dots$ (рис. 15).

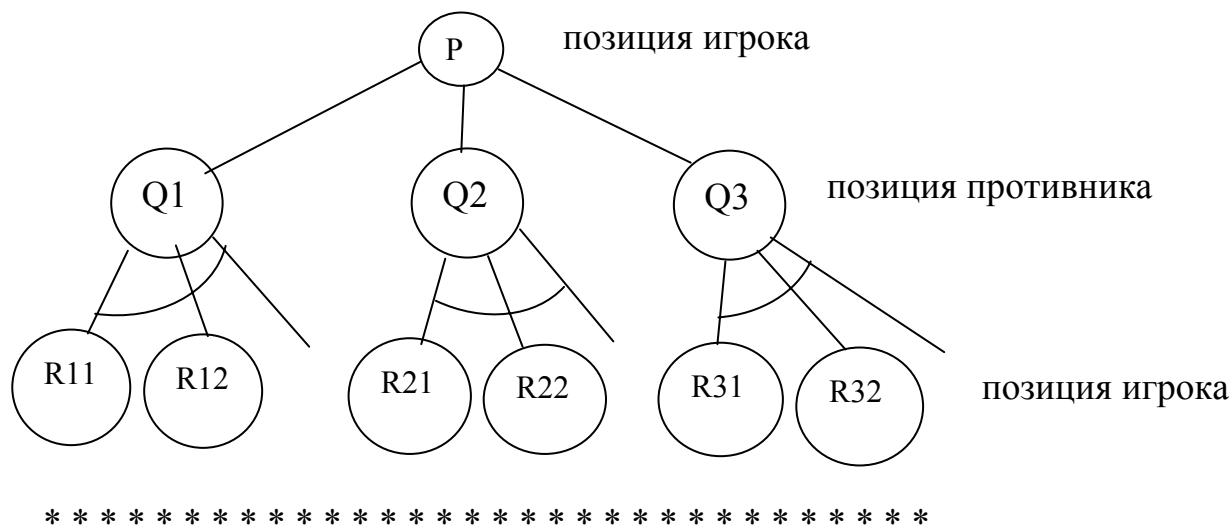


Рис. 15 Формулировка игровой задачи для игры двух лиц
в форме И/ИЛИ-дерева; участники игры: «игрок» и «противник»

Далее, каждый вариант хода противника в позиции $Q1$ приводит к одной из позиций игрока $R11, R12, \dots$. В И/ИЛИ-дереве, показанном на рис. 15, вершины соответствуют позициям, а дуги – возможным ходам. Уровни позиций игрока чередуются в дереве с уровнями позиций противника. Для того чтобы выиграть в позиции P , нужно найти ход, переводящий позицию P в выигранную позицию Qi (при некотором i). Таким образом, игрок выигрывает в позиции P , если он выигрывает в $Q1$, или $Q2$, или $Q3$, или \dots . Следовательно, P – это ИЛИ-вершина. Для любого i позиция Qi – это позиция противника, поэтому если в этой позиции выигрывает игрок, то он выигрывает и после каждого варианта хода противника. Другими словами, игрок выигрывает в Qi , если он выигрывает во всех позициях $Ri1$, и $Ri2$, и \dots . Таким образом, все позиции противника – это И-вершины. Целевые вершины – это терминальные (окончательные) позиции, выигранные согласно правилам игры, например позиции, в которых король противника получает мат. Позициям проигранным соответствуют задачи, не имеющие решения. Для того чтобы решить игровую задачу, мы должны построить решающее дерево, гарантирующее победу игрока независимо от отве-

тов противника. Такое дерево задает полную стратегию достижения выигрыша: для каждого возможного продолжения, выбранного противником, в дереве стратегии есть ответный ход, приводящий к победе.

Ниже приводится простая программа, которая определяет, является ли некоторая позиция игрока выигранной.

выигранная(P):-

терм_выигранная(P). % терминальная выигранная позиция

выигранная(P):-

not(терм_проигранная(P)), % не терминальная проигранная
% позиция

ход(P,P1), % разрешенный ход из позиции P в позицию P1

% ни один из ходов противника не ведет к не-выигрышу

not(ход(P1,P2),

not('выигранная(P2))).

Здесь правила игры встроены в предикат `ход(P,P1)`, который порождает все разрешенные ходы, а также в предикаты `терм_выигранная(P)` и `терм_проигранная(P)`, которые распознают терминальные позиции, являющиеся, согласно правилам игры, выигранными или проигранными. В последнем из правил программы, содержащем двойное отрицание, говорится: не существует хода противника, ведущего к не выигранной позиции. Другими словами, все ходы противника приводят к позициям, выигранным с точки зрения игрока.

Программа, которую мы составили, демонстрирует основные принципы программирования игр. Но практически приемлемая реализация таких сложных игр, как шахматы или го, потребовала бы привлечения значительно более мощных методов. Огромная комбинаторная сложность этих игр делает наш наивный переборный алгоритм, просматривающий дерево вплоть до терминальных игровых позиций, абсолютно непригодным. Для шахмат, например, пространство поиска имеет астрономические размеры – около 10^{120} позиций.

Минимаксный принцип

Поскольку полный просмотр игрового дерева в большинстве игр невозможен, были разработаны другие методы, предусматривающие просмотр только части дерева игры. Среди этих методов существует стандартный метод поиска, используемый в игровых (особенно в шахматных) программах и основанный на *минимаксном принципе*. Дерево игры (И/ИЛИ-граф) просматривается только вплоть до некоторой глубины (обычно на несколько ходов), а затем для всех концевых вершин дерева поиска вычисляются оценки позиций при помощи некоторой оценочной функции. Идея состоит в том, чтобы, получив оценки этих терминальных поисковых вершин, не продвигаться дальше и получить тем самым экономию времени. Далее оценки терминальных позиций распространяются вверх по дереву поиска в соответствии с минимаксным принципом. В результате все вершины дерева поиска получают свои оценки. И, наконец, игровая программа, участвующая в некоторой реальной игре, делает свой ход – ход, ведущий из исходной (корневой) позиции в наиболее перспективного (с точки зрения оценки) ее преемника.

Обратите внимание на то, что мы здесь делаем определенное различие между «деревом игры» и «деревом поиска». Дерево поиска – это только часть дерева игры (его верхняя часть), т.е. та его часть, которая была явным способом порождена в процессе поиска. Таким образом, терминальные поисковые позиции совсем необязательно должны совпадать с терминальными позициями самой игры.

Очень много зависит от оценочной функции, которая для большинства игр, представляющих интерес, является приближенной эвристической оценкой шансов на выигрыш одного из участников игры. Чем выше оценка, тем больше у него шансов выиграть, и чем ниже оценка, тем больше шансов на выигрыш у его противника. Поскольку один из участников игры всегда стремится к высоким оценкам, а другой – к низким, мы дадим им имена МАКС и МИН соответственно. МАКС всегда выбирает ход с максимальной оценкой, в противоположность ему МИН всегда выбирает ход с минимальной оценкой. Пользуясь

этим принципом (минимаксным принципом) и зная значения оценок для всех вершин «подножья» дерева поиска, можно определить оценки всех остальных вершин дерева. На рис. 16 показано, как это делается. На этом рисунке видно, что уровни позиций с ходом МАКСа чередуются с уровнями позиций с ходом МИНа. Оценки вершин нижнего уровня определяются при помощи оценочной функции. Оценки всех внутренних вершин можно определить, двигаясь снизу вверх от уровня к уровню, пока мы не достигнем корневой вершины. В результате, как видно на рис. 16, оценка корня оказывается равной 4 и, соответственно, лучшим ходом МАКСа из позиции a – $a-b$. Лучший ответ МИНа на этот ход – $b-d$ и т.д. Эту последовательность ходов называют также *основным вариантом*. Основной вариант показывает, какова «минимаксно-оптимальная» игра для обоих участников. Обратите внимание на то, что оценки всех позиций, входящих в основной вариант, совпадают.

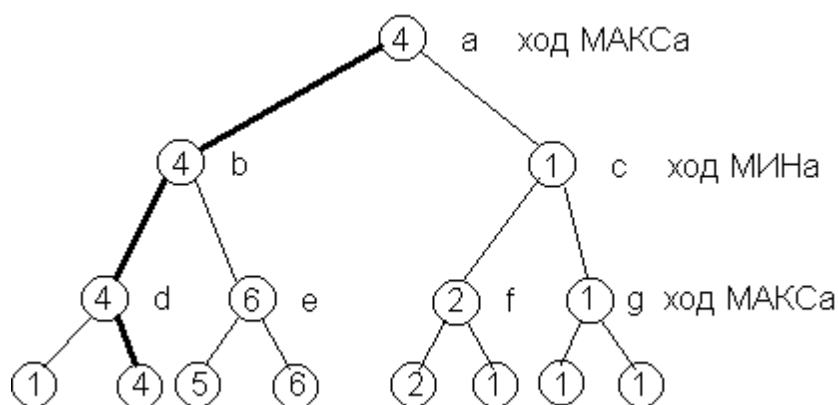


Рис. 16. Статический (нижний уровень) и минимаксные рабочие оценки вершин дерева поиска. Выделенные ходы образуют основной вариант, т.е. минимаксно-оптимальную игру с обеих сторон

Мы различаем два вида оценок: оценки вершин нижнего уровня и оценки внутренних вершин (рабочие оценки). Первые из них называют также «статическими», так как они вычисляются при помощи «статической» оценочной функции, в противоположность рабочим оценкам, получаемым «динамически» при распространении статических оценок вверх по дереву.

Правила распространения оценок можно сформулировать следующим образом. Будем обозначать статическую оценку позиции P через $v(P)$, а ее рабочую оценку – через $V(P)$. Пусть P_1, P_2, \dots, P_n – разрешенные преемники позиции P . Тогда соотношения между статическими и рабочими оценками можно записать так:

$$V(P) = v(P),$$

если P – терминальная вершина позиции дерева поиска ($n = 0$);

$$V(P) = \max_i V(P_i),$$

если P – позиция с ходом МАКСа;

$$V(P) = \min_i V(P_i)$$

если P – позиция с ходом МИНа.

Приведем упрощенную программу на Прологе, вычисляющую минимаксную рабочую оценку для некоторой заданной позиции.

% Минимаксная процедура: minimax(P,GP,V)

% P - позиция, V - ее минимаксная оценка;

% лучший ход из позиции P ведет в позицию GP

minimax(P,GP,V):-

 ходы(P,List),!, % List - список разрешенных ходов

 лучшая(List,GP,V);

 'статическая оценка'(P,V). % позиция P не имеет преемников

лучшая([P],P,V):-

 minimax(P,_,V),!.

лучшая([P1|List],GP,GV):-

 minimax(P1,_,V1),

 лучшая(List,P2,V2),

 выбор(P1,V1,P2,V2,GP,GV).

выбор($P_0, V_0, P_1, V_1, P_0, V_0$):-

'ход МИНа'(P_0), $V_0 > V_1$, !;

'ход МАКСа'(P_0), $V_0 < V_1$, !.

выбор($P_0, V_0, P_1, V_1, P_1, V_1$).

Основное отношение этой программы $\text{minimax}(P, GP, V)$, где V – минимаксная оценка позиции P , а GP – наилучшая позиция-преемник позиции P (лучший ход, позволяющий достигнуть оценки V). Предикат $\text{ходы}(P, \text{List})$ задает разрешенные ходы игры: List – это список разрешенных позиций-преемников позиции P . Предполагается, что цель ходы имеет неуспех, если P является терминальной поисковой вершиной (листом дерева поиска). Отношение $\text{лучшая}(\text{List}, GP, V)$ выбирает из списка позиций-кандидатов List «наилучшую» позицию GP . V – оценка позиции GP , а следовательно, и позиции-предка. Под «наилучшей» оценкой мы понимаем либо максимальную, либо минимальную оценку в зависимости от того, с чьей стороны ожидается ход.

4. Экспертные системы

4.1. Функции и структура экспертной системы

Экспертная система – это программа, которая ведет себя подобно эксперту в некоторой, обычно узкой, прикладной области. Типичные применения экспертных систем включают в себя такие задачи, как медицинская диагностика, локализация неисправностей в оборудовании и интерпретация результатов измерений. Экспертные системы должны решать задачи, требующие для своего решения экспертных знаний в некоторой конкретной области. В той или иной форме экспертные системы должны обладать такими знаниями. Поэтому их также называют *системами, основанными на знаниях*. Однако не всякую систему, основанную на знаниях, можно рассматривать как экспертную. Экспертная система должна также уметь каким-то образом *объяснять* свое поведение и свои решения пользователю, так же, как это делает эксперт-человек. Объяснение полученного результата опирается на тот маршрут, который сохранился в памяти системы от процесса поиска решения. Используя этот маршрут, интеллектуальная система формирует пользователю объяснение на профессиональном естественном языке, позволяющее ему представить все принципиальные шаги решения. Это особенно необходимо в областях, для которых характерна неопределенность, неточность информации (например, в медицинской диагностике). В этих случаях способность к объяснению нужна для того, чтобы повысить степень доверия пользователя к советам системы, а также для того, чтобы дать возможность пользователю обнаружить возможный дефект в рассуждениях системы. В связи с этим в экспертных системах следует предусматривать дружественное взаимодействие с пользователем, которое делает для пользователя процесс рассуждения системы «прозрачным».

Часто к экспертным системам предъявляют дополнительное требование – способность иметь дело с неопределенностью и неполнотой. Информация о поставленной задаче может быть неполной или ненадежной; отношения между объектами предметной области могут быть приближенными. Например, может

не быть полной уверенности в наличии у пациента некоторого симптома или в том, что данные, полученные при измерении, верны; лекарство может стать причиной осложнения, хотя обычно этого не происходит. Во всех этих случаях необходимы правдоподобные рассуждения.

Можно выделить два типа экспертных систем:

- для специалистов невысокого профессионального уровня (экспертная система хранит знания, полученные от специалистов экстракласса);
- для специалистов высокого класса – поиск и просмотр больших массивов информации и выполнение рутинных операций.

В самом общем случае для того чтобы построить экспертную систему, мы должны разработать механизмы выполнения следующих функций системы:

- решение задач с использованием знаний о конкретной предметной области – возможно при этом возникнет необходимость иметь дело с неопределенностью;
- взаимодействие с пользователем, включая объяснение намерений и решений системы во время и после окончания процесса решения задачи.

Каждая из этих функций может оказаться очень сложной и зависит от предметной области, а также от различных практических требований.

При разработке экспертных систем принято делить их на три модуля, как показано на рис. 17:

- (1) базу знаний;
- (2) машину логического вывода;
- (3) интерфейс с пользователем.

База знаний содержит знания, относящиеся к конкретной прикладной области, в том числе отдельные факты, правила, описывающие отношения или явления, а также, возможно, методы, эвристики и различные идеи, относящиеся к решению задач в этой прикладной области. *Машина логического вывода* умеет активно использовать информацию, содержащуюся в базе знаний. *Интерфейс с пользователем* отвечает за бесперебойный обмен информацией между

пользователем и системой; он также дает пользователю возможность наблюдать за процессом решения задач, протекающим в машине логического вывода. Принято рассматривать машину вывода и интерфейс как крупный единый модуль, обычно называемый *оболочкой экспертной системы*, или, для краткости, просто *оболочкой*.

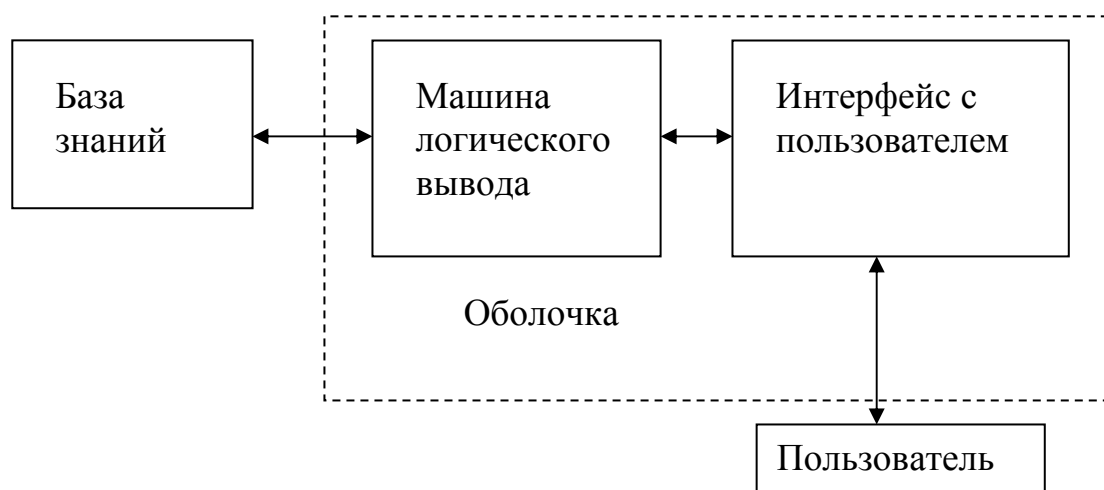


Рис. 17. Структура экспертной системы

Совокупность методов и процедур, которые применяет *инженер по знаниям*²³ при заполнении им базы знаний, называется приобретением знаний. Приобретение знаний предполагает использование источников знаний двух типов: пассивных и активных. К первым относятся официальные документы, инструкции, печатные издания, кинофотодокументы и многие другие источники, в которых содержатся сведения, важные для описания знаний о предметной области. Ко второму типу источников знаний относятся люди – специалисты в данной предметной области. Инженер по знаниям с помощью специальных психологических методик и инструментальных средств в процессе диалога получает от экспертов необходимые сведения. Все приобретенные знания для ввода в базу знаний формализуются в соответствии с требованиями той модели

²³ Инженер по знаниям – специалист, основной задачей которого является проектирование баз знаний и наполнение их знаниями о проблемной области. В процессе этой деятельности инженер по знаниям выбирает форму представления знаний, удобную для данной проблемной области, организует приобретение знаний.

знаний, которая соответствует выбранному проектировщиком системы представлению знаний.

В описанной выше структуре экспертной системы собственно знания отделены от алгоритмов, использующих эти знания. Такое разделение удобно по следующим соображениям. База знаний, очевидно, зависит от конкретного приложения. С другой стороны, оболочка, в принципе, независима от приложений. Таким образом, разумный способ разработки экспертной системы для нескольких приложений сводится к созданию универсальной оболочки, после чего для каждого приложения достаточно подключить к системе новую базу знаний. Разумеется, все эти базы знаний должны удовлетворять одному и тому же формализму, который оболочка «понимает». Практический опыт показывает, что для сложных экспертных систем наш сценарий с одной оболочкой и многими базами знаний работает не так гладко, как бы этого хотелось, за исключением тех случаев, когда прикладные области очень близки. Тем не менее даже если переход от одной прикладной области к другой требует модификации оболочки, то, по крайней мере, основные принципы ее построения обычно удается сохранить.

4.2. Продукции и неопределенность

В качестве кандидата на использование в экспертной системе можно рассматривать, в принципе, любой непротиворечивый формализм, в рамках которого можно описывать знания о некоторой проблемной области. Однако для логического программирования популярным формальным языком является язык *продукций* (правил типа «если-то»). Каждое такое правило есть, вообще говоря, некоторое условное утверждение, но возможны и другие различные интерпретации. Вот примеры:

- если предварительное условие P , то заключение (вывод) C ;
- если ситуация S , то действие A ;
- если выполнены условия $C1$ и $C2$, то не выполнено условие C .

Продукции обычно оказываются весьма естественными выразительными средствами представления знаний. Кроме того, они обладают следующими привлекательными свойствами:

- *модульность*: каждое правило описывает небольшой, относительно независимый фрагмент знаний;
- возможность *инкрементного наращивания*: добавление новых правил в базу знаний происходит независимо относительно других правил;
- *удобство модификации* (как следствие модульности): старые правила можно изменять и заменять на новые независимо относительно других правил;
- применение правил способствует *прозрачности* системы.

Последнее свойство – это важное отличительное свойство экспертных систем. Под прозрачностью мы понимаем способность системы к объяснению принятых решений и полученных результатов. Применение продукций облегчает получение ответов на следующие основные типы вопросов пользователя:

- 1) Вопросы типа «как»: *Как вы пришли к этому выводу?*
- 2) Вопросы типа «почему»: *Почему вас интересует эта информация?*

Продукцию часто применяют для определения логических отношений между понятиями предметной области. Про чисто логические отношения можно сказать, что они принадлежат к «категорическим знаниям», «категорическим» потому, что соответствующие утверждения всегда истинны или ложны. Однако многие области экспертных знаний не являются категорическими. Как правило, в заключениях эксперта много догадок (впрочем, высказанных с большой уверенностью), которые обычно верны, но могут быть и исключения. Как данные, относящиеся к конкретной задаче, так и импликации, содержащиеся в правилах, могут быть не вполне определенными. Неопределенность можно промоделировать, приписывая утверждениям некоторые характеристики, отличные от «истина» и «ложь». Характеристики могут иметь свое внешнее выражение в форме дескрипторов, таких, как, например, *верно*, *весьма вероятно*, *вероятно*, *маловероятно*, *невозможно*. Другой способ: степень уверенности может выражаться в форме действительного числа, заключенного в некотором

интервале, например между 0 и 1. Такую числовую характеристику называют по-разному – «коэффициент определенности», «степень доверия» или «субъективная уверенность». Более естественным было бы использовать математические вероятности, но попытки применить их на практике приводят к трудностям. Происходит это по следующим причинам:

- Экспертам, по-видимому, неудобно мыслить в терминах вероятностей. Их оценки правдоподобия не вполне соответствуют математическому смыслу вероятностей.
- Работа с вероятностями, корректная с точки зрения математики, потребовала бы или какой-нибудь недоступной информации, или каких-либо упрощающих допущений, не вполне оправданных с точки зрения практического приложения.

Поэтому даже если выбранная мера правдоподобия лежит в интервале 0 и 1, более правильным будет называть ее из осторожности «субъективной уверенностью», подчеркивая этим, что имеется в виду оценка, данная экспертом. Оценки экспертов не удовлетворяют всем требованиям теории вероятностей. Кроме того, вычисления над такими оценками могут отличаться от исчисления вероятностей. Но, несмотря на это, они могут служить вполне адекватной моделью того, как человек оценивает достоверность своих выводов.

Остановимся подробнее на вопросе, почему вероятностный вывод не пригоден для реальных задач. Пусть мы имеем импликацию $A \supset B$. Когда можно сделать вывод, что B истинно? Неопределенность может возникнуть в двух пунктах: 1) Насколько вероятно, что A – истинно? 2) Насколько вероятно, что будет B при наличии A ?

Вероятность A можно положить $p(A) = 0.9$; вторую неопределенность можно задать при помощи условной вероятности B при наличии A – скажем, $p(B|A) = 0.95$. Какова вероятность B ? Имеем по формуле

$$p(B) = p(B|A) * p(A) + p(B|\text{не } A) * p(\text{не } A).$$

Для того чтобы определить вероятность B , необходимо знать вероятность $p(B|\text{не } A)$, которая во многих случаях неизвестна. Другой пример, рассмотрим

подсчет вероятности при конъюнкции $A \& B \supset C$. Пусть даны $p(A)$, $p(B)$ и $p(C|A \& B)$, посчитаем вероятность C . Имеем формулу

$$p(C) = p(C|A \& B) * p(A \& B) + p(C|\text{не}(A \& B)) * p(\text{не}(A \& B)).$$

Здесь уже два члена с неизвестными вероятностями $p(A \& B)$ и $p(C|\text{не}(A \& B))$.

Вообще говоря, если вы хотите разработать серьезную экспертную систему для некоторой выбранной вами предметной области, вы должны провести консультации с экспертами в этой области и многое узнать о ней сами. Достигнуть определенного понимания предметной области после общения с экспертами и чтения литературы, а затем облечь это понимание в форму представления знаний в рамках выбранного формального языка – это искусство, называемое инженерией знаний. Как правило, это сложная задача, требующая больших усилий, чего мы не можем себе позволить в рамках данного курса. Но язык Пролог позволяет быстро создать «игрушечную» экспертную систему – этому посвящена отдельная лабораторная работа.

4.3. Требования к современным экспертным системам

Если экспертные системы первого поколения рассматривались как машины для логического вывода, то экспертные системы второго поколения рассматриваются как усилители интеллектуальных возможностей человека.

Перечислим требования к современным экспертным системам.

Представление знаний

- Используются не поверхностные знания, а глубинные, представляющие собой теории предметных областей (аналогичные естественно-научным теориям) и общие стратегии решения проблем. Когда появляется новая проблема, экспертная система определяет конкретные знания, которые необходимо привлечь для решения.
- Система имеет не только модель предметной области, но и модель самой себя (определяет границы своей компетентности – «знаю, что я это знаю»).
- Система включает средства для одновременной работы с несколькими моделями предметной области.

- Имеется база знаний с неполной информацией.

Механизм вывода

- Замена дедукции на аргументацию. При обосновании решения основной операцией становится поиск аргументов, подтверждающих утверждение (привлечением фактов, из которых следует истинность данного утверждения или которые увеличивают уверенность в его истинности).
- Сочетание достоверного (дедуктивного) и правдоподобного (индуктивного, по аналогии) вывода.
- Способность системы по мере необходимости ослаблять или усиливать принятые в задаче допущения.
- Присутствие немонотонных рассуждений: поступившие факты могут изменить истинность выведенных ранее заключений.
- Экспертная система должна не только объяснить свое решение, но и обосновать: доказать или проверить то, что полученное системой решение не противоречит знаниям, которые хранятся в памяти системы²⁴.
- Одна из функций системы – оправдание. С помощью оправдания некоторое решение системы обосновывается не путем логических рассуждений или обращения к имеющимся в системе знаниям, а путем обращения к имеющейся в системе ценностной структуре. Оправдание убеждает в том, что данное решение не противоречит этой ценностной структуре.

Приобретение знаний и обучение

- Имеются средства управления процессом наполнения экспертной системы знаниями и настройки на предметную область, позволяющие выбирать модель представления знаний, в наибольшей степени соответствующую структуре знаний эксперта.
- Автоматическое обнаружение закономерностей в знаниях. Экспертная система получает знания от эксперта и самостоятельно извлекает их из базы

²⁴ Таким образом, обоснование является относительным. При изменении содержимого базы знаний обоснование может либо сохранить свою силу, либо стать неверным.

знаний путем выдвижения гипотез и построения их обоснования. То есть происходит самообучение на примерах.

- Происходит анализ имеющихся знаний, при обнаружении противоречий между старыми знаниями и вновь полученными от эксперта или выведенными эмпирически экспертная система обращается к пользователю, требуя разрешения конфликта.

5. Искусственный интеллект и автоматизированное проектирование

Система автоматизированного проектирования (САПР)

Создавая новые технические изделия, инженер-проектировщик решает задачи различного уровня сложности.

Если он работает над однотипными техническими устройствами и системами, для которых разработаны специальные методы проектирования, то ему нужно рассчитать нужные параметры изделия в соответствии с теми требованиями, которые содержатся в техническом задании, и получить на выходе нужную документацию на изготовление желаемого изделия. Примером может служить хорошо отработанная система процедур, используемых при создании электродвигателей, рассчитанных на различную мощность и различные режимы работы.

Задачи усложняются, если проектируемые изделия таковы, что создание каждой новой серии, например марки автомобиля или самолета, требует не механического использования ранее накопленного опыта, а его творческого применения. В процессе такой работы, конечно, что-то используется из прошлого опыта, срабатывают многие общие приемы и методы, но часть работы приходится выполнять на «пустом месте», создавать новые конструкции моделей, приемы и методы расчета.

Самый сложный уровень проектирования – тот, который обычно называется изобретательским или уровнем поискового проектирования (конструирования). Инженер создает принципиально новые изделия, аналогов которых ранее не существовало.

Конечно, доля нестандартных действий проектировщика меняется от жесткого расчета по готовой схеме до чистого изобретательства достаточно плавно, и разделение на уровни в значительной мере условно. Но тем не менее они позволяют выделять среди систем автоматизированного проектирования два

класса систем. Системы первого класса (классические САПР) в основном ориентированы на использование готовых решений. Системы второго класса (интеллектуальные САПР) поддерживают специальными средствами ту часть труда проектировщика, которая связана с поиском нестандартных решений.

Рассмотрим структуру классической САПР. Инженер через устройство ввода вводит в систему задание на проектирование. Оно поступает в управляющую программу, организующую все необходимые процедуры внутри системы. Эту программу обычно называют «монитор». Первая задача монитора – поиск в базе данных, в которой хранится информация о ранее найденных проектных решениях, информация о типовых блоках, узлах и деталях, нормативная документация, информация о свойствах и областях применимости различных материалов и т.п. С помощью информационно-поисковой системы в базе данных происходит поиск решений – аналогов. Если таковые находятся, то они предъявляются проектировщику. Возможно, что это именно то, что нужно.

Если готового решения нет, то монитор, используя стандартные процедуры проектирования, хранящиеся обычно в виде программных библиотек, проводит нужные расчеты и выдает полученное решение проектировщику. Тот оценивает его и при положительном решении выдает команду монитору для подготовки и выдачи документации на проектируемое изделие через графопостроитель, принтер или другие специальные устройства вывода.

Эта структура САПР – простейшая. По мере усложнения задач, решаемых в процессе проектирования, она также усложняется. Например, может оказаться необходимым не просто рассчитать параметры и характеристики нового изделия, но и промоделировать работу изделия в той среде, где оно будет работать, провести натурный эксперимент или имитационное моделирование, когда физическая модель проектируемого изделия заменяется его математической моделью. В таких случаях в структуру САПР включается также блок имитационного моделирования.

Часто в непосредственном контакте с САПР работает система подготовки рабочей документации изделия, включая, например, системы подготовки про-

грамм для станков с числовым программным управлением. Таким образом, классические САПР представляют собой совокупность аппаратных и программных средств, обеспечивающих автоматизацию всех основных этапов проектирования изделий, аналоги которых уже освоены промышленностью.

Интеллектуальные системы автоматизированного проектирования

Как и всякая другая система искусственного интеллекта, интеллектуальная САПР (ИСАПР) имеет в своем составе базу знаний, в которой хранится вся необходимая для ее работы информация о предметной области, в которой решается задача проектирования. В этой базе знаний собран и тот опыт, который накоплен проектировщиками, и экспертная информация о возможных путях поискового конструирования, опирающаяся на методы моделирования рассуждений, типичных для специалистов, работающих в данной области.

Общая схема ИСАПР может быть такой, как она показана на рис. 18. Когда на вход системы поступает задание на проектирование, которое в ИСАПР может формулироваться на ограниченном профессиональном естественном языке, оно с помощью естественного языкового интерфейса и других диалоговых средств понимается системой, уточняется у пользователя и переводится в специальное внутреннее представление. После этого делается попытка свести процесс проектирования к стандартным процедурам, реализуемым в классических САПР. Если эта попытка оказывается безуспешной, то логический блок передает задачу на вход экспертной системы, ориентированной на решение задачи проектирования в данной предметной области. Взаимодействуя с базой знаний и САПР, экспертная система ищет решение задачи.

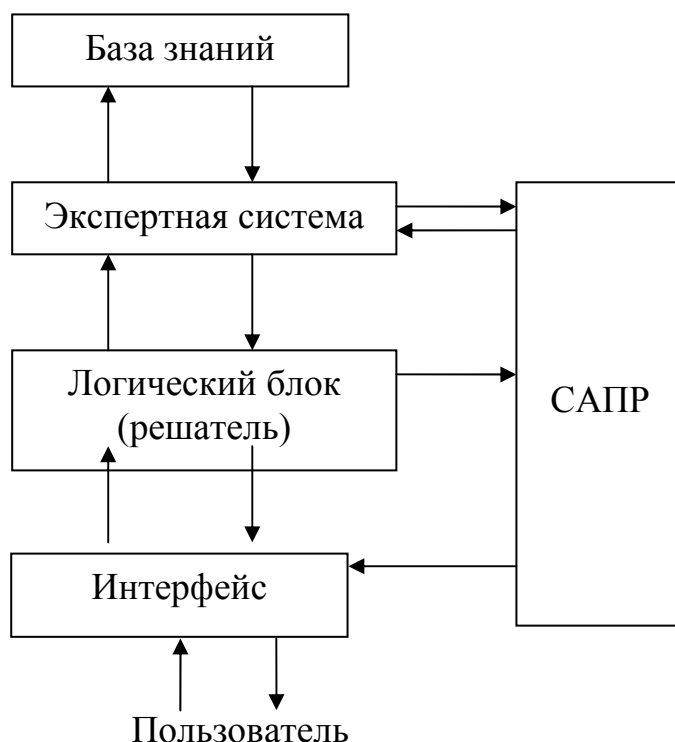


Рис. 18. Структура экспертной системы

В рамках современных ИСАПР интегрируются различные процедуры, задачи, этапы и уровни проектирования, обеспечивается непрерывный сквозной цикл автоматизированного проектирования, начиная с этапа подготовки технического задания и выработки технического предложения и кончая созданием рабочего и технического проектов. Автоматизируются не только рутинные, но и эвристические и творческие задачи, в частности, процедуры поискового конструирования, структурного синтеза и оптимизации. ИСАПР помогает поддерживать и интенсифицировать творческую активность разработчиков, повышать качество и производительность труда проектировщиков различных категорий, помогая сохранять и тиражировать уникальный проектный (экспертный) опыт и строить интеллектуальный интерфейс между проектировщиком и системой. Человек начинает доверять результатам компьютерной обработки информации. В результате повышается качество проектируемых образцов, так как увеличиваются число просматриваемых вариантов и глубина проработки каждого из них. Сокращаются и сроки проектирования, так как шире используются средства моделирования, ускоряются проектные расчеты и графические работы.

В процессе работы ИСАПР решаются все десять основных задач технического проектирования.

1. Составляется обоснованное техническое задание – это внешнее проектирование.
2. Анализируется техническое задание – это внутреннее проектирование.
3. Проводится концептуальный анализ: выбирается конструктивно-компоновочная схема, анализируется стоимость проекта.
4. Проводятся структурный синтез и оптимизация.
5. Ведется поисковое конструирование (изобретательство).
6. Проект планируется.
7. Конструкции перепроектируются и дорабатываются.
8. Повышается эффективность и качество инженерного анализа благодаря планированию вычислений и обучению пользователя владению прикладными программами. Проводится имитационное моделирование, выбираются численные методы расчета, результаты контролируются.
9. Проверяется соответствие отраслевым стандартам.
10. Готовятся рабочие чертежи и документация.

Каждая из этих задач требует весьма сложного программного и информационного обеспечения. Поэтому ИСАПР – это дорогостоящие и весьма сложные системы, но без них вряд ли возможно создавать такие сложнейшие технические изделия, как современные самолеты или подводные лодки, атомные электростанции или космические корабли.

Рекомендуемая литература

Рекомендуем аннотированный список книг, содержание которых так или иначе, касается тем изложенных в данном учебном пособии.

1. *Братко И.* Алгоритмы искусственного интеллекта на языке PROLOG. – 3-е изд.: Пер. с англ. – М.: Издательский дом «Вильямс», 2004. – 640 с.
2. *Стерлинг Л., Шапиро Э.* Искусство программирования на языке Пролог. – М.: Мир, 1990. – 235 с.

Хорошие учебники по логическому программированию. Содержат также много алгоритмов, используемых в реализациях систем искусственного интеллекта.

3. *Годфруа Ж.* Что такое психология. Т. 1 – М.: Мир, 1992.

Классический двухтомный учебник по психологии. В первом томе содержится материал, посвященный высшим функциям человека и обработке информации.

4. *Деннет Д. С.* Виды психики: на пути к пониманию сознания. – М.: Идея-Пресс, 2004. – 184 с.

Основная идея книги – сознание у человека появилось постепенно в процессе эволюции. Те животные, которые эволюционно близки к человеку, обладают сознанием в соответствующей степени.

5. *Джексон П.* Введение в экспертные системы: Пер. с англ.: Учебное пособие. – М.: Издательский дом «Вильямс», 2001. – 624 с.
6. *Лорьер Ж.Л.* Системы искусственного интеллекта. – М.: Мир, 1991. – 568 с.
7. *Люгер Д. Ф.* Искусственный интеллект: стратегии и методы решения сложных проблем. – М.: Издательский дом «Вильямс», 2003. – 864 с.
8. *Рассел С., Норвиг П.* Искусственный интеллект: современный подход. 2-е изд. – М.: Издательский дом «Вильямс», 2006. – 1408 с.

Современные учебники по искусственному интеллекту. Перечислены в порядке возрастания ценности для читателя.

9. *Лем С.* Библиотека XXI века. – М.: ООО «Издательство АСТ», 2002, 602 с.
10. *Лем С.* Диалоги. – М.: АСТ: Транзиткнига. 2005, 522 с.
11. *Лем С.* Молох. – М.: АСТ: Транзиткнига. 2005, 781 с.
12. *Лем С.* Сумма технологии – М.: ООО «Издательство АСТ»; СПб.: Terra Fantastica, 2002. – 668 с.

В этих книгах много материала, связанного с проблематикой искусственного интеллекта и излагаемого с точки зрения кибернетики и философии.

13. *Пенроуз Р.* Новый ум короля: о компьютерах, мышлении и законах физики. – М.: Едиториал УРСС, 2003. 384 с.
14. *Пенроуз Р.* Тени разума: в поисках науки о сознании. Ч. 1. Понимание разума и новая физика. – Москва; Ижевск: Институт компьютерных исследований, 2003. 368 с.

В этих книгах (особенно во второй) доказывается невозможность «сильного» искусственного интеллекта.

15. *Хофштадтер Д.* Гёдель, Эшер, Бах: эта бесконечная гирлянда. – Самара: Издательский Дом «Бахрах-М», 2001. – 752 с.

Центральная мысль этой замечательной книги (см. раздел 1.3 «Этапы в разработке ИИ», год 1979) заключается в том, что автореферентность является ключевой идеей, которая должна привести к созданию искусственного интеллекта.

16. *Хофштадтер Д., Деннет Д.* Глаз разума. – Самара: Издательский Дом «Бахрах-М», 2003. – 432 с.

Классическая антология эссе включает работы Хорхе Луиса Борхеса, Ричарда Доукинза, Джона Сирла, Роберта Нозика, Станислава Лема и многих других. Здесь представлены различные взгляды на природу человеческого мышления и природу искусственного интеллекта, здесь исследуются, сопоставляются, сталкиваются такие понятия, как «сознание», «душа», «личность»...

17. *Смаллиан Р.* Как же называется эта книга? – М.: Мир, 1981. – 238 с.

18. *Смаллиан Р.* Принцесса или тигр? – М.: Мир, 1985. – 221 с.

Научно-популярные книги Р. Смаллиана содержат много логических задач, в том числе связанных с автореферентностью.

19. *Барендрегт Х.* Лямбда-исчисление. Его синтаксис и семантика. – М.: Мир, 1985. – 606 с.

20. *Зюзьков В.М.* Теория алгоритмов: Учебное пособие. – Томск: Изд-во Том. ун-та, 2005. – 148 с.

Лямбда-исчисление можно рассматривать как теоретическую модель вычислимости и формализации понятия алгоритма. Программные реализации автореферентности (самоссылочности), автоаппликации (самоприменимости) и авторепликативности (самовозпроизведения) наиболее легко и просто описываются в рамках лямбда-исчисления. Книга Барендрегта содержит только математические аспекты теории. В книге В.М. Зюзькова описывается связь лямбда-исчисления с функциональным программированием.