Министерство образования и науки Российской Федерации

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

С. Н. Павлов

СИСТЕМЫ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

Часть 1

Учебное пособие

Томск
«Эль Контент»
2011

УДК 004.89(075.8)
ББК 32.813я73
П 12

Рецензенты:
Сергеев В. Л., докт. техн. наук, проф. кафедры геологии и разработки нефтяных месторождений Томского политехнического университета;
Кориков А. М., проф., зав. кафедрой автоматизированных систем управления ТУСУРа

Павлов С. Н.
П 12 Системы искусственного интеллекта : учеб. пособие. В 2-х частях. / С. Н. Павлов. — Томск: Эль Контент, 2011. — Ч. 1. — 176 с.

ISBN 978-5-4332-0013-5

В учебном пособии рассматриваются теоретические и организационно-методологические вопросы разработки и применения систем искусственного интеллекта. Изложены базовые принципы, подходы, классификация, методы, модели и стратегии систем различного назначения, традиционно считающиеся интеллектуальными: интеллектуальные информационные системы, понимание естественного языка и изображений, представление знаний и обучение, логический вывод и планирование действий. Большое внимание уделяется экспертным системам, обработки естественного языка, машинному зрению.

Пособие предназначено для студентов вузов, обучающихся по специальностям «Компьютерная инженерия», «Программное обеспечение вычислительной техники и автоматизированных систем», а также аспирантов и специалистов, интересующихся вопросами систем искусственного интеллекта.

УДК 004.89(075.8)
ББК 32.813я73

ISBN 978-5-4332-0013-5

Оглавление

ВВЕДЕНИЕ

Об искусственном интеллекте, системах искусственного интеллекта, интеллектуальных системах, интеллектуализированных системах пишут и говорят часто. Много из того, что вчера называли общими и специальными терминами, сегодня называют интеллектуальным. Практически любой созданный, а точнее выпущенный на рынок информационный или технический объект объявляется интеллектуальной системой, правда, в основном в СМИ, рекламных материалах. В действительности это отчасти мода, а отчасти широкое научное и практическое осознание интеллектуальности как одной из важных характеристик окружающего нас мира.

Примерно в 70-е годы прошлого столетия — начале фазы компьютерной революции был совершен концептуальный прорыв в новой области информатики и вычислительной техники, названной искусственным интеллектом. В эти годы была принята новая концепция, которая утверждала, что эффективность программы при решении задачи зависит от знаний, которыми она обладает, а не только от формализмов и методов вывода, которые она использует.

Наиболее значительными работами в области искусственного интеллекта являются разработки мощных компьютерных систем или экспертных систем, т. е. систем основанных на знаниях. Такие программы решения задач с представлением и применением фактических и эвристических знаний, совместной работой экспертов и инженеров по знаниям, разработчиков систем и логическим выводом позволяют переходить к новым информационным технологиям, к новой технологии программирования.

В настоящее время идет бурное развитие интеллектуальных систем, интеллектуальных концепций и технологий. Дисциплины, связанные с системами искусственного интеллекта, появились в связи с тенденциями образовательного процесса в сферах практической деятельности, связанных с решением задач интерпретации, диагностики, мониторинга, прогнозирования, планирования, проектирования, обучения, управления для плохо формализуемых проблем и зашумленных данных (знаний) при ограниченных ресурсах. Современный подход к решению таких проблем базируется на методах искусственного интеллекта.

Дисциплина «Системы искусственного интеллекта» играет фундаментальную роль в подготовке специалистов в области информатики и вычислительной техники. Различные учебные курсы, относящиеся к проблематике искусственного интеллекта (ИИ), включены в учебные планы многих институтов, университетов по рекомендациям международных ассоциаций ACM, AIS, AITP в 1997 г.

Настоящее учебное пособие «Системы искусственного интеллекта» посвящено вопросам организации, проектирования, разработки и применения систем, предназначенных для обработки информации, базирующихся на применении базовых принципов, подходах, методах и стратегиях разработки систем обычного назначения.

В настоящее время имеется достаточно обширная литература по системам искусственного интеллекта и программам в «демонстрационном варианте», т. е. при относительно небольшой базе знаний. Но для настоящей предметной области требуется весь арсенал средств и методов, накопленных за последние 40 лет. Разработка этого учебного пособия и есть попытка представления методов ИИ для серьезных практических областей: информатики, экономики, машинного перевода, распознавания образов и т. д.

Учебное пособие ставит цель: познакомить читателя с принципами создания и функционирования систем искусственного интеллекта; приобрести студентами и аспирантами знания, умения и навыки по реализации на компьютере таких систем на основе логических рассуждений, механизмов вывода, зашумленных данных.

Для изучения учебного пособия «Системы искусственного интеллекта» предполагаются знания по программированию, ЭВМ, высшей математике, дискретной математике, базам данных, теории информации и математической статистике.

Автор надеется, что предложенное изложение подходов, методов и моделей ИИ привлечет множество различных специалистов по многим приложениям, имеющих отношение к знаниям, логическому выводу, технологиям, что вызвано реальной потребностью практики.

Структурно учебное пособие состоит из десяти разделов. В первом разделе рассмотрена структура исследования в области искусственного интеллекта: понятие искусственного интеллекта, этапы развития ИИ; классификация нейробионического и информационного направлений; примеры различных классификаций СИИ.

Во втором разделе представлены задачи и общие методы решения хорошо определенных задач.

Дедуктивные рассуждения (принципы, методы, стратегии доказательства теорем), абдуктивные и индуктивные рассуждения и методы вывода при поиске решения задач представлены в разделе три.

Четвертый раздел посвящен неопределенностям знаний и способам их обработки: виды неопределенности описания задач; особенности данных и знаний и подходов представления неопределенностей (теорий вероятности, свидетельств, возможности); нечеткие знания (множества, отношения, лингвистические переменные, элементы теории приближенных рассуждений на примерах modus ponens и modus tollens).

В пятом разделе представлены продукционные системы логического вывода и управления выводом: прямая и обратная цепочки рассуждений, стратегии разрешения конфликта, а также эффективность поиска решения задачи.

В шестом разделе рассматривается планирование в интеллектуальных системах. Изложены классификация планирования, методы планирования в пространстве состояний, редукции задачи, ключевых состояний и ключевых операторов, анализа средств и целей, а так же примеры планирования систем.

Раздел семь излагает самое развивающееся направление систем искусственного интеллекта — экспертные системы (ЭС). Предложена классификация ЭС и интегрированных ЭС и примеры. Рассмотрена типовая структура, компоненты, этапы разработки, представления знаний и блок объяснений в ЭС.

Раздел восемь содержит знания и представления знаний в интеллектуальных системах, где решаются задачи, связанные с формализацией и представлением знаний.

В девятом разделе излагаются системы понимания естественного языка и машинного перевода. Представлены связь искусственного интеллекта и компьютерной лингвистики, понимание текста на естественном языке, понятие и процесс машинного перевода: традиционного и статистического.

Десятый раздел знакомит читателя с зрительным восприятием мира. Он содержит основные сведения о распознавании образов, методы распознавания, системы зрения роботов и машинное зрение.

В конце каждой главы учебного пособия приведены список литературы, а также контрольные вопросы и задания.

Учебное пособие в большей степени основано на материалах отечественных и в меньшей — зарубежных работ, являясь обобщением опыта, накопленного в этой области. Из отечественных работ следует отметить огромный вклад следующих специалистов: Д. А. Поспелова, Т. А. Гавриловой, В. Ф. Хорошевского, Ю. Ю. Тельнова, Э. В. Попова, Д. В. Гаскарова, А. В. Андрейчикова, И. П. Норенкова, В. Л. Стефанюка, Г. В. Рыбиной, В. Н. Вагина, В. Н. Аверина и др.

В заключение автор благодарит профессора А. М. Корикова за помощь в издании, критику и полезные замечания.

Автор осознает, что объем и качество учебного пособия не могут соответствовать общему стандарту представления о системах искусственного интеллекта и будет благодарен читателям за замеченные недостатки.

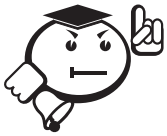
Соглашения, принятые в книге

Для улучшения восприятия материала в данной книге используются пиктограммы и специальное выделение важной информации.

Эта пиктограмма означает определение или новое понятие.

Эта пиктограмма означает внимание. Здесь выделена важная информация, требующая акцента на ней. Автор здесь может поделиться с читателем опытом, чтобы помочь избежать некоторых ошибок.

Эта пиктограмма означает теорему. Данный блок состоит из *Названия теоремы* (Слова Теорема и Номера теоремы) и Текста теоремы.



В блоке «На заметку» автор может указать дополнительные сведения или другой взгляд на изучаемый предмет, чтобы помочь читателю лучше понять основные идеи.

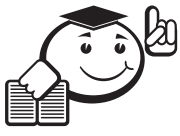
Пример

Эта пиктограмма означает пример. В данном блоке автор может привести практический пример для пояснения и разбора основных моментов, отраженных в теоретическом материале.

Выводы

Эта пиктограмма означает выводы. Здесь автор подводит итоги, обобщает изложенный материал или проводит анализ.

Контрольные вопросы и задания к главе



Литература к главе

Глава 1

СТРУКТУРА ИССЛЕДОВАНИЯ В ОБЛАСТИ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

Исследование и создание машин, обнаруживающих поведение, которое у людей называют интеллектуальным, назовем искусственным интеллектом. Самыми распространенными и современными машинами являются компьютерная техника и средства коммуникации, следовательно, направление искусственного интеллекта относится к области компьютеров и вычислительных систем.

Слово «интеллект» происходит от латинского слова *«intelligentie»*, которое, в свою очередь, образовалось от глагола *«intellgere»*, означающего способность понимать, определять смысл.

1.1 Понятие «искусственный интеллект»

Несмотря на все попытки дать точное определение понятию «искусственный интеллект» (ИИ), строгого определения до сих пор не существует, да и при появлении новых научных идей оно изменяется. Обозначим хотя бы границы этого понятия. И. Рич определяет ИИ как область исследования, направленную на создание компьютеров, которые выполняют такие функции, которые в настоящий момент человек выполняет лучше [16]. К таким функциям, которые проявляются у человека, относят восприятие, анализ, рассуждение, использование знаний, планирование действий, логический вывод и т. д. Очень близкое определение ИИ дает Дж. Ален: «ИИ — это наука о создании машин, решающих задачи, которые могут решать люди. . . » [15]. Здесь в фокусе ИИ оказываются те задачи, которые успешно решаются человеком и плохо — компьютерами. Эти два определения сопоставляют возможности человека и машин. Еще в 1950 году был предложен эмпирический тест А. Тьюринга для определения уровня интеллектуальности машин. В соответствии с тестом эксперт мог вступать в диалог либо с компьютером, либо с человеком. Тьюринг считал поведение компьютера интеллектуальным, если в диалоге

участвовал компьютер, а эксперт был не в состоянии определить, с кем он ведет диалог. В дальнейшем стали считать, что машинный интеллект отличается от человеческого интеллекта и, вероятно, попытка уподобления его естественному интеллекту ошибочна. Важность теста Тьюринга очевидна для оценивания качества современных программ ИИ, но он отвлекал научные силы от решения основной задачи ИИ — разработки общей теории машинного интеллекта, и использования этой теории для разработки интеллектуальных систем, решающих практические задачи.

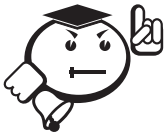
Хотя каждый из нас имеет достаточно определенное субъективное представление о том, что следует понимать под человеческим интеллектом, имеется множество определений искусственного интеллекта, предложенных в последнее время.

1) Способность успешно реагировать на любую, особенно новую ситуацию путем надлежащих корректировок поведения.
2) Способность понимать взаимосвязи между фактами действительности для выработки действий, ведущих к достижению поставленной цели.
3) Разработка новых моделей и методов решения задач, традиционно считавшихся интеллектуальными и не поддававшихся ранее формализации и автоматизации.
4) Разработка новых технологий программирования и переход на новые архитектуры ЭВМ.
5) Способность решать прикладные задачи, для которых ранее создаваемые системы были непригодны.
6) Область компьютерной науки, занимающейся автоматизацией разумного поведения.
7) Формализованные методы, позволяющие решать с помощью компьютера задачи управления не хуже, чем естественный интеллект.
8) Новые идеи, подходы решения задач на ЭВМ — новая технология программирования, переход на параллельные машины.
9) Раздел информатики, посвященный моделированию интеллектуальной деятельности человека.
10) Ветвь информатики, которая связана с автоматизацией интеллектуального поведения.
11) Наука о вычислениях, которые делают возможными восприятие, логический вывод и действие.
12) Информационная технология, связанная с процессами логического вывода, обучения и восприятия.
13) Одно из направлений информатики, целью которого является разработка компьютерных систем, способных выполнять функции, традиционно считавшиеся интеллектуальными, — понимание языка, логический вывод, использование накопленных знаний, обучение, планирование действий и т. д.
14) Наука, поставившая своей целью изучение и моделирование атрибута человека. Какова природа мышления? Какие процессы происходят в нашем организме, когда мы думаем, чувствуем, видим, понимаем? Возможно ли

в принципе понять, как работает наш мозг, и заставить мыслить неживую природу?

15) Наука, ставящая своей целью создание искусственных систем, достаточно хорошо имитирующих интеллектуальную деятельность человеческой личности, способностям которой мы не перестали удивляться.

Известный британский специалист А. Эндрю особое внимание уделил биологическим и биофизическим проблемам и моделям ИИ; Д. Хофштадтер указал на тесную связь ИИ с фундаментальной математикой, живописью и классической музыкой; недавно изданная в Нью-Йорке книга Т. Мунаката представляет нейронные сети и генетические алгоритмы, обычно рассматриваемые в основном направлении ИИ лишь как вспомогательные технические средства.



Итак, на сегодняшний день не существует определения, которое однозначно описывало бы ИИ. Среди многих точек зрения доминируют следующие три [5].

1) Фундаментальные исследования, в процессе которых разрабатываются новые модели и методы для решения задач, считающихся интеллектуальными и не поддававшихся ранее формализации и автоматизации.
2) Исследования, связанные с новыми идеями решения задач на ЭВМ, с разработкой новых технологий программирования и переходом к компьютерам не фон-неймановской архитектуры.
3) Исследования, в процессе которых появляется множество прикладных систем, способных решать задачи, для которых ранее создаваемые системы были не пригодны.

Под интеллектом обычно понимается возможность ставить и достигать цели при изменяющихся обстоятельствах, способность выбирать из множества целей те, которые скорее ведут к желаемому состоянию, адаптация к изменениям в среде и внутренним состояниям путем изменения их изменений.

ИИ решает сложные задачи: многофункциональные, интегрированные и интеллектуальные, где интеллектуальность напрямую связана с использованием не только синтаксической, но и семантической и прагматической информации.

Интеллектуальность системы (устройства) в «докреативный» период их развития большинство авторов обычно связывают с осуществлением процедур анализа окружающей обстановки, влияющей на эффективность принимаемых решений. В настоящее время интеллектуальность уже связывают с возможностью эффективного действия в условиях, «неожиданных» для системы. Такие системы все чаще стали называть креативными, то есть творческими.

А теперь дадим определение системы искусственного интеллекта (СИИ):

СИИ — это компьютерная, креативная система (многофункциональная, интегрированная, интеллектуальная) со сложной структурой, использующая накопление и корректировку знаний (синтаксической, семантической, прагматической информации) для постановки и достижения цели (целенаправленного поведения), адаптации к изменениям среды и внутреннего состояния путем изменения среды или внутреннего состояния.

Сегодня практически любой созданный, а точнее выпущенный на рынок, информационный или технический объект объявляется интеллектуальной системой, правда, в основном в СМИ и рекламных материалах. Иногда в ведущих научных организациях представляются интеллектуальные системы (объекты) с усложнением структуры обычных систем, то есть систем с большими функциональными возможностями. Например: интеллектуальный датчик или измерительный датчик принято считать интеллектуальным, если в месте расположения измерительного преобразователя имеется микропроцессорное устройство, осуществляющее фильтрацию и другие простые операции по предварительной обработке данных непосредственно там, где установлен датчик. То же самое наблюдается в нанотехнологии. Здесь термином интеллектуальный материал называется любой материал, который можно отнести к классу динамических — он может менять свои базовые свойства или структуру, основываясь на анализе внешней обстановки.

Такие интеллектуальные материалы применяются, например в технологии Steals, поскольку они меняют свой цвет, рассеивающиеи и поглощающие свойства и т. п. в ответ на внешние условия и команды летчика.

Д. В. Гаскаров вводит три определения для интеллектуальных систем.

Интеллектуальная система — это информационно-вычислительная система с интеллектуальной поддержкой при решении задач без участия оператора (ЛПР).

Интеллектуализированная система — это информационно-вычислительная система с интеллектуальной поддержкой при решении задач с участием ЛПР.

Система с интеллектуальной поддержкой — система, способная самостоятельно принимать решения.

Другими словами: способность системы получать и анализировать информацию, понимать ее и делать новые выводы, формулировать заключения, то есть

«мыслить», помогая естественному интеллекту — человеку, который, корректируя, улучшает принятое интегрированное решение.

Под системой искусственного интеллекта (СИИ) будем понимать аппаратно-программный комплекс, обладающий способностью [1, 6]:

- к накоплению и корректировке знаний на основе активного восприятия информации о мире и обобщенного опыта;
- к целенаправленному поведению на основе логического вывода и алгоритмов различной степени неопределенности представления и управления сложных систем.

Д. В. Гаскаров определяет интеллектуальную систему, как — информационно-вычислительную с интеллектуальной поддержкой при решении задач без участия лица, принимающего решения (ЛПР), а интеллектуализированную систему как информационно-вычислительную систему с интеллектуальной поддержкой при решении задач с участием оператора (ЛПР).

Итак, интеллектуальная поддержка — это способность системы принимать самостоятельное решение на основе имеющихся математического, алгоритмического, программного и интеллектуального обеспечений для сложных целенаправленных, динамических, слабо формализуемых, адаптированных систем с самообучением и самоорганизацией.

1.2 Этапы развития искусственного интеллекта

Еще в 1200-х годах появились попытки создания искусственного человека и его разума. Изобретатель Раймонд Луллий сконструировал машину, состоящую из кругов, размеченных буквами и раскрашенных в разные цвета, которые символизировали различные понятия, элементы стихии, субъекты и объекты знания. Разнообразное их сочетание, приводили с помощью логических операций к выводу «формул знаний».

В 40-х годах 20 в. с появлением электронно-вычислительных машин искусственный интеллект обрел второе рождение.

Исследования в области искусственного интеллекта имеют две цели:

- выяснение сущности естественного интеллекта (человеческого интеллекта);
- использование машинного интеллекта для преобразования новых знаний и для решения интеллектуальных задач.

Первую цель выполняют психологи. Они контролируют модель поведения человека при решении задач и затем ее корректируют.

Вторую цель выполняет исследователь. Он синтезирует интеллектуальное поведение системы независимо от методов, которыми пользуются люди.

В соответствии с этими целями искусственный интеллект разделился на два основных направления: нейрокибернетику и кибернетику «черного ящика».

Первое направление утверждает, что мыслить может только человеческий мозг и, соответственно, любая машина должна быть выполнена как человеческий мозг, то есть воспроизвести его структуру, принцип действия.

В конце 1950-х годов начали разрабатываться и создаваться первые нейросети и нейрокомпьютеры американскими учеными У. МакКалоком, У. Питтсом, Ф. Розенблаттом, представляющие и в настоящее время нейрокомпьютерное направление СИИ.

Второе направление говорит не о принципе действия мыслящего устройства, а об адекватном моделировании его функционирования, то есть поисков алгоритмов решения интеллектуальных задач, другими словами, собственных моделей мышления, а не человеческих.

Становление искусственного интеллекта. Работы в области искусственного интеллекта начались с зарождения нейрокибернетики. Так как мозг человека состоит из множества нервных клеток — нейронов, то исследователи пытались строить разумные машины, имитируя поведение коллектива нейронов. Основную идею нейрокибернетики можно сформулировать следующим образом. Единственный объект, способный мыслить, — это человеческий мозг. Поэтому любое мыслящее устройство должно быть обязательно выполнено по образу и подобию человеческого мозга, воспроизводить его структуру, его принцип действия. Таким образом, нейрокибернетика занимается аппаратным моделированием структуры мозга и его деятельности.

В 1943 году У. МакКолак и У. Питтс предложили модель формального логического нейрона который мог находится в двух устойчивых состояниях. Д. Хебб в 1949 году разработал простое правило, позволяющее изменять веса связей между нейронами с целью их обучения. В 1951 году М. Минского и Д. Эдмондс разработали нейрокомпьютер, который содержал 40 нейронов.

Термин «искусственный интеллект» был предложен на семинаре Дартсмутского колледжа (США) в 1956 году. Первые работы по ИИ проводились в Массачусетстком технологическом институте под руководством М. Минский и Дж. Маккарти, в университете Карнеги-Меллона под руководством Г. Саймона и А. Ньюэлла. Они и считаются «отцами» ИИ.

Эвристический поиск и доказательство теорем (1956–1969). Работы Г. Саймона и А. Ньюэлла по разработке программы «Логик-теоретик» были предназначены для доказательства теорем в исчислении высказываний. С помощью ее были доказаны ряд теорем, но поиск решений был не эффективен.

А. Ньюэл и Г. Саймон после анализа методов решения приступили к синтезу общих методов поиска решений, разработав следующую программу «Универсальный решатель задач» или GPS (General Problem Solver). Она разрабатывалась с целью имитации процесса решения задач человеком и базировалась на идеях эвристического поиска. GPS основывалась на модели лабиринтного поиска. Согласно этому подходу решение интеллектуальной задачи выполнялось путем перебора огромного количества вариантов, который представлялся в виде движения по лабиринту. Программа GPS могла настраиваться на предметную область. Для этого необходимо было задать структуру состояний задачи и операторы, преобразующие эти состояния. Решение задачи осуществлялось на основе поисковых алгоритмов в пространстве возможных решений по эвристическим правилам, которые направляли поиск к искомой цели. В настоящее время такая модель признается тупиковой и имеет ограниченное использование.

В этот период развития ИИ основные исследования и разработки были представлены Дж. Маккарти, Дж. Робинсоном, К. Грином, Д. Хеббом, Ф. Розенблаттом, М. Минским:

- разработка языка Лисп (Дж. Маккарти);
- идея представления знаний и логического вывода в системах искусственного интеллекта (Дж. Маккарти);
- разработка метода резолюции (Дж Робинсон);
- разработка вопрос-ответной системы на основе логического представления знаний (К. Грин);
- создание персептрона для распознавания образов (Ф. Розенблат);
- написание книги об ограниченных возможностях персептрона (М. Минский, С. Пейперт).

Таким образом, к концу 60-х годов основное внимание в области ИИ стало уделяться методам представления задач и поиску решений, в частности представлению задач в логической форме и автоматическому доказательству теорем на основе метода резолюций.

Представление знаний (1969–1979). 70-е годы прошлого столетия характеризуются следующими исследованиями.

1) Отказ от поиска универсального алгоритма мышления, а моделирование конкретных знаний экспертов. К концу 60-х годов было обнаружено, что для решения практически важных задач недостаточно одних знаний общего характера (общих стратегий поиска решений). Успешное решение прикладных задач возможно только при наличии хорошо структурированных специальных знаний.
2) Решаются задачи понимания естественного языка и обработки изображений.
3) Предложено эволюционное программирование (моделирование) Дж. Голландом, то есть процесс моделирования человека заменялся моделированием процесса его эволюции. Впервые предложено решение сложных задач моделированием эволюции с помощью компьютерных алгоритмов (генетических алгоритмов).

Программа DENDRAL, разработанная в 1969 году Э. Фейгенбаумом, Б. Букхененом, Э. Лидербергом, содержала детальные сведения об области органиченной химии и помогала специалистам определять молекулярную структуру органических соединений по данным, полученным с помощью масс-спектрометра. Масс-спектометр, разделяя молекулы на фрагменты, измеряет массу и электрический заряд каждого из фрагментов. Чтобы определять множество форм молекул, которые могут состоять из таких фрагментов, в программе использовались эмпирические знания химиков, представленные в форме правил «если-то». Это позволило резко сократить число предлагаемых вариантов решений.

DENDRAL была первой успешно реализованной программой, аккумулирующей знания экспертов. Такие программы получили название «экспертные системы». Они содержат большой объем практических знаний, что позволяет получать ответы (решения).

Далее Э. Фейгенбаум, Б. Букхенен, Э. Шортлифф разрабатывают экспертную систему MYCIN. Она содержит около 450 правил, позволяющих диагностировать инфекционные заболевания крови. MYCIN уже позволяет обрабатывать и получать правдоподобные заключения на основе неопределенных (ненадежных) знаний. С этой целью факты и сами правила в системе характеризовались числовой функцией принадлежности — коэффициентом уверенности (степени достоверности).

После удаления из системы MYCIN базы знаний была представлена оболочка EMYCIN (оставлена только логика управления правилами), которую можно было наполнять знаниями.

С этого момента искусственный интеллект перестал быть наукой и стал приносить практическую пользу. Огромный успех имела экспертная система PROSPECTOR (1979 г.), используемая в геологоразведке месторождений. С появлением экспертных систем бизнес в сфере интеллектуальных информационных технологий впервые становится рентабельным. В системе PROSPECTOR база знаний представлялась в виде семантической сети и система обеспечивала взаимодействие с пользователем на естественном языке.

Семантические сети были предложены в 1967 г. М. Куиллианом. Существенный шаг при решении задачи понимания естественного языка, сделал Виноград, который разработал программу SHRDLU (1968 г.) (перемещение с помощью естественно-языковых команд кубиков и пирамид). Дальнейшее совершенствование систем понимания естественного языка связано с именами Р. Шенка и В. Вудса. Шенк разработал программу, преобразующую входные естественно-языковые высказывания к элементарным концептам, которые можно было представить в памяти ЭВМ. Далее он разработал модель представления знаний: скрипты или сценарии. Наконец, в 1973 году В. Вудс создал систему LUNAR, которая позволяла геологам задавать вопросы на естественном языке относительно образцов пород, доставленных с Луны.

Расширение приложений систем ИИ требовало развития моделей представления знаний. В 1973 году А. Колмероэ создает язык логического программирования Пролог, ставший популярным в США, Европе, России. В США создается язык PLANNER, поддерживающий предикатный уровень представления знаний. Теорию фреймов в 1975 году предложил М. Минский. Затем были разработаны языки для работы с фрэймовыми моделями: FRL, KRL, GUS и т. д.

Коммерческий успех компьютерной индустрии 1979–1986. Первой интеллектуальной системой, нашедшей применение в промышленности, стала экспертная система К1, разработанная Мак-Дермотом в 1982 году. Система К1 применялась для конфигурации компьютерных систем семейства VAX. Коммерческая версия системы, разработанная корпорацией Digital Equipment совместно с университетом Карнегги-Меллона (США), получила название XCON. К 1986 году эта система позволяла корпорации экономить 70 млн долларов ежегодно. Кроме этого, применение системы сократило число ошибок с 30 % до 1 %.

В 1981 году Япония объявляет о начале проекта машин 5-го поколения, базирующихся на принципах ИИ. Этот проект способствовал активизации исследований в области ИИ во многих странах.

Начиная с 1985 года, экспертные системы, а затем и системы, воспринимающие естественный язык (ЕЯ-системы), а затем и нейронные сети (НС) стали активно использоваться в коммерческих приложениях.

Коммерческие успехи к фирмам-разработчикам систем искусственного интеллекта пришли не сразу. На протяжении 1960–1985 гг. успехи ИИ касались в основном исследовательских разработок, которые демонстрировали пригодность СИИ для практического использования.

Этап разработки и становления интеллектуальных систем 1-го поколения (1986–1996). В рамках исследований по ИИ сформировалось самостоятельное направление — экспертная система, или инженерия знаний. В задачу этого направления входят исследования и разработка программ (устройств), использующих знания и процедуры вывода для решения задач, кажущихся трудными для людей-экспертов.

Огромный интерес к ЭС со стороны пользователей вызван по крайней мере тремя причинами:

- ЭС ориентированы на задачи, для которых отсутствуют или неизвестны алгоритмы их решения;
- ЭС позволяют пользователям, не знающим программирования, самостоятельно разрабатывать задачи, используя свой опыт и знания;
- ЭС позволяет получать результаты, не уступающие, а иногда и превосходящие возможности людей-экспертов.

Этот период охватывает технологии разработки традиционных (простых) ЭС и начала разработки интегрированных ЭС. И те и другие в настоящее время объединяются в ЭС первого поколения.

Работы Дж. Хопфилда, Д. Румельхарте и Г. Хинтона по моделям нейронных сетей послужили толчком к лавинообразному росту применений таких моделей для решения практических задач. Помимо теоретических разработок, этому способствовало и появление мощных аппаратно-программных средств, позволяющих моделировать нейронные сети достаточного уровня сложности.

В этот период источником знаний являются эксперт с эмпирическими знаниями, а базы знаний составляли отдельные формы (модели): продукционные (в основном), фреймовые, семантические сети, решающие деревья. Логический вывод представлялся детерминированным и дедуктивным, а язык общения с пользователем — фразами и терминами жесткой конструкции прикладной области.

Этап разработки интеллектуальных систем II поколения (1996–2000). Этот этап характеризуется интегрированными и гибридными принципами построения систем, где основой являются БЗ с любыми функциями знаний (библиотек) и автоматическим извлечением их; дедуктивным, абдуктивным, индуктивным, нечетким логическим выводом; проблемно-ориентированным языком общения, и обработкой не только статической, но и динамической информации (см. часть 2 главу 7).

Предыдущие этапы развития ИИ характеризуются разрозненными подходами, основанными на выделении частного свойства понятия «интеллект».

Пример

Например:

1) принцип знаниецентризма обусловил развитие и господство в течение определенного периода когностивистских моделей ИИ, в частности логических. В соответствии с такими моделями на первый план выдвигается способность интеллектуальной системы рассуждать, а действия (поведение) рассматриваются как нечто вторичное. Основное внимание при этом уделяется логическому выводу;
2) выделение в определении ИИ функции обучения (адаптации) способствовало развитию коннекционистских (нейронные сети) и эволюционных моделей (генетические алгоритмы);
3) трактовка ИИ с позиций способности к восприятию и коммуникации привела к развитию моделей понимания изображений и естественного языка.

Начиная с начала 90-х годов в ИИ стали преобладать две основные тенденции: интеграция и децентрализация [13,16].

Интеграционные процессы проявились в разработке интегрированных и гибридных систем искусственного интеллекта, объединяющих в себе преимущества разнородных моделей, например нечеткие экспертные системы и нейронные сети. В таких интегрированных системах могут поддерживаться различные модели представления знаний, разные типы рассуждений, модели восприятия и распознавания образов.

Процессы *децентрализации* связаны с рассмотрением ИИ с позиций коллективного поведения большого числа взаимодействующих между собой интеллектуальных агентов. При этом интеллект агента рассматривается как подсистема управления деятельностью в процессе взаимодействия с другими агентами.

В основе распределенного (децентрализованного) интеллекта лежит функционально-структурная единица — агент, способная [13]:

- воздействовать на других агентов и самих себя;
- образовывать свои собственные цели;
- общаться с другими агентами;
- функционировать без прямого вмешательства со стороны любых средств и осуществлять самоконтроль (автономность);
- воспринимать часть среды своего функционирования;
- строить локальное представление среды;
- выполнять обязанности и оказывать услуги;
- самовоспроизводиться.

Важно отметить, что при решении конкретной задачи агенты образуют структурное сообщество, в котором наблюдается определенная кооперация между агентами.

Например, можно говорить об агентах, выполняющих простую передачу сообщений, о координирующх агентах, которые организуют взаимодействие в сообществе (группе) агентов, о поисковых агентах и т. д.

Таким образом, согласно коллективистской модели, основным объектом исследований ИИ является сообщество неоднородных, взаимодействующих агентов, а основное содержание разработок связано с созданием интеллектуальных агентов, обладающих заданными свойствами, и вычислительных структур, поддерживающих взаимодействие агентов.

Начало XXI века характеризуется началом исследований и разработкой систем с интеллектуальным интерфейсом: самообучающиеся, адаптивные, а затем и гибридные системы искусственного интеллекта, объединяющие в себе возможности, представленные нейронными сетями и моделями представления знаний. Объединение нейронных сетей и баз знаний может выполняться различными способами. В простейших случаях нейронная сеть выполняет предварительную или завершающую обработку информации в системах, основанных на знаниях. В более интересных случаях осуществляется встраивание нейронных сетей в базы знаний, и наоборот.

В настоящее время ведутся исследования и разработки систем гибридного интеллекта на основе адаптивного информационного взаимодействия коллективов людей и ЭВМ, т. е. взаимодействие естественных интеллектов в природе, обществе, технике, которые включают в себя системы искусственного интеллекта.

Пример

Например, процесс совместной интеллектуальной деятельности студентов и ЭВМ как компонентов системы гибридного интеллекта есть процесс формирования абстрактного алгоритма решения управления данным объектом.

Такой процесс может осуществляться как переход коллектива операторов от индивидуальных по языку и субъективных по соотношению с объективной реальностью и между собой (интуитивных) отражений с использованием ассоциативных информационных моделей к психическим моделям, адекватным оперативной структуре задачи, и к возможности построения информационных моделей. Это и есть путь к гибридизации участвующих интеллектов или создания системы человек — машина — среда.

1.3 Классификация искусственного интеллекта

Среди специалистов ИИ нет единой точки зрения на область искусственного интеллекта и даже на цели исследования. В настоящее время различают две точки зрения (направления) к моделированию искусственного интеллекта (AI — *artifical intelligence*): искусственный разум, направленный на моделирование внутренней структуры системы, и машинный интеллект, заключающийся в строгом задании результата функционирования.

Разделение работ по искусственному интеллекту на два направления связано с существованием двух точек зрения на вопрос, каким образом строить системы искусственного интеллекта [4–10].

Сторонники нейробионики моделируют искусственным образом процессы, происходящие в мозгу человека. Этот путь ИИ исследует понимание механизмов восприятия, выявление способов работы мозга, создание технических средств для моделирования биологических структур и протекающих в них процессах. Эта точка зрения состоит в том, что именно изучение механизмов естественного мышления и анализ данных о способах формирования разумного поведения человека могут создать основу для построения систем искусственного интеллекта, причем построение это должно осуществляться, прежде всего, как моделирование, воспроизведение техниескими средствами принципов и конкретных особенностей функционирования биологических объектов.

Практически это разработка элементов, подобных нейронам, объединение их в системы — нейросети, нейрокомпьютеры. В настоящее время эти технологии являются очень перспективными и быстро развивающимися.

Сторонники второй точки зрения убеждены, что «важнее всего результат», т. е. хорошее совпадение поведения искусственно созданных и естественных интеллектуальных систем, а что касается внутренних механизмов формирования поведения, то разработчик искусственного интеллекта вовсе не должен копировать или даже учитывать особенности естественных, живых аналогов.

Вторая, но доминирующая точка зрения на искусственный интеллект называется информационной, где основной целью является не построение технического аналога биологической системы, а адекватное моделирование функционирования системы, т. е. создания средств решения задач, традиционно считающихся интеллектуальными.

Первое направление искусственного интеллекта рассматривает данные о нейрофизиологических и психологических механизмах интеллектуальной деятельности и, в более широком плане, разумного поведения человека. Оно стремится воспроизвести эти механизмы с помощью тех или иных технических устройств, с тем чтобы «поведение» таких устройств хорошо совпадало с поведением человека в определенных, заранее задаваемых пределах. Развитие этого направления тесно связано с успехами наук о человеке. Для него характерно стремление к воспроизведению более широкого, чем в машинном интеллекте, спектра проявлений разумной деятельности человека. Системы искусственного разума базируются на математической интерпретации деятельности нервной системы во главе с мозгом человека и реализуются в виде нейроподобных сетей на базе нейроподобного элемента — аналога нейрона.

Нейроподобные сети в последнее время являются одним из самых перспективных направлений в области искусственного интеллекта и постепенно входят в бытность людей в широком спектре деятельности.

Сети первой группы, такие, как сети обратного распространения ошибки, сети Хопфилда и др., используются для распознавания образов, анализа и синтеза речи, перевода с одного языка на другой и прогнозирования. Это вызвано такими особенностями сетей, как восстановление изображения по его части, устойчивость к зашумлению входного сигнала, прогнозирование изменения входов и параллельность

вычислений. Также немаловажной характеристикой является способность функционировать даже при потере некоторой части сети.

Сети второй группы используются как системы управления в реальном времени несложных объектов. Это управление популярными в последнее время интеллектуальными агентами, исполняющими роль виртуальных секретарей. Особенностями данной группы является появление некоторых внутренних стимулов, возможность к самообучению и функционированию в реальном времени.

И, наконец, *сети третьей группы*, являющиеся дальнейшим развитием предыдущих, представляют собой уже нейроподобные системы, и нацелены они на создание экзотических в настоящее время виртуальных личностей, информационных копий человека, средой обитания которых является глобальная сеть Интернет. Данное направление только зарождается, но есть немалый шанс, что мы станем свидетелями ситуации рождения виртуальных людей, подробно описанной фантастами и режиссерами.

Сейчас в Интернете повсеместно можно встретить признаки зарождения подобных проектов, призывы объединиться всем научным потенциалом способного думать человечества в целях очеловечивания Интернета, преобразования его в разумную систему или среду обитания разумных систем. Раз существуют подобные предпосылки, значит, ничто не остановит полет человеческой мысли на пути достижения поставленной цели.

Выводы

На основании вышеизложенного можно сделать вывод о том, что основные направления искусственного интеллекта связаны с моделированием, но в случае машинного интеллекта мы имеем дело с моделированием феноменологическим, имитационным, а в случае искусственного разума — с моделированием структурным.

Второе направление, таким образом, рассматривает продукт интеллектуальной деятельности человека, изучает его структуру, и стремится воспроизвести этот продукт средствами современной техники. Моделирование систем машинного интеллекта достигается за счет использования законов формальной логики, теории множеств, графов, семантических сетей и других достижений науки в области дискретных вычислений. Основные результаты заключаются в создании экспертных систем, систем разбора естественного языка и простейших систем управления вида «стимул-реакция». Ясно, что успехи этого направления искусственного интеллекта оказываются тесно связанны с развитием возможностей ЭВМ и искусства программирования, то есть с тем комплексом научно-технических исследований, которые часто называют компьютерными науками.

1.3.1 Нейробионическое направление

Нейронные модели не только повторяют функции мозга, но и способны выполнять функции, имеющие свою собственную ценность. Поэтому возникли и остаются в настоящее время две взаимно обогащающие друг друга цели нейронного моделирования: первая — понять функционирование нервной системы человека на уровне физиологии и психологии и вторая — создать вычислительные системы (искусственные нейронные сети), выполняющие функции, сходные с функциями мозга.

Именно эта последняя цель и находится в центре внимания для нас.

Еще в 1949 г. была создана модель человеческого обучения — модель Д. Хэбба. Он предложил закон обучения, явившийся стартовой точкой для алгоритмов обучения искусственных нейронных сетей.

В пятидесятые и шестидесятые годы группа исследователей создала первые искусственные нейронные сети. Выполненные первоначально как электронные сети, они были позднее перенесены в более гибкую среду компьютерного моделирования.

Идея нейрокомпьютера появилась практически одновременно с зарождением последовательных фоннеймановских ЭВМ. Ключевая работа МакКаллока и Питтса по нейро-вычислениями (McCulloch and Pitts,1943) появилась в 1943 году на два года раньше знаменитой докладной записки фон Неймана о принципах организации вычислений в последовательных универсальных ЭВМ. Они предложили схему компьютера, основанного на аналогии с работой человеческого мозга, и создали упрощенную модель нервной клетки — нейрон.

Но нейрокомпьютер предъявляет жесткие требования к вычислительной мощности аппарата, и только в семидесятые годы нейросетевые методы решения прикладных задач стали приобретать популярность.

Первый экспериментальный нейрокомпьютер Snark был построен Марвином Минским в 1951 году, но первый успех нейрокомпьютинга связывают с разработкой персептрона (от английского *perception* — восприятие) американцем Френком Розенблаттом. Это была одна из первых моделей нейронных сетей, которая вызвала большой интерес из-за своей способности обучаться распознаванию простых образов. Персептрон состоял из бинарных нейроподобных элементов и имел простую топологию, что позволило достаточно полно проанализировать ее работу и создать многочисленные физические реализации.

Работы Минского и Пейперта, доказавшие ограниченные возможности простейшего персентрона, погасили энтузиазм большинства исследователей на два десятилетия.

Широкий интерес научной общественности к нейросетям начался в начале восмидесятых годов после теоретических работ физика Хопфилда (Hopfield, 1982,1984). Он и его последователи обогатили теорию параллельных вычислений многими идеями: коллективное взаимодействие нейронов, энергия сети, температура обучения и т. д.

Практическое применение нейросетей началось после публикации Румельхартом метода обучения многослойного персептрона, названного ими методом обратного распространения ошибки. Таким образом, ограничения персептронов, доказательства которых представили Минский и Пейперт, оказались преодолимыми.

Удельная стоимость современных нейровычислений на порядок ниже, чем

у традиционных компьютеров, а быстродействие — в сотни раз выше. Системы нейронной обработки можно классифицировать следующим образом (рис. 1.1).

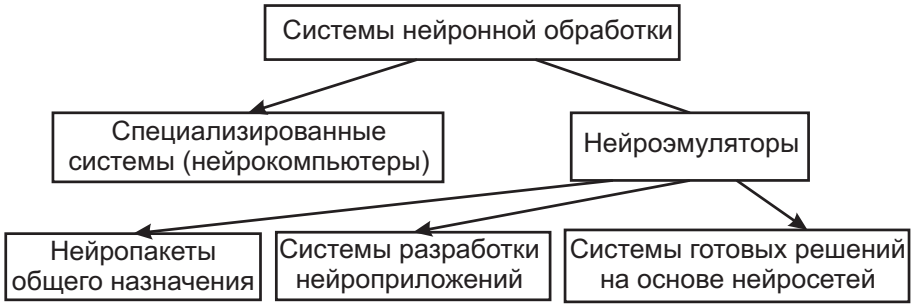


Рис. 1.1 – Классификация систем нейронной обработки.

Реально *нейрокомпьютеры* используются в специализированных системах, где требуется обучать и постоянно переобучать сотни нейросетей, объединенные в единые информационные комплексы, или в системах реального времени, где скорость обработки данных критична (например, при обработке экспериментов на современных ускорителях элементарных частиц используют нейрокомпьютер CNAPS с производительностью 10^{10} и каждый процессор состоит из 512 нейронов).

Доступность и возросшие вычислительные возможности современных компьютеров привели к широкому распространению программ, использующих принципы нейросетевой обработки данных, но исполняемых на последовательных компьютерах. Этот подход не использует параллелизм, но ориентируется исключительно на способность нейросетей решать неформализованные задачи и реализуется *нейроэмуляторами*.

Нейропакеты общего назначения — законченные независимые программные продукты, предназначенные для широкого класса задач (например, статистической обработке данных).

Множество их можно найти в Internet как shareware или freeware. Это обычные многослойные персептроны с одним или несколькими правилами обучения.

Коммерческие пакеты (BrainMaker Professional, NeuroForecaster, Jopa — IQ300) имеют собственный встроенный блок предобработки данных, хотя иногда для этой цели удобнее использовать стандартные электронные таблицы. Так, нейропродукты группы ФИАН встраиваются непосредственно в Microsoft Exel в качестве специализированной функции обработки данных.

Системы разработки нейроприложений — это программное обеспечение, способное генерировать программный код, использующий обученные нейросети для обработки данных.

Удобным инструментом разработки сложных нейросетей является MATLAB с прилагающимся к нему нейросетевым лассментариием, ограниченно вписавшимся

в матричную идеологию этой системы. MATLAB предоставляет удобную среду для синтеза нейросетевых методик с прочими методами обработки (*wavelet* — анализ, статистика, финансовый анализ и т. п.).

Системы готовых решений на основе нейросетей — это конечный результат.

Здесь нейросети спрятаны от пользователя в недрах готовых автоматизированных комплексов, предназначенных для решения конкретных производственных задач (управление предприятием, организационно-техническими системами, реакторами и т. д.). Например, продукт Falcon встраивается в банковскую автоматизированную систему обслуживания платежей по пластиковым карточкам.

Таким образом, нейропакеты не требуют самостоятельного программирования, легко осваиваются, это инструмент быстрого и дешевого решения прикладных задач; нейроприложения могут использоваться для создания сложных систем обработки данных в реальном времени; системы готовых решений не предполагают знакомства пользователя с нейросетями, представляют комплексное решение проблемы.

Также можно представить возможные классификации систем по типу входных и выходных сигналов на бинарные и аналоговые сети. Первые оперируют с двоичными сигналами, и выход каждого нейрона может принимать только два значения: логический ноль («заторможенное» состояние) и логическая единица («возбужденное» состояние). В аналоговых сетях выходные значения нейронов способны принимать непрерывные значения.

Еще одна классификация делит нейросети на синхронные и асинхронные. В первом случае в каждый момент времени свое состояние меняет лишь один нейрон, во втором — состояние меняется сразу у целой группы нейронов, как правило, у всего слоя.

Далее представим классификацию по типу связей и типу обучения (Encoding — Decoding), т. е. классификацию базовых нейроархитектур, впервые предложенную Бартом Коско, в табл. 1.1.

Таблица 1.1 – Классификация нейросетей.

Тип связей (Decoding)	Тип обучения (Coding)	
	С «учителем»	Без «учителя»
Без обратных связей	Многослойные персептроны (аппроксимация функций, классификация)	Соревновательные сети, карты Кохонена (сжатие данных, выделение признаков)
С обратными связями	Рекуррентные аппроксиматоры (предсказание временных рядов, обучение в режиме on-line)	Сеть Хопфилда (ассоциативная память, кластеризация данных, оптимизация)

Приложения нейронных сетей охватывают разнообразные области применений: распознавание образов, обработка зашумленных данных, дополнение образов, ассоциативный поиск, лассификация, оптимизация, прогноз, диагностика, обработка сигналов, абстрагирование, управление процессами, сегментация данных, сжатие информации, моделирование сложных процессов, машинное зрение, распознавание речи и т. д.

В настоящее время в каждой предметной области можно найти постановки нейросетевых задач.

Экономика и бизнес. Предсказание рынков, автоматический дилинг, оценка риска невозврата кредитов, предсказание банкротства, оценка стоимости недвижимости, автоматическое рейтингование, оптимизация портфелей, оптимизация товарных и денежных потоков, автоматическое считывание чеков и форм, безопасность транзакций по пластиковым карточкам. Пример: средства нейросетевого анализа больших потоков данных для крупной розничной торговли (http://www.retek.com).

Медицина. Обработка медицинских изображений, мониторинг состояния пациентов, диагностика, факторный анализ эффективности лечения, очистка показаний приборов от шума. Пример: система ранней диагностики меланомы сосудистой оболочки глаза (http://www.chat.ru/ neurocon).

Авионика. Обучаемые автоматы, распознавание сигналов радаров, адаптивное пилотирование сильно поврежденного самолета. Пример: автоматический переключатель режимов полета в реальном масштабе времени в зависимости от вида повреждения самолета.

Связь. Сжатие видеоинформации, быстрое кодирование-декодирование, оптимизация сотовых сетей и схем маршрутизации пакетов. Пример: специальная схема кодирования цветов (http://www.ee.duke.edu/ cec/JPL/paper.html), допускающая степень сжатия 240:1.

Интернет. Ассоциативный поиск информации, электронные секретари и агенты пользователя в сети, фильтрация информации в pusk-системах, коллаборативная фильтрация, рубрикация новостных лент, адресная реклама, адресный маркетинг для электронной торговли. Например, семейство AGENTWARE, создающих и использующих профили интересов пользователей в виде персональных автономных нейроагентов (нейросекретарей) (http://www.agentware.com).

Автоматизация производства. Оптимизация режимов производственного процесса, комплексная диагностика качества продукции, предупреждение аварийных ситуаций, робототехника. Например, нейросистема диагностики двигателей компанией Ford Motors.

Политические технологии. Анализ и обобщение социологических опросов, предсказание диагностики рейтингов, выявление значимых факторов, объективная кластеризация электората, визуализация социальной динамики населения.

Безопасность и охранные системы. Системы идентификации личности, распознавание голоса, лиц в толпе, распознавание автомобильных номеров, анализ аэрокосмических снимков, мониторинг информационных потоков, обнаружение подделок. Например, система обнаружения подделок чеков (Tearing up the Rules, Banking Technology, ноябрь 1993).

Ввод и обработка информации. Обработка рукописных чеков, распознавание подписей, отпечатков пальцев и голоса. Ввод в компьютер финансовых и налоговых

документов. Пример: пакеты серии FlexRead для распознавания и автоматического ввода рукописных платежных документов и налоговых деклараций.

Представители разных наук считают, что перевод известных методов на новый язык нейросетевых схем ничего принципиально нового не дает. Статистики говорят, что нейросети — это всего лишь частный способ статистической обработки данных, специалисты по оптимизации — что методы обучения нейросетей давно известны в их области, теория аппроксимации функций рассматривает нейросети наряду с другими методами многомерной аппроксимации. Нам же представляется, что именно синтез различных методов и идей в едином нейросетевом подходе и является неоценимым достоинством нейрокомпьютеринга. Единая терминология для широкого круга задач ускоряет и удешевляет разработку приложений. Но самое главное заключается в том, что нейросетевые алгоритмы заведомо параллельны и в будущем это даст огромный экономический эффект.

1.3.2 Информационное направление

Информационное направление разделяется на три.

1. *Эвристическое программирование* — это разработка оригинальных методов, алгоритмов решения задач, подобных человеческим, а в некоторых случаях даже и лучших. Под эвристикой понимается правило, стратегия, метод или прием, используемые для повышения эффективности системы, которая пытается найти решения сложных задач. Эвристическая программа — это программа для компьютера, использующая эвристики. В соответствии со словарем Вебстера «эвристический» означает «способствующий открытию».

Рис. 1.2 – Программы решения интеллектуальных задач.

Эвристические программы могут играть в шахматы, шашки, карточные игры, находить ответы на вопросы, находить решения из области математических исчислений; доказывать теоремы в математической логике и геометрии; способны обучаться на основе своего опыта; решать различные классы задач. Здесь

исследователь воспроизводит в компьютере методы, используемые людьми, т.к. интеллект человека выше интеллекта компьютера.

Структура программ решения интеллектуальных задач, предложенная Д. А. Поспеловым, представлена на рисунке 1.2 [5].

2. *Системы, основанные на знаниях.* Второе, основное, направление в искусственном интеллекте образует его фундамент. Именно здесь создается теория данного научного направления, решаются основные проблемы, связанные с центральным объектом изучения искусственного интеллекта Системы, основанные на знаниях знаниями.

Рис. 1.3 – Системы, основанные на знаниях.

На рис. 1.3 показана структура этого направления. Всякая предметная (проблемная) область деятельности может быть описана в виде некоторой совокупности сведений о структуре этой области, основных ее характеристиках, процессах, протекающих в ней, а также о способах решения возникающих в ней задач. Все эти сведения образуют знания о предметной области. При использовании интеллектуальных систем для решения задач в данной предметной области необходимо собрать о ней сведения и создать концептуальную модель этой области. Источниками знаний могут быть документы, статьи, книги, фотографии, киносъемка и многое другое. Из этих источников надо извлечь содержащиеся в них знания. Этот процесс может оказаться достаточно трудным, ибо надо заранее оценить важность

тех или иных знаний для работы интеллектуальной системы. Специалисты, которые занимаются всеми вопросами, связанными со знаниями, теперь называются инженерами по знаниям или инженерами знаний. Эта новая профессия порождена развитием искусственного интеллекта.

В области извлечения знаний можно выделить два основных направления: формализация качественных знаний и интеграция знаний. Первое направление связано с созданием разнообразных методов, позволяющих переходить от знаний, выраженных в текстовой форме, к их аналогам, пригодным для ввода в память интеллектуальной системы. В связи с этой проблемой развивались не только традиционные методы обработки экспериментальных данных, но и совершенно новое направление, получившее название нечеткой математики. Возникновение этого направления связано с именем американского специалиста Л. Заде. Нечеткая математика и ее методы оказали существенное влияние на многие области искусственного интеллекта и, в частности, на весь комплекс проблем, связанных с представлением и переработкой качественной информации.

Когда инженер по знаниям получает знания из различных источников, он должен интегрировать их в некоторую взаимосвязанную и непротиворечивую систему знаний о предметной области. Проблема интеграции знаний пока еще не стоит столь остро, но уже ясно, что без ее решения вряд ли будет возможно создавать представления о предметной области, обладающие теми же богатыми нюансами, которыми она обладает для специалиста. Знаний, содержащихся в источниках информации, отчужденных от специалиста, как правило, недостаточно.

Значительная часть профессионального опыта остается вне этих источников, в головах профессионалов, не могущих словесно их выразить. Такие знания часто называют профессиональным умением или интуицией. Для того, чтобы приобрести такие знания, нужны специальные приемы и методы. Они используются в инструментальных системах по *приобретению знаний*, создание которых — одна из современных задач инженерии знаний.

Полученные от экспертов знания нужно оценить с точки зрения их соответствия ранее накопленным знаниям и формализовать их для ввода в память интеллектуальной системы. Кроме того, знания, полученные от различных экспертов, надо еще согласовать между собой. Нередки случаи, когда эти знания оказываются внешне несовместимыми и даже противоречивыми. Инженер по знаниям должен путем опроса экспертов устранить эти противоречия. Следующая большая проблема, изучаемая в искусственном интеллекте, — это *представление знаний* в памяти системы. Для этого разрабатываются разнообразные модели представления знаний. В настоящее время в интеллектуальных системах используются четыре основные модели знаний. Первая модель, возможно, наиболее близка к тому, как представляются знания в текстах на естественном языке. В ее основе лежит идея о том, что вся необходимая информация может быть описана как совокупность троек вида: (a R b), где a и b два объекта или понятия, а R — двоичное отношение между ними. Такая модель графически может представляться в виде сети, в которой вершинам соответствуют объекты или понятия, а дугам — отношения между ними. Дуги помечены именами соответствующих отношений. Такая модель носит название семантической сети. Впервые такое представление знаний под названием «язык синтагматических цепей» было, по-видимому, использовано в работах по

ситуационному управлению, получивших развитие в СССР в середине шестидесятых годов.

Семантические сети, в зависимости от характера отношений, допустимых в них, имеют различную природу. В ситуационном управлении эти отношения, в основном, описывали временные, пространственные и каузальные связи между объектами, а также результаты воздействия на объекты со стороны управляющей системы. В системах планирования и автоматического синтеза программ эти отношения являются связями типа «цель-средство» или «цель-подцель». В классифицирующих системах отношения передают связи по включению объемов понятий (типа «род-вид», «класс-элемент» и т. п.). Распространены так называемые функциональные семантические сети, в которых дуги характеризуют связи вида «аргумент-функция». Такие сети используются в качестве моделей вычислительных процессов или моделей функционирования дискретных устройств.

Таким образом, семантические сети — модель весьма широкого назначения. Теория семантических сетей еще не завершена, что привлекает к ней внимание многих специалистов, работающих в искусственном интеллекте. При различных синтаксических ограничениях на структуру семантической сети возникают более жесткие типы представления. Например, реляционные представления, характерные для реляционных баз данных, или каузальные представления в логике, получившие широкое распространение в машинных методах логического вывода или в языках логического программирования типа языка Пролог. В некотором смысле, фреймовые представления знаний, широко распространенные в искусственном интеллекте, также являются видом семантических сетей, для перехода к которому надо удовлетворить ряд ограничений синтаксического характера. С понятием «фрейм» в искусственном интеллекте произошла некоторая трансформация смысла. Это понятие было введено в научный оборот М. Минским, который под фреймом некоторого объекта или явления понимал то его минимальное описание, которое содержит всю существенную информацию об этом объекте или явлении и обладает тем свойством, что удаление из описания любой его части приводит к потере существенной информации, без которой описание объекта или явления не может быть достаточным для их идентификации. Однако позже в работах по представлению знаний [11, 12] требование по минимальности описания перестали соблюдать и под фреймами стали понимать структуры вида: <имя фрейма; (множество слотов)>. Каждый слот есть пара вида: (имя слота, значение слота). Допускается, чтобы слот сам был фреймом. Тогда в качестве значений слота выступает множество слотов. Другими возможностями для заполнения слотов могут быть константы, переменные, любые допустимые выражения в выбранной модели знаний, ссылки на другие слоты и фреймы и т. п. Таким образом, фрейм представляет собой достаточно гибкую конструкцию, позволяющую отображать в памяти интеллектуальной системы разнообразные знания.

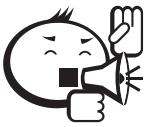
Две другие распространенные модели знаний опираются на классическую логическую модель вывода. Это либо логические исчисления, типа исчисления предикатов и его расширения, либо системы продукций, т. е. правила вида: «Если А, то Б», задающих элементарные шаги преобразований и умозаключений. Эти две модели знаний отличаются явно выраженной процедурной формой. Поэтому часто говорят, что они описывают процедурные знания, а модели знаний, опирающиеся

на семантические сети, описывают декларативные знания. Оба вида знаний могут сосуществовать друг с другом. Например, в качестве значений некоторых слотов во фрейме могут выступать продукции.

Именно такие смешанные представления оказываются сейчас в центре внимания исследователей, так как они сулят наиболее хорошие перспективы по представлению знаний.

Перечисленные модели знаний возникли в искусственном интеллекте как бы насильственно. Они не опираются на аналоги структур для представления знаний, которыми пользуются люди. Это связано с плохой изученностью форм представления знаний у человека.

В интеллектуальных системах для хранения и использования знаний создаются специальные представления знаний, включающие в свой состав всю совокупность процедур, необходимых для записи знаний, извлечения их из памяти и поддержки хранилища знаний в рабочем состоянии. Системы представления знаний часто оформляются как базы знаний, являющиеся естественным развитием баз данных. Теория баз знаний составляет заметную часть современного искусственного интеллекта.



Именно в них сосредотачиваются сейчас все основные процедуры *манипулирования знаниями*.

Среди этих процедур можно, прежде всего, отметить процедуры пополнения знаний. Все человеческие знания, содержащиеся в текстах, таковы, что они принципиально не полны. Воспринимая тексты, мы как бы пополняем их за счет той информации, которая нам известна и которая имеет отношение к данному тексту. Аналогичные процедуры должны происходить в базах знаний. Новые знания, поступающие в них, должны вместе с теми сведениями, которые уже были ранее записаны в базу, сформировать расширение поступивших знаний. Среди этих процедур особое место занимают псевдофизические логики (времени, пространства, действий и т. п.), которые, опираясь на законы внешнего мира, пополняют поступающую в базу знаний информацию.

Знания, как и у человека, в интеллектуальных системах хранятся не бессистемно. Они образуют некоторые упорядоченные структуры, что облегчает поиск нужных знаний и поддержание работоспособности баз знаний. Для этого используются различные классифицирующие процедуры. Типы классификаций могут быть весьма различными. Это могут быть родовидовые классификации, классификации типа «часть-целое» или ситуативные классификации, когда в одно множество объединяются все те знания, которые релевантны некоторой типовой ситуации. В этой области исследования по искусственному интеллекту тесно соприкасаются с теорией классификации, давно существующей, как некоторая самостоятельная ветвь науки.

В процессе классификации часто происходит абстрагирование от отдельных элементов описаний, от отдельных фрагментов знаний об объектах и явлениях. Это приводит к появлению обобщенных знаний. Обобщение может идти на несколько шагов, что приводит, в конце концов, к абстрактным знаниям, для которых нет

прямого праобраза во внешнем мире. Манипулирование абстрактными знаниями повышает интеллектуальные возможности систем, делая эти манипуляции весьма общими по своим свойствам и результатам. Обобщение знаний и формирование понятий в системах искусственного интеллекта — одно из активно развивающихся направлений, в котором работает немало специалистов.

Особое место занимают процедуры, связанные с выводом на знаниях, получением на основании имеющихся знаний новых знаний. Вывод на знаниях зависит от той модели, которая используется для их представления. Если в качестве представления используются логические системы или продукции, то вывод на знаниях весьма близок к стандартному логическому выводу. Это же происходит при представлении знаний в каузальной форме. Во всех случаях в интеллектуальных системах используются методы вывода, опирающиеся на идею обратного вывода С. Ю. Маслова (как на языке ПРОЛОГ при клаузальной форме представления). Но простое заимствование идей и методов математической логики, под знаком чего происходило развитие работ в искусственном интеллекте в семидесятых годах, не привело к сколько-нибудь значительным результатам. Основное отличие баз знаний и баз данных интеллектуальных систем от тех объектов, с которыми имеет дело формальная логическая система, это их открытость. Возможность появления в памяти интеллектуальной системы новых файлов, новых сведений приводит к тому, что начинает разрушаться принцип монотонности, лежащий в основе функционирования всех систем, изучаемых математической логикой.

Согласно этому принципу если некоторое утверждение выводится в данной системе, то никакие дополнительные сведения не могут изменить этот факт. В открытых системах это не так. Новые сведения могут изменить ситуацию, и сделанный ранее вывод может стать неверным.

Немонотонность вывода в открытых системах вызывает немалые трудности при организации вывода на знаниях. В последнее десятилетие сторонники логических методов в искусственном интеллекте делают попытки построить новые логические системы, в рамках которых можно было бы обеспечить немонотонный вывод. На этом пути больше трудностей, чем результатов. И дело не только в немонотонности вывода. По сути, системы, с помощью которых представляются знания о предметных областях, не являются строго аксиоматическими, как классические логические исчисления. В последних аксиомы описывают извечные истины, верные для любых предметных областей. А в интеллектуальных системах каждая предметная область использует свои, специфические, верные только в ней утверждения. Поэтому и системы, которые возникают при таких условиях, следует называть квазиаксиоматическими. В таких системах вполне возможна смена исходных аксиом в процессе длительного вывода и, как следствие, изменение этого вывода.

И, наконец, еще одна особенность вывода на знаниях, доставляющая немало забот исследователям, занятым формированием решений в интеллектуальных системах. Это неполнота сведений о предметной области и протекающих в ней процессах, неточность входной информации, не совсем полная уверенность в квазиаксиомах. А это означает, что выводы в интеллектуальных системах носят не абсолютно достоверный характер, как в традиционных логических системах, а приближенный, правдоподобный характер. Такие выводы требуют развитого аппарата

вычисления оценок правдоподобия и методов оперирования с ними. Подобные исследования сейчас в искусственном интеллекте развиваются широким фронтом. По сути, рождается новая теория вывода, в которую лишь как небольшая часть входит достоверный вывод, изучавшийся в течение многих десятилетий логиками.

В интеллектуальных системах специалисты стремятся отразить основные особенности человеческих рассуждений, опыт тех специалистов, которые обладают профессиональными умениями, пока не полностью доступными искусственным системам. Поэтому бурно развивается та область, которую в искусственном интеллекте называют моделированием человеческих рассуждений.

К ним относятся аргументации на основе имеющихся знаний, рассуждения по аналогии и ассоциации, оправдание заключения в системе имеющихся прагматических ценностей и многое другое, чем люди пользуются в своей практике. Привнесение всех этих приемов в интеллектуальные системы, без сомнения, сделает их рассуждения более гибкими, успешными и человечными.

Когда некто высказывает свое мнение, то для того чтобы согласиться или не согласиться с ним, необходимо знать те же основания, которые лежат в основе его мнения. Если эти основания неизвестны, то имеется возможность попросить своего оппонента объяснить, как он пришел к своему мнению. Аналогичная функция возникла и в интеллектуальных системах. Поскольку они принимают свои решения, опираясь на знания, которые могут быть неизвестны пользователю, решающему свою задачу с помощью интеллектуальной системы, то он может усомниться в правильности полученного решения. Интеллектуальная система должна обладать средствами, которые могут сформировать пользователю необходимые *объяснения*. Объяснения могут быть различного типа. Они могут касаться самого процесса решения оснований, которые были для этого использованы, способов отсечения альтернативных вариантов и т. п. Все это требует развитой теории объяснения, что стимулирует сейчас активность исследований в этом направлении.

3. *Интеллектуальное программирование.* Это направление в искусственном интеллекте соответствует программистскому взгляду на эту область. Трудоемкость разработки интеллектуальных приложений зависит от использованного языка, инструментальных систем, парадигмы программирования, средств разработки ИИС и приобретения знаний, систем когнитивной графики (рис. 1.4) [2, 4].

Среди огромного количества языков программирования в работах по искусственному интеллекту используется небольшая часть.

1) Традиционные языки программирования типа C, C++, но, как правило, для создания инструментальных средств.

2) Специализированные языки программирования:

   - LISP и все его многочисленные версии, ориентированные на обработку списков;
   - язык логического программирования PROLOG. В самом начале он был стимулирован к жизни работами в области машинного перевода, но Р. Ковальский понял, что в идее каузальных логических выражений и в процедурах вывода кроется куда более общая идея, соответствующая логическому выводу;
   - язык рекурсивных функций РЕФАЛ.

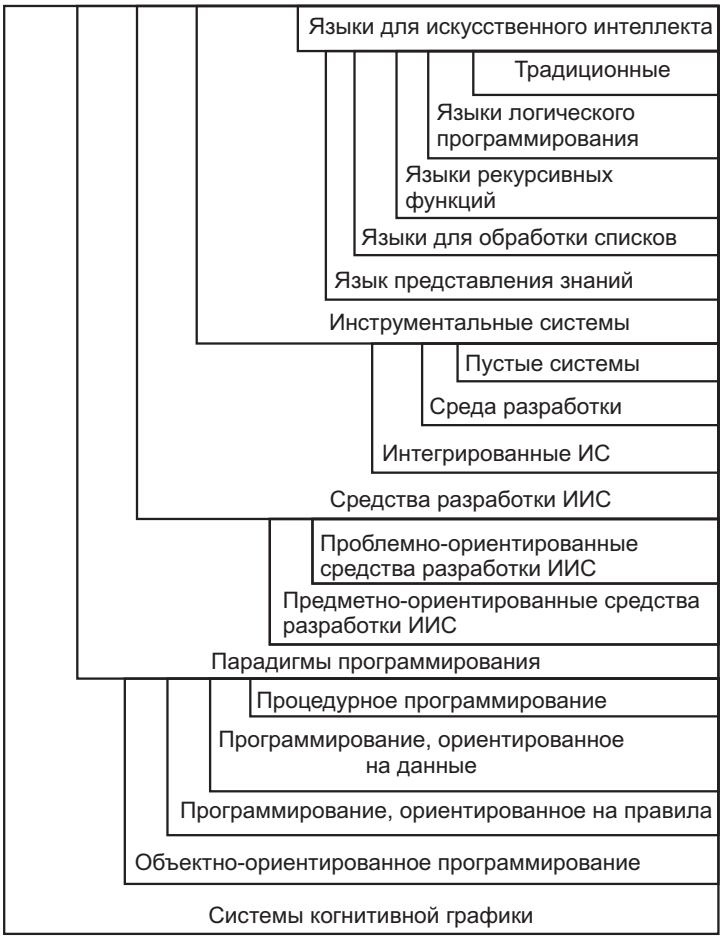


Рис. 1.4 – Инструментальные средства интеллектуальных систем.

3) Особняком стоят языки для представления знаний. Это языки, ориентированные на фреймы KL-1, KRL, FRL или язык ПИЛОТ, ориентированный на модель знаний в виде продукций

Инструментальные системы развиваются достаточно быстро и предназначены для быстрого проектирования и разработки самых разнообразных интелектуальных систем. Общая идея состоит в том, чтобы создать некоторую систему-прототип, затратив на ее создание достаточно много усилий. Но затем использовать для решения задач в конкретной предметной области.

Если в системе-прототипе заранее зафиксированы средства заполнения базы знаний и манипулирования знаниями в ней, но сама база знаний не заполнена, то такая инструментальная система называется пустой.

Сейчас интерес к пустым системам резко уменьшился. Оказалось, что даже для однотипных предметных областей переход от одной области к другой может потребовать модификации тех или иных средств для манипулирования знаниями, а иногда и формы представления знаний. Поэтому основные усилия разработчиков сейчас направлены на создание систем-оболочек. В таких инструментальных системах имеется возможность при переходе к конкретной системе варьировать в достаточно широких пределах форму представления знаний и способы манипулирования ими. Здесь программисты, специализирующиеся в области

интеллектуальных программных систем, ожидают большего прорыва в интеллектуальное программирование завтрашнего дня.

Разработаны ИС или (ЭС) общего назначения, содержащие все программные компоненты, которые не требуют от разработчиков приложений знания программирования и не имеют БЗ, т. е. знаний о конкретных предметных средах. Например, ЭКО, Leonardo, Nexpert Object, Kappa, EXSYS, GURU, APT, KEE и т. д.

В последнее время «оболочка» заменяется термином «среда разработки». Если инструментальная система используется на стадии разработки, использования и сопровождения употребляется термин «полная среда» (*complete environment*). Примерами таких интегрированных инструментальных систем типа Work bench являются:

KEATS, Shelly, VITAL. Для ясности перечислим компоненты для первой системы:

- средства фрагментирования текстовых источников знаний;
- язык представления знаний средствами фреймовой модели;
- графический интерфейс для создания гипертекстов и концептуальных моделей;
- интерпретатор правил прямого и обратного выводов;
- инструмент визуализации логического вызова;
- интерфейс манипулирования таблицами в БЗ;
- язык описания и распространения ограничений;
- немонотонная система поддержания истинности.

Средства разработки интеллектуальных систем. Сюда отнесем проблемно/предметно-ориентированные оболочки и среды:

- проблемно-ориентированные средства, которые предназначены для решения задач определенного класса (прогнозирования, планирования, управления и т. д.) и содержит соответствующие функциональные модули;
- предметно-ориентированные средства — это знания о типах предметной области.

К парадигмам программирования отнесем следующие:

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

В первой парадигме процедурам отводится активная роль и активизируется она вызовом. Используются для описания детерминированной последовательности действий процесса(ов).

Во второй парадигме данным отводится активная роль, т. е. совершаются действия (процедуры) при обращении к этим активным данным.

В парадигме, ориентированной на правила, задаются правила «условие-действие», и при сопоставлении образа БД и образца БЗ правило выполняется. Поведение системы здесь не задается заранее известной последовательностью правил, а формируется на основе данных в текущий момент.

В объектно-ориентированной парадигме программа организуется вокруг сущностей, называемых объектами, которые включают локальные процедуры и локальные данные (переменные). Поведение (функционирование) здесь осуществляется пересылкой сообщений между объектами. Объект, получив сообщение, осуществляет его локальную интерпретацию, основываясь на локальных процедурах и данных.

Системы когнитивной графики одно из направлений в интеллектуальном программировании. Одна из центральных идей искусственного интеллекта — это идея о том, что суть самого феномена интеллекта состоит в совместной работе двух систем переработки информации: зрительной, создающей образную картину мира, и символической, способной к абстрактному мышлению, к оперированию с понятиями, интегрирующими образы внешнего мира. Возможность перехода от зрительной картины к ее текстовому (символическому) описанию и от текста к некоторой зрительной картине, составляет, по-видимому, основу того, что называется мышлением.

Успехи искусственного интеллекта определяются тем, что символьная система достаточно хорошо изучена и промоделирована в искусственных системах. Со зрительной системой дело обстоит хуже. Мы пока еще точно не знаем о том, как хранятся зрительные образы в памяти человека, как они обрабатываются, как они соотносятся с текстами, им соответствующими.

Когнитивная графика и занимается приемами соотнесения текстов и зрительных картин через общее представление знаний, интегрирующих текстовые и зрительные образы. Это направление признается очень перспективным: новые способы решения задач и переход к новой технологии их решения.

Примерами являются программы оживления картин, но не на основе жестких процедур, а в соответствии с некоторыми текстами на ограниченном естественном языке.

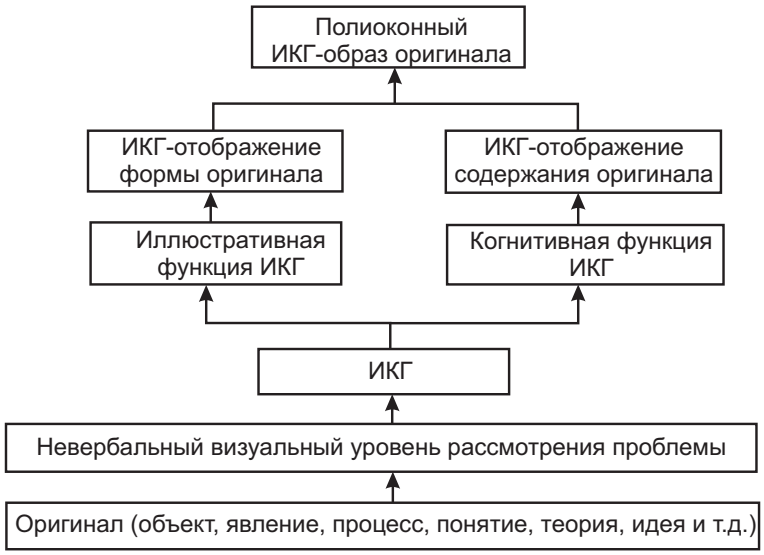


Рис. 1.5 – Функциональное содержание ИКГ.

Если интерактивная компьютерная графика реализует две связанные между собой функции: иллюстративную и когнитивную, то одновременный вывод

ИКГ-изображений в разные окна дисплея создает у пользователя синтетический полиоконный ИКГ-образ. Иллюстративная функция обеспечивает визуальную адекватность графического образа оригиналу, т. е. визуальную «узнаваемость» оригинала. Когнитивная функция позволяет (при определенных условиях) изображать в наглядной графической форме внутреннее содержание оригинала. Функциональное содержание ИКГ представлено на рис. 1.5 [4].

Авторы, конечно, осознают, что дать полную картину исследований систем искусственного интеллекта в одной книге невозможно, поэтому приведем классификации систем искусственного интеллекта (направлений) наиболее значимых, как нам кажется, для цельного представления проблемы.

1.3.3 Примеры различных классификаций систем искусственного интеллекта

А. В. Андрейчиков и О. Н. Андрейчикова предлагают следующие классы систем:

1) Интеллектуальная информационная система — одно из главных направлений в искусственном интеллекте. Целью такой системы является исследование и применение знаний высококвалифицированных экспертов для решения задач. Это имитация человеческого искусства анализа неструктурированных проблем, т. е. разработка моделей представления, извлечения и структурирования знаний. Частным случаем таких систем являются экспертные системы.
2) Естественно-языковой интерфейс и машинный перевод. Это исследование методов и разработка систем, обеспечивающих реализацию процесса общения человека с компьютером на естественном языке (системы Е-Я общения). Системы машинного перевода с одного языка на другой больших потоков информаций — это интеллектуальная система, состоящая из базы знаний и сложных моделей, базирующихся на структурно-логическом подходе, включающих анализ и синтез естественно-языковых сообщения.
3) Генерация и распознавание речи. Системы речевого общения создаются в целях повышения скорости ввода информации в ЭВМ, разгрузки зрения и рук, а также для реализации речевого общения на значительном расстоянии.
4) Обработка визуальной информации. В таких системах решаются задачи обработки, анализа и синтеза изображений. Задача обработки изображений — это трансформирование графических образов в новые изображения. Задача анализа — это преобразование исходных изображений в данные другого типа (например, текст). И наконец, синтез изображений — получение графических объектов по алгоритмам построения изображения (машинная графика).
5) Обучение и самообучение. Это разработка моделей, методов и алгоритмов для систем автоматического накопления и формирования знаний с использование процедур анализа и обобщения данных. Это системы Data-mining, Knowledge, Discovery и другие.

6) Расппознавание образов. Система распознавания объектов осуществляет отнесение объектов к классам, а классы описываются совокупностями определенных значений признаков.

7) Игры и машинное творчество. Машинное творчество — это музыкальные программы, машинные программы живописи и графики, стихов, прозаических произведений, изобретение новых объектов, интеллектуальные компьютерные игры.

8) Программное обеспечение систем искусственного интеллекта. Инструментальные средства для разработки систем искусственного интеллекта включают специальные языки программирования, ориентированные на обработку символьной информации (LISP, SMALLTALK), языки логического программирования (PROLOG), языки представления знаний (OPS-5, KRL, FRL), интегрированные программные среды, содержащие арсенал инструментальных средств (KE, ARTS, GURU, G2), а также оболочки экспертных систем (BUILD, EMYCIN, EXSYS, Professional, ЭКСПЕРТ).

9) Новые архитектуры компьютеров. Это создание компьютеров не фон-неймовской архитектуры, ориентированные на обработку символьной информации. Известны параллельные и векторные компьютеры, но весьма высокой стоимости в настоящее время.

10) Интеллектуальные роботы. В настоящее время используются программируемые манипуляторы с жесткой схемой управления.

Гаскаров Д. В. предлагает следующие наиболее распространенные типы классификации интеллектуализированных систем.

1) По степени реализации (стадии существования) рис. 1.6.

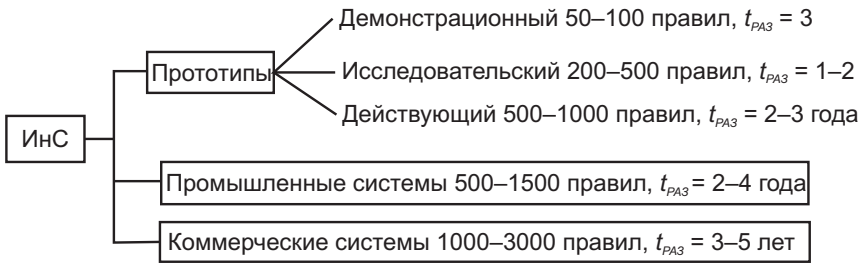


Рис. 1.6 – Классификация ИнС по степени реализации.

2) По степени сложности (рис. 1.7).

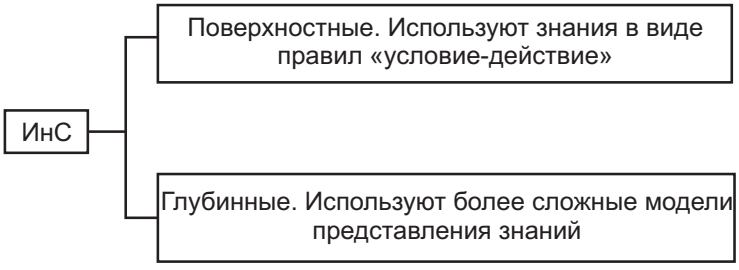


Рис. 1.7 – Классификация ИнС по степени сложности.

3) По степени интеграции (рис. 1.8).

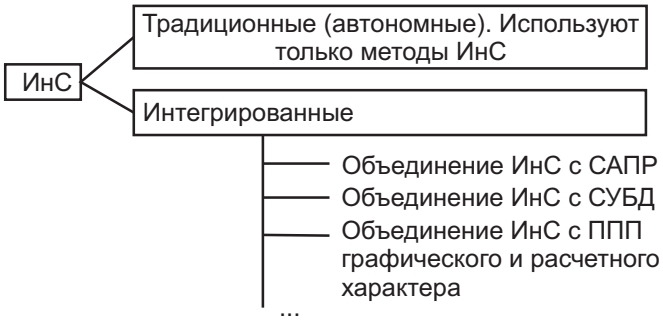


Рис. 1.8 – Классификация ИнС по степени интеграции.

4) По сложности и типу ЭВМ (рис. 1.9).
5) По типу области экспертизы (типу предметной области, рис. 1.10).

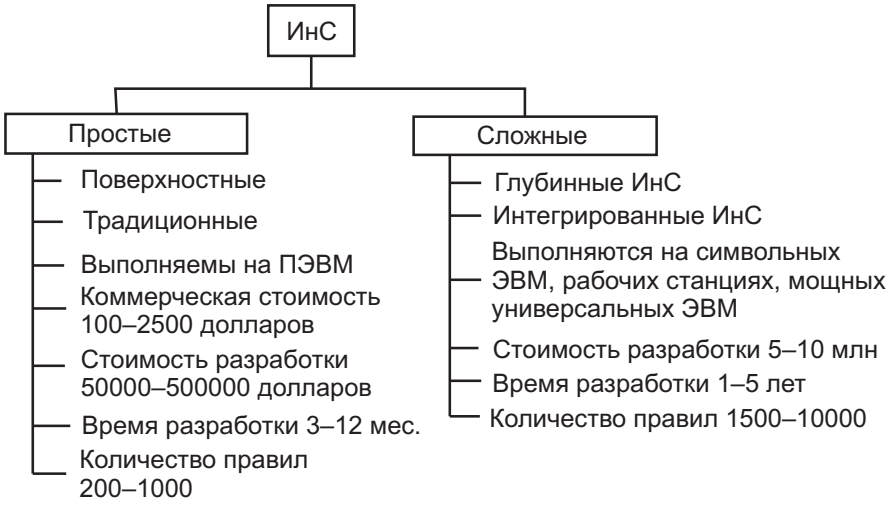


Рис. 1.9 – Классификация ИнС по сложности и типу ЭВМ.

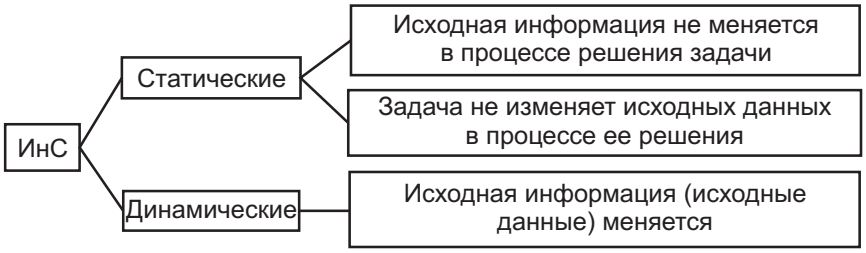


Рис. 1.10 – Классификация ИнС по типу области экспертизы.

6) По эволюции развития (рис. 1.11).
7) По типам решаемых задач (рис. 1.12).

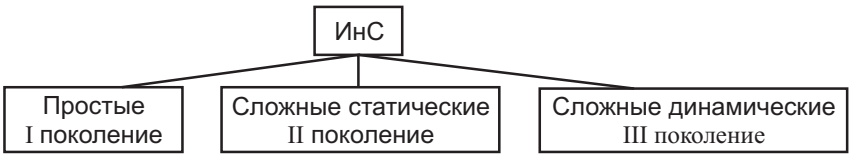


Рис. 1.11 – Классификация ИнС по эволюции развития.

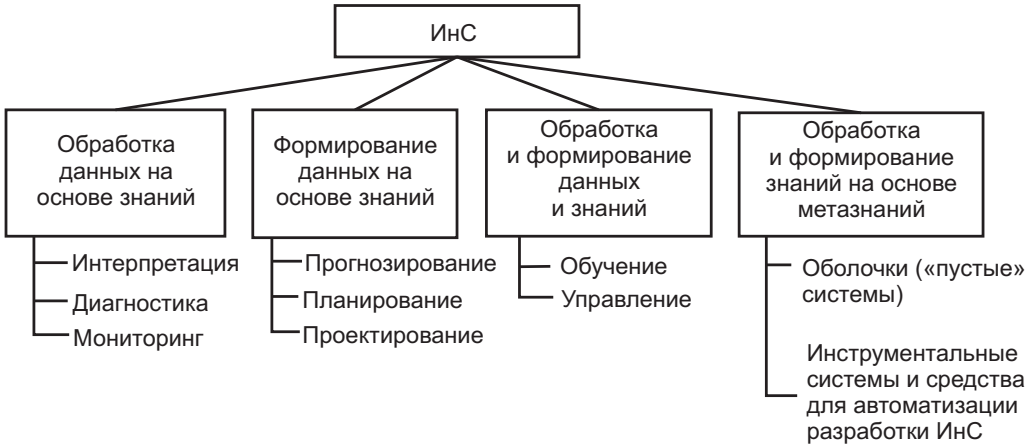


Рис. 1.12 – Классификация ИнС потипам решаемых задач.

Ясницкий Л. Н. классифицирует системы искусственного интеллекта по следующим направлениям:

1) Системы, основанные на знаниях.
2) Нейросетевые и нейрокомпьютерные технологии.
3) Распознавание образов.
4) Игры и творчество.
5) Компьютерная лингвистика.
6) Интеллектуальные роботы.
7) Компьютерные вирусы.
8) Интеллектуальное компьютерное моделирование.

В. Н. Бондарев, Ф. Г. Аде считают основными направлениями исследований в искусственном интеллекте следующие:

1) Представление задач и поиск решений.
2) Доказательство теорем.
3) Представление знаний.
4) Экспертные системы.
5) Обучение и выявление закономерностей.
6) Общение на естественном языке.
7) Распознавание образов.
8) Компьютерное зрение.
9) Языки программирования систем искусственного интеллекта.

Е. В. Луценко предлагает рассматривать следующие классы систем искусственного интеллекта [8]:

Системы с интеллектуальной обратной связью и интеллектуальными интерфейсами. Интеллектуальный интерфейс (Intelligent interface) — интерфейс непосредственного взаимодействия ресурсов информационного комплекса и пользователя посредством программ обработки текстовых запросов пользователя.

Примером может служить программа идентификации и аутентификации личности по почерку. Аутентификация — это проверка, действительно ли пользователь является тем, за кого себя выдает. При этом пользователь должен предварительно сообщить о себе идентификационную информацию: свое имя и пароль, соответствующий названному имени. Идентификация — это установление его личности. И идентификация, и аутентификация являются типичными задачами распознавания образов, которое может проводиться по заранее определенной или произвольной последовательности нажатий клавиш.

Системы с биологической обратной связью (БОС) — системы, поведение которых зависит от психофизического (биологического) состояния пользователя.

- Мониторинг состояния сотрудников на конвейере с целью обеспечения высокого качества продукции.
- Компьютерные тренажеры для обучения больных с функциональными нарушениями управлению своим состоянием.
- Компьютерные игры с БОС.

Системы с семантическим резонансом — системы, поведение которых зависит от состояния сознания пользователя и его психологической реакции на смысловые стимулы.

Системы виртуальной реальности.

1) Виртуальная реальность (ВР) — модельная трехмерная (3D) окружающая среда, создаваемая компьютерными средствами и реалистично реагирующая на взаимодействие с пользователями. Технической базой системы ВР являются современные мощные персональные компьютеры и программное обеспечение высококачественной трехмерной визуализации и анимации. В качестве устройств ввода-вывода в системах ВР применяются виртуальные шлемы с дисплеями, в частности шлемы со стереоскопическими очками, и устройства 3D-ввода, например мышь с пространственно управляемым курсором или «цифровые перчатки», которые обеспечивают тактильную обратную связь с пользователем.
2) Автоматизированные системы распознавания образов. Система распознавания образов — это класс систем искусственного интеллекта, обеспечивающих:
   - формирование конкретных образов объектов и обобщенных образов классов;
   - обучение, т. е. формирование обобщенных образов классов на основе ряда примеров объектов, классифицированных (т. е. отнесенных к тем или иным категориям — классам) учителем и составляющих обучающую выборку;

- самообучение, т. е. формирование кластеров объектов на основе анализа неклассифицированной обучающей выборки;
- распознавание, т. е. идентификацию (и прогнозирование) состояний объектов, описанных признаками, друг с другом и с обобщенными образами классов;
- измерение степени адекватности модели;
- решение обратной задачи идентификации и прогнозирования (обеспечивается не всеми моделями).

3) Автоматизированные системы поддержки принятия решений. Системы поддержки принятия решений (СППР) — это компьютерные системы, почти всегда интерактивные, разработанные, чтобы помочь менеджеру (или руководителю) в принятии решений управления, объединяя данные, сложные аналитические модели и удобное для пользователя программное обеспечение в единую мощную систему, которая может поддерживать слабоструктурированное и неструктурированное принятие решения. СППР находится под управлением пользователя от начала до реализации и используется ежедневно. Предназначена для автоматизации выбора рационального варианта из исходного множества альтернативных в условиях многокритериальности и неопределенности исходной информации.

4) Экспертные системы. Экспертная система (ЭС) — это программа, которая в определенных отношениях заменяет эксперта или группу экспертов в той или иной предметной области. ЭС предназначены для решения практических задач, возникающих в слабоструктурированных и трудно формализуемых предметных областях. Исторически, ЭС были первыми системами искусственного интеллекта, которые привлекли внимание потребителей. Экспертные системы используются в маркетинге для сегментации рынка и выработки маркетинговых программ, а также в банковском деле для определения тенденции рынка, в трейдинге для программирования котировок акций и валют, в аудите для подготовки заключений о финансовом состоянии предприятий.

5) Генетические алгоритмы и моделирование эволюции. Генетические алгоритмы (ГА) — это адаптивные методы функциональной оптимизации, основанные на компьютерном имитационном моделировании биологической эволюции. Генетический алгоритм — новейший способ решения задач оптимизации.

6) Когнитивное моделирование. Это способ анализа, обеспечивающий определение силы и направления влияния факторов на перевод объекта управления в целевое состояние с учетом сходства и различия в влиянии факторов на объект управления.

Основано на когнитивной структуризации предметной области, т. е. на выявление будущих целевых и нежелательных состояний объекта управления и наиболее существенных (базисных) факторов управления и внешней среды, влияющих на переход объекта в эти состояния, а также установление на качественном уровне причинно-следственных связей между ними, с учетом взаимовлияния

факторов друг на друга. Результаты когнитивной структуризации отображаются с помощью когнитивной карты (модели). Например, в экономической сфере это позволяет в сжатые сроки разработать и обосновать стратегию экономического развития предприятия, банка, региона или даже целого государства с учетом влияния изменений во внешней среде; в сфере финансов и фондового рынка — учесть ожидания участников рынка.

7) Выявление знаний из опыта (эмпирических фактов) и интеллектуальный анализ данных (Data mining). Ителлектуальный анализ данных (ИАД или Data mining) — это процесс обнаружения в «сырых» данных ранее неизвестных, нетривиальных, практически полезных и доступных интерпретации знаний, необходимых для принятия решений в различных сферах человеческой деятельности. Достижения технологи data mining активно используются в банковском деле для решения проблем телекоммуникации, анализа биржевого рынка и др.
8) Нейронные сети. Искусственная нейронная сеть (ИНС, нейросеть) — это набор нейронов, соединенных между собой. Как правило, передаточные функции всех нейронов в сети фиксированы, а веса являются параметрами сети и могут изменяться. Некоторые входы нейронов помечены как внешние входы сети, а некоторые выходы — как внешние выходы сети. Подавая любые числа на входы сети, мы получаем какой-то набор чисел на выходах сети. Практически любую задачу можно свести к задаче, решаемой нейросетью.

Таким образом, анализ систем (направлений) искусственного интеллекта показал, что в основном авторы вышепредставленных классификаций повторяют направления Д. А. Поспелова с небольшими изменениями, указывающими на приоритеты авторов и на развитие систем искусственного интеллекта в последнее время.

Проанализировав вышеприведенные классификации, мы можем предложить следующие направления искусственного интеллекта.

1) Экспертные системы I поколения (простые, сложные) и II поколения (интегрированные, многофункциональные, интеллектуальные, креативные (творческие) и гибридные).
2) Интеллектуальные производственные системы: вопрос-ответные системы, расчетно-логические системы, интеллектуальные САПР, САНИ, АСУ, СППР.
3) Нейросети и нейрокомпьютеры (нейросетевые и нейрокомпьютерные технологии).
4) Построение и автоматизация построения БЗ, анализ, обработка и выявления знания.
5) Обучение и самообучение (консультационные системы, интеллектуальные тренажеры, системы школьного и вузовского образования).
6) Эволюционное моделирование (генетические алгоритмы, классифицирующие системы, генетическое программирование, эволюционное программирование, эволюционные стратегии)
7) Системы машинного перевода.

8) Системы Е-Я общения (ведение диалога, понимание-преобразование высказываний с Е-Я на внутренний язык, обработка высказываний — формулирование выходных высказываний на Е-Я).
9) Системы речевого общения (синтез (текст-смысл), анализ и распознавание речи (смысл-текст)).
10) Системы обработки визуальной информации (обработка, анализ и синтез изображений).
11) Системы распознавания образов.

Иногда в литературе системы машинного перевода и системы Е-Я общения объединяются в одно направление — компьютерную лингвистику. И последнее — авторы считают целесообразным выделить интеллектуальные системы управления, наиболее актуальную часть систем, основанных на знаниях (статические, динамические, реального времени).

Можно выделить и менее значимые направления систем искусственного интеллекта:

1) Игры и машинное творчество.
2) Архитектуры компьютеров.
3) Компьютерные вирусы.
4) Интеллектуальное математическое моделирование.
5) Интеллектуальные роботы.
6) Системы с интеллектуальной обратной связью и интеллектуальными интерфейсами.
7) Многоагентные системы.

Контрольные вопросы и задания к главе 1

1) Проведите анализ представленных определений искусственного интеллекта.
2) Сформулируйте определение искусственного интеллекта, данное Д. А. Поспеловым.
3) Какие сложные задачи решает искусственный интеллект?
4) Проведите сравнение интеллектуальных систем в докреативный и креативный периоды их развития.
5) Представьте определение СИИ.
6) Приведите примеры интеллектуальных систем.
7) Расскажите о трех определениях для интеллектуальных систем, представленных Гаскаровым Д. Б.
8) Дайте характеристику двух целей искусственного интеллекта.
9) Сформулируйте два основных направления искусственного интеллекта.

10) Расскажите о становлении искусственного интеллекта.
11) Проведите анализ эвристического поиска и доказательства теорем при решении задач.
12) Опишите представление знаний в интеллектуальных системах.
13) Расскажите об этапе разработки и становления интеллектуальных систем I поколения.
14) Расскажите об этапе разработки и становления интеллектуальных систем II поколения.
15) Сравните две точки зрения на область искусственного интеллекта.
16) Опишите нейросети трех групп.
17) Представьте классификацию систем нейронной обработки.
18) Представьте классификацию нейронных систем по типу входных и выходных сигналов.
19) Представьте классификацию интеллектуальных нейронных систем по параметрам управления.
20) Назовите классификацию нейросетей по типу связей и типу обучения.
21) Приведите предметные области, использующие нейросетевые задачи.
22) На какие три части делится информационное направление (вторая точка зрения на искусственный интеллект)?
23) Опишите программы решения интеллектуальных задач в информационном направлении.
24) Опишите системы, основанные на знаниях в информационном направлении.
25) Опишите интеллектуальное программирование в информационном направлении.
26) Что вы понимаете под когнитивной графикой?
27) Представьте функциональное содержание интерактивной компьютерной графики.
28) Приведите примеры различных классов систем искусственного интеллекта.
29) Дайте характеристику классификации Д. В. Гаскарова систем искусственного интеллекта.
30) Сформулируйте классификацию Л. Н. Ясницкого, В. П. Бондарева, Е. В. Луценко.

Литература к главе 1

[1] Андрейчиков А. В. Интеллектуальные информационные системы : учебник / А. В. Андрейчиков, О. Н. Андречикова. — М. : Финансы и статистика, 2006. — 424 с. — №12.

[2] Бондарев В. Н. Искусственный интеллект : учеб. пособие для вузов / В. Н. Бондарев, Ф. Г. Аде — Севастополь : Изд-во СевНТУ, 2002. — 615 с.

[3] Гаскаров Д. В. Интеллектуальные информационные системы: учебник для вузов / Д. В. Гаскаров. — М. : Высш. шк., 2003. — 431 с.

[4] Искусственный интеллект : справочник: в трех кн. / под ред. Д. А. Поспелова. — М. : Радио и связь, 1990. — Кн. 2 : Модели и методы.

[5] Искусственный интеллект : справочник: в трех кн. / под ред. Э. В. Попова. — М. : Радио и связь, 1990.

[6] Ларичев О. И. Системы основанные на экспертных знаниях: история, совершенное состояние и некоторые перспективы // Труды Седьмой национальной конференции по искусственному интеллекту с международным участием. — М. : Изд-во физико-математической литературы, 2000.

[7] Луценко Е. В. Интеллектуальные информационные системы / Е. В. Луценко. — Краснодар : КубГАУ, 2006. — 615 с.

[8] Оссовский С. Нейронные сети для обработки информации : пер. с польского Н. Д. Руданского / С. Оссовский. — М. : Финансы и статистика, 2002. — 344 с.

[9] Павлов С. Н. Интеллектуальные информационные системы : учеб. пособие / С. Н. Павлов. — Томск : Томский межвузовский центр дистанционного образования, 2004. — 328 с.

[10] Попов Э. В. Алгоритмические основы интеллектуальных роботов и искусственного интеллекта / Э. В. Попов, Г. Р. Фирдман. — М. : Изд-ва «Наука», 1976. — 456 с.

[11] Представление и исследование знаний : пер. с япон. / Х. Уэно [и др.] ; под ред. Х. Уэно, М. Исидзука; — М. : Мир, 1989. — 220 с.

[12] Приобретение знаний : пер. с япон. / под ред. С. Осуги, Ю. Саэки; — М. : Мир, 1990. — 304 с.

[13] Тарасов В. Б. О системно-организационном подходе в искусственном интеллекте / В. Б. Тарасов // VI Международная конференция «Знания-диалог-решения»: сб. научн. тр. — Ялта, 1997. — с. 57–70.

[14] Тельнов Ю. Ф. Интеллектуальные информационные системы в экономике / Ю. Ф. Тельнов. — М.: Московский государственный университет экономики, статистики и информатики, 1998. — 174 с.

[15] Allen J. AI Growing up / J. Allen // AI MAGAZINE. — 1998. — V. 19. — №4. — P. 13–23.

[16] Russell S. L. Artificial intelligence: a modern approach / S. L. Russell, P. Norvig. — Upper Saddle River, New Jersey: Prentice — Hall Inc., 1995. — 905 p.

Глава 2

ЗАДАЧИ И МЕТОДЫ ИХ РЕШЕНИЯ

2.1 Задачи систем искусственного интеллекта

Задачи систем искусственного интеллекта охватывают самые разные предметные области, среди которых лидируют бизнес, производство, медицина, проектирование и системы управления. Все задачи можно классифицировать по следующим общим основаниям [1–3, 7–9].

1) Задачи анализа и синтеза. В задаче анализа задана модель сущности и требуется определить неизвестные характеристики модели. В задаче синтеза задаются условия, которым должны удовлетворять характеристики «неизвестной» модели сущности, и требуется построить модель этой сущности.
2) Статические и динамические. В статических задачах явно не учитывают фактор времени и/или не изменяют знания об окружающем мире в процессе своих решений.
3) Использование общих утверждений для представления знаний, не содержащих явных ссылок на конкретные сущности, которые необходимо определить.
4) Использование частных ссылок, содержащие ссылки на конкретные сущности (объекты).

По типу решаемой задачи различают следующие задачи:

- интерпретация: процесс определения смысла данных (построение описаний по наблюдаемым данным);
- диагностика: процесс соотнесения объекта с некоторым классом объектов и/или обнаружение неисправностей в системе (отклонение параметров системы от нормативных в технике и в живых организмах);
- мониторинг: непрерывная интерпретация данных в реальном масштабе времени и сигнализация о выходе тех или иных параметров за допустимые пределы;

- прогнозирование: построение планов действий объектов, будущих событий на основе моделей прошлого и настоящего. В прогнозирующих системах часто используют динамические модели, в которых значения параметров «подгоняются» под заданную ситуацию. Выводимые из этих моделей следствия составляют основу для прогнозов с вероятностными оценками;
- планирование: конструирование плана действий объектов способных выполнять некоторые функции, т. е. программы действий. Оно основано на моделях поведения реальных объектов, которые позволяют проводить логический вывод последствий планируемой деятельности;
- проектирование: разработка ранее не существовавшего объекта и подготовка спецификаций на создание объектов с заранее определенными свойствами. Степень новизны может быть разной и определяется видом знаний и методами их обработки;
- обучение: диагностика, интерпретация, планирование, проектирование. Системы обучения выполняют такие функции, как диагностика ошибок, подсказывание правильных решений, аккумулирование знаний о гипотетическом «ученике» и его характерных ошибках, диагностирование слабости в познаниях обучаемых и нахождение соответствующих средств для их ликвидации. Системы обучения способны планировать акт общения с учеником;
- управление: интерпретация, прогноз, планирование, моделирование, оптимизация выработанных решений, мониторинг, т. е. функция системы, поддерживающая определенный режим ее функционирования или управления поведением сложной системы в соответствии с заданными спецификациями;
- отладка, ремонт: выработка рекомендаций по устранению неисправностей;
- поддержка принятия решений — совокупность процедур, обеспечивающих ЛПР необходимой информацией и рекомендациями для процесса принятия решений (выбор и/или, генерация альтернатив).

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

Термин «решение задач» (*problem solving*) употребляется в искусственном интеллекте в ограниченном смысле. Речь идет о хорошо определенных задачах, решаемых на основе поисковых алгоритмов.

Задача считается хорошо определенной, если для нее имеется возможность задать пространство возможных решений (состояний), а также способ просмотра этого пространства с целью поиска конечного (целевого) состояния, соответствующего решаемой задаче. Поиск конечного состояния задачи заключается в применении к каждому состоянию алгоритмической процедуры с целью проверки, не является ли это состояние решением задачи. Данная процедура продолжается до тех пор, пока не будет найдено решение.

Примерами хорошо определенных задач являются: доказательство теорем, поиск маршрута на графе, планирование робота в среде с препятствиями и т. д.

Человек обычно не решает задачу в той форме, в которой она изначально формулируется. Он стремится представить задачу таким образом, чтобы ему удобно было ее решать. Для этого он выполняет преобразование исходного представления задачи с целью сокращения пространства, в котором необходимо выполнять поиск решения задачи. Этап выбора подходящей формы представления задачи настолько обыден, что мы часто не осознаем его важности. В то же время форма или способ представления задачи в значительной мере определяет успех ее решения. При выборе способа представления задачи обычно учитывают два обстоятельства: представление задачи должно достаточно точно моделировать реальность; способы представления должны быть такими, чтобы решателю задач было удобно с ним работать.

Поскольку под решателем задач в искусственном интеллекте понимается компьютер, то мы и рассмотрим способы представления задач, удобные для их решения на ЭВМ. К ним относятся следующие наиболее часто используемые способы (формы) [4–6]:

- представление задач в пространстве состояний;
- представление, сводящее задачу к подзадачам;
- представление задач в виде теорем.

Данные представления и соответствующие универсальные методы поиска решений разрабатывались преимущественно на начальных этапах развития искусственного интеллекта. Позже было замечено, что для решения многих практических задач одних универсальных стратегий недостаточно. Необходим также большой объем знаний и наличие практического опыта. Исследование в области ИИ сосредоточились на представлении и приобретении знаний. Однако это не снизило значимости разработанных стратегий поиска решений, так как они представляют некоторые общие схемы управления механизмом вывода систем, основанных на знаниях. Рассматриваемые ниже способы представления задач и методы поиска их решений играют важную роль во многих системах ИИ, включая экспертные системы, понимание естественного языка, доказательство теорем и обучение.

2.2 Общие способы решения задач

Процесс решения задачи, как правило, включает два этапа: представление задачи и поиск (перебор). Успех решения задачи в значительной мере определяется формой ее представления. Формы представления задачи могут быть различными и зависят как от природы самой задачи, так и от ее решателя.

Пример

Например, при вычислении интеграла человек стремится путем преобразований представить его так, чтобы воспользоваться его табличными интегралами.

Выбирая кратчайший маршрут движения, человек может воспользоваться схемным представлением возможных путей перемещения и оценками расстояний между промежуточными пунктами.

Характер человеческого мышления таков, что этап и форма представления задачи, которую он использует, не всегда им осознаются. Так, шахматист не может четко объяснить форму представления шахматной ситуации, позволяющую ему не перебирать все возможные варианты продолжения при поиске очередного хода. Поэтому этап представления задачи часто выпадает из поля зрения человека. Тем не менее важность этого этапа осознается сразу же при попытке построить программно реализуемый алгоритм решения задачи.

Поиск формы представления задачи, удобный для ее машинного решения, является трудно формализуемым творческим процессом. Можно выделить следующие употребительные формы: представление в пространстве состояний, представление путем сведения задачи к подзадачам, представление в виде теоремы, комбинированное представление.

Полное представление задачи в пространстве состояний включает описание всех состояний или только начальных, задание операторов, отображающих одни состояния в другие, и задание целевого состояния.

Возможны различные формы описания состояний задачи. В частности, могут быть использованы строки, векторы, матрицы и графы. Выбирая ту или иную форму описания состояний, следует позаботиться о том, чтобы применение оператора, преобразующего одно состояние в другое, оказалось достаточно простым.

Операторы могут быть заданы с помощью таблицы, связывающей каждое входное состояние с некоторым выходным. Для больших задач такое задание оператора практически затруднительно. При описании состояния в форме строки (вектора) удобно задать оператор в виде правила переписывания.

Процедура поиска решения в пространстве состояний состоит в том, чтобы найти последовательность операторов, которая преобразует начальное состояние в целевое. Решением задачи будет указанная последовательность операторов.

Деревом называется ориентированный граф, в каждую вершину которого входит только одна дуга, за исключением одной вершины, называемой корнем дерева.

Таким образом, в дереве каждая вершина, за исключением корня, является концом ровно одной дуги и началом одной или нескольких дуг.

Вершины V_i порождаются вершиной V. V — родительская вершина, а V_i — дочерние вершины.

Примем, что корень находится на 0 уровне. Вершины, порожденные корнем, — 1 уровень и т. д.

Существуют два типа структур взаимосвязи подзадач: И-структуры и И-ИЛИ-структуры. В структурах типа И для решения основной задачи требуется решить все подзадачи. В структурах И-ИЛИ подзадачи разбиваются на группы, внутри которых они связаны отношением И, а между группами — отношением ИЛИ.

В этом случае для решения исходной задачи достаточно решить все подзадачи только какой-либо одной группы.

Для описания представления сведения задач к подзадачам можно использовать граф, называемый графом редукции задачи (рис. 2.1). При этом вершинам будут соответствовать задачи, а дугам — операторы редукции задач. Корню соответствует

исходная задача, вершинам 1-го уровня — задачи, порожденные исходной задачей.

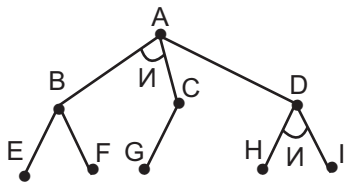


Рис. 2.1 – Дерево редукции задачи.

Задача A может быть решена, если будут решены задачи B и C или задача D. Задача B будет решена, если будут решены задачи E или F. Задача C — если решена G. Задача D — если решены задачи H и I.

Для указания связности вершин используется специальная кривая. Если имеются связанные вершины (дуги), то обычно, вводя при необходимости дополнительные вершины, дерево редукции задачи преобразуют так, чтобы каждая группа связанных вершин имела отдельную родительскую вершину (рис. 2.2).

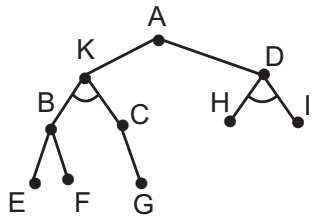


Рис. 2.2 – Преобразованное дерево редукции.

Будем рассматривать только такие деревья редукции.

Пример

Задача о выборе маршрута, или задача о коммивояжере.

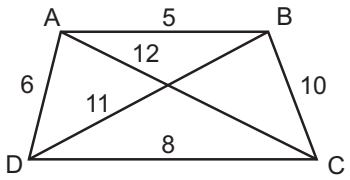


Рис. 2.3

Транспортный робот должен построить маршрут так, чтобы побывать в каждом из n заданных пунктов в точности по разу и возвратиться в исходный пункт. При этом если таких маршрутов несколько, желательно выбрать такой, который имеет минимальную протяженность.

Состояния в этой задаче можно задавать строкой, обозначающей список пунктов, пройденных к текущему моменту.

Операторы — это отображения, соответствующие решению робота направиться из данного пункта в следующий. Целевым состоянием, если не требуется минимальная протяженность маршрута, является любое состояние, описание которого начинается и кончается *A* и содержит названия всех других пунктов. Например, состояние *ABCDA*.

. . . . . .

Представление задачи, сводящее задачу к подзадачам, предусматривает разбиение исходной задачи на множество подзадач, раздельное решение которых дает решение исходной задачи. Каждая из подзадач может, в свою очередь, быть также разбита на подзадачи. Число уровней разбиения теоретически не ограничено. На практике разбиение продолжается до получения на нижнем уровне множества задач (подзадач), способ решения которых известен. Такие задачи условимся называть элементарными.

Введем понятия теории графов. Графом (или геометрическим графом) называется геометрическая конфигурация, состоящая из множества *V* точек, взаимосвязанных с множеством *E* непрерывных, самонепересекающихся кривых. Точки из множества *V* называются вершинами, кривые из множества *E* — ребрами. Если на всех ребрах задано направление, то граф называют ориентированным графом, а его ребра — дугами.

Если заданы два графа $G1$ и $G2$, причем множества вершин и кривых графа $G2$ являются подмножествами множеств вершин и кривых $G1$, то $G2$ называют подграфом графа $G1$, $G1$ — надграфом графа $G2$.

Структурно граф (дерево) редукции задачи, отличается от графа (дерева) состояния тем, что в нем имеются связанные дуги. Связанные вершины называют И-вершинами, несвязанные — ИЛИ-вершинами, а граф называется графом типа И-ИЛИ.

Вершины, соответствующие элементарным задачам называются заключительными.

Вершины, не имеющие дочерних вершин и не являющиеся заключительными, называются тупиковыми. Тупиковым вершинам соответствуют задачи, которые в рамках данного представления неразрешимы.

Таким образом, заключительные вершины являются разрешимыми, а тупиковые — неразрешимыми.

Вершина, не являющаяся ни заключительной, ни тупиковой, будет разрешимой тогда и только тогда, когда все ее дочерние вершины разрешимы, если они являются связанными, или хотя бы одна из дочерних вершин разрешима, если они являются несвязанными.

Очевидно, задача *A* является разрешимой в том и только в том случае, если вершины *H* и *I* являются заключительными или заключительными являются вершины *G* и *F* или *G* и *E*.

Граф редукции задачи может быть задан в явном виде (на рис. 2.2). Но чаще он, как и граф состояния, задается в неявном виде посредством описания исходной задачи и операторов редукции.

2.3 Методы решения задач

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

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

Требования пользователя к результату задачи, решаемой с помощью поиска, можно характеризовать:

1) количеством решений: одно решение, несколько решений, все решения;
2) свойствами результата: ограничения, которым должен удовлетворять полученный результат;
3) и (или) способом его получения.

Существующие методы решения задач, используемые в интеллектуальных системах, можно классифицировать следующим образом [1, 7]:

1) методы поиска в одном пространстве — методы, предназначенные для использования в следующих условиях: области небольшой размерности, полнота модели, точные и полные данные;
2) методы поиска в иерархических пространствах — методы, предназначенные для работы в областях большой размерности;
3) методы поиска при неточных и неполных данных;
4) методы поиска, использующие несколько моделей, предназначенные для работы с областями, для адекватного описания которых одной модели недостаточно.

Предполагается, что перечисленные методы при необходимости должны объединяться для того, чтобы позволить решать задачи, сложность которых возрастает одновременно по нескольким параметрам.

2.3.1 Поиск решений в одном пространстве

Методы поиска решений в одном пространстве обычно делятся:

1) на поиск в пространстве состояний (рассмотрим подробно);
2) поиск методом редукции;
3) эвристический поиск;
4) поиск методом «генерация-проверка».

Поиск в пространстве состояний

В большинстве СИИ информация, доступная стратегии управления, явно недостаточна для того, чтобы выбрать подходящее правило на каждом шаге процесса, поэтому сами процедуры вывода являются, как правило, поисковыми процедурами. Поскольку понятиям знания о задаче могут соответствовать состояния задачи, а правилам вывода — операторы перехода из одного состояния в другое, то процедура поиска решения называется поиском в пространстве состояний.

Поиск решений в пространстве состояний сводится к определению последовательности операторов, отображающих начальные состояния в целевые. Причем если такая последовательность не одна и задан критерий оптимальности, то поиск сводится к нахождению оптимальной последовательности операторов, обеспечивающих оптимум заданного критерия оптимальности.

Методы поиска решений в пространстве состояний удобно рассмотреть, используя дерево (граф) состояний. На дереве состояний поиск решения сводится к определению пути (оптимального, если задан критерий оптимальности) от корня дерева к целевой вершине, т. е. к вершине, соответствующей целевому состоянию. Поиск решения можно наглядно проиллюстрировать на дереве состояний, когда начальное состояние одно. Поэтому сначала рассмотрим задачи, определяемые тройкой (S_0, F, G), в которой множество S_0 начальных состояний состоит из одного элемента, F — множество операторов, G — множество целевых состояний.

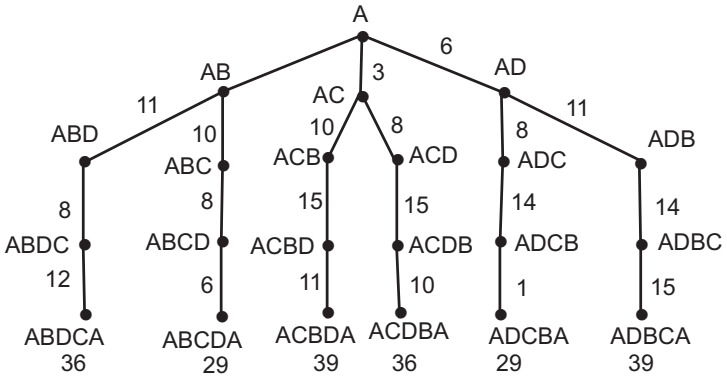


Рис. 2.4 – Граф состояния задачи.

Для построения дерева состояния следует, используя операторы из F, применимые к корню дерева (начальное состояние), построить вершины 1-го уровня. Затем, используя операторы из F, применимые к вершинам 1-го уровня, построить вершины 2-го уровня и т. д. Процесс применения операторов к какой-либо вершине дерева для построения всех ее дочерних вершин называется раскрытием вершин. Поэтому операторы преобразования мира (операторы из F) интерпретируются как правила раскрытия вершин. Применение правил раскрытия к начальной вершине порождает совокупность дочерних вершин. Каждая из них соответствует некоторому состоянию мира, в которое он может перейти из начального состояния. Дуги, связывающие начальную вершину с дочерней, идентифицируют как соответствующие операторы преобразования. Для всех дочерних вершин производится проверка, не являются ли они целевыми, т. е. не соответствуют ли целевым состояниям. Если целевая вершина не обнаружена, то выполняется следующий

этап порождения вершин путем применения правил раскрытия к каждой вершине, порожденной на предыдущем этапе, и т. д. Процедура продолжается до обнаружения целевой вершины.

Рассмотрим процедуру построения графа состояний на примере выбора маршрута транспортным роботом, который должен, выйдя из склада (пункт *A*), обойти обрабатывающие центры (пункты *B*, *C*, *D*) и вернуться в склад. Запрещается, чтобы робот побывал в каком-либо из обрабатывающих центров более одного раза. Схема маршрутов и расстояние между пунктами представим на рис. 2.4.

Условимся состояние в этой задаче обозначать словом (набором букв), образованным из букв, соответствующих наименованию пунктов, пройденных к текущему моменту и расположенных в том порядке, в каком были пройдены соответствующие пункты. Тогда очевидно, что начальным будет состояние *A*, а целевыми будут состояния, которые начинаются и оканчиваются словами *A* и содержат остальные буквы по одному разу.

Операторы (или правила раскрытия) в данном примере соответствуют выбору того или иного маршрута.

На графе состояний этой задачи начальной вершине (корню дерева) соответствует состояние *A*. Корень дерева порождает три дочерние вершины, соответствующие состояниям *AB*, *AC*, *AD*. Каждая из вершин, порожденных корнем, порождает по две вершины и каждая из вершин 2 и 3 уровней — по одной. На дугах указаны расстояния, которые проходит робот при переходе из одного состояния в другое.

Поиск решения имеет итеративный характер, причем число итераций и вершин, раскрытых до нахождения целевой вершины, существенно зависит от порядка (последовательности), в котором раскрывались вершины. Порядок раскрытия вершин принято называть стратегией поиска.

Можно выделить два основных типа стратегий поиска: «слепой» перебор и упорядоченный перебор вершин-кандидатов на раскрытие. «Слепой» перебор характеризуется тем, что расположение целевых вершин или их близость не влияют на порядок раскрытия. Существует несколько алгоритмов «слепого» перебора. Рассмотрим три наиболее типичных: алгоритм полного перебора, алгоритм равных цен, алгоритм перебора в глубину [4–6].

Алгоритм полного перебора. Вершины раскрываются в том порядке, в котором они были порождены. Первой раскрывается начальная вершина. Проверяется, нет ли среди порожденных вершин целевой. Если есть, то поиск заканчивается. Если нет, то раскрывается первая из порожденных вершин и проверяется наличие целевых вершин. Затем раскрывается вторая из порожденных вершин и т. д.

Для структурированной записи алгоритмов поиска в пространстве состояний введем понятия списков «открытых» и «закрытых» вершин. Список «закрытых» вершин — это список, где размешаются идентификаторы «раскрытых» и идентификатор вершины, которую предстоит раскрыть, в данный момент. Вершины из списка «закрыт», кроме последней, раскрывать нельзя.

Список «открытых» вершин — это список, где размещаются вершины, которые могут и должны быть раскрыты. Стратегии поиска различаются правилами размещения вершин в списке «открыт» и выбора очередной вершины для раскрытия.

В структуризованном виде алгоритм полного перебора можно представить следующим образом:

1) поместить начальную вершину в список «открыт»;
2) если список «открыт» пуст, то подать сигнал о неудаче поиска, в ином случае перейти к следующему шагу;
3) взять первую вершину из списка «открыт» и перенести ее в список «закрыт»; присвоить вершине идентификатор *v*;
4) раскрыть вершину *v*. Поместить все дочерние неповторяющиеся (т. е. не встречающиеся в списке «закрыт») вершины в конец списка «открыт» и построить указатели, ведущие от них к вершине *v*. Если вершина *v* не имеет дочерних вершин или имеет только повторяющиеся дочерние вершины, то перейти к шагу 2;
5) проверить, не является ли одна из дочерних вершин *v* целевой. Если является, то выдать решение; в ином случае перейти к шагу 2.

В этом алгоритме предполагается, что начальная вершина (корень) не может быть целевой. Алгоритм, безусловно, позволяет найти оптимальное (минимальное по числу дуг) решение. Например, использование этого алгоритма для решения задачи о выборе маршрута (с минимальным расстоянием) по существу сводится к построению графа состояний. Имея этот граф и сравнивая расстояния различных маршрутов, ведущих к целевым состояниям, можно выбрать оптимальные маршруты. Как следует из графа состояний, таких маршрутов два — *ABCDA*, *ADCBA*.

Алгоритм перебора в глубину. Определим глубину вершины числом, равным номеру ее уровня. При методе перебора в глубину всегда вскрывается та из вершин, которая имеет наибольшую глубину. Так как несколько вершин могут иметь одинаковую наибольшую глубину, предполагается, что имеется правило выбора одной из них. Кроме того, обычно по тем или иным соображениям задается граничная глубина; вершины, имеющие глубину, равную граничной, не раскрываются. Таким образом, метод перебора в глубину можно определить как метод перебора, при котором всегда раскрывается та из вершин, которая имеет наибольшую глубину, меньшую граничной.

Рассмотрим алгоритм перебора в глубину в структуризованном виде:

1) поместить начальную вершину в список «открыт»;
2) если список «открыт» пуст, то выдается сообщение о неудаче, в ином случае перейти к шагу 3;
3) взять первую вершину из списка «открыт» и перенести в список «закрыт». Этой вершине присвоить идентификатор *v*;
4) если глубина вершины *v* равна граничной глубине, то перейти к 2, в ином случае к 5;
5) раскрыть вершину *v*. Поместить все дочерние вершины в начало списка «открыт» и построить указатели, идущие от них к вершине *v*. Если *v* не имеет дочерних вершин, то перейти к шагу 2;
6) если одна из этих вершин целевая, то на выход выдать решение, получаемое путем просмотра назад в соответствии с указателями, в ином случае перейти к шагу 2.

Приведем один из возможных графов решения задачи выбора маршрута с помощью алгоритма перебора в глубину. При применении этого алгоритма решение

зависит от стратегии упорядочения вершин в списке «открыт». В приведенном решении найден неоптимальный маршрут (рис. 2.5).

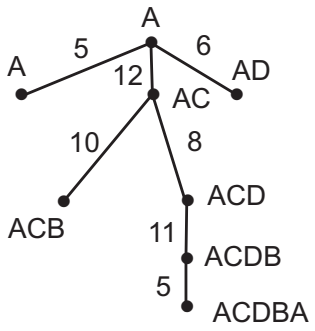


Рис. 2.5 – Граф решения задачи о выборе маршрута при использовании перебора в глубину.

Во всех рассмотренных алгоритмах перебора предполагается, что начальная вершина только одна. Если начальных вершин несколько, то эти алгоритмы изменяются на шаге 1: на шаге 1 в список «открыт» помещаются все начальные вершины.

Достоинством методов «слепого» перебора является, во-первых, простота алгоритмической реализации, во-вторых, обязательность получения решения, если оно существует. Недостатком этих методов является резкое возрастание числа вершин, которые необходимо раскрыть в процессе поиска решения, т. е. увеличение размерности задачи. Это существенно сужает круг практических задач, которые могут быть решены методами «слепого» перебора.

Алгоритм упорядоченного перебора. Для большинства практических задач удается сформулировать эмпирические правила, позволяющие уменьшить объем перебора. Эти правила используют специфическую информацию о решаемой задаче, формулируемую на основе опыта, интуиции и здравого смысла исследователя. Информацию такого рода часто называют *эвристической*, а основанные на ней алгоритмы — *эвристическими*.

Основная идея эвристических алгоритмов заключается в упорядочении списка открытых вершин в соответствии с некоторой мерой, оценивающей «перспективность» вершины или пути, на котором находится данная вершина. Такую меру называют *оценочной функцией*. Для очередного раскрытия из списка «открыт» выбирается вершина, имеющая минимальное значение оценочной функции. После каждого шага раскрытия производится переупорядочение вершин в списке в соответствии со значениями оценочной функции. Процедуру такого типа называют *алгоритмом упорядоченного перебора*. Существуют различные идеологии построения оценочных функций. Рассмотрим наиболее распространенную [4, 6, 7].

Пусть $g(v)$ — стоимость кратчайшего (оптимального) пути из любой начальной вершины $s_0 \in S_0$ до некоторой вершины v. Стоимость кратчайшего пути из вершины v до ближайшей целевой вершины обозначим $h(v)$.

Функция $f(v) = g(v) + h(v)$, очевидно, выражает стоимость кратчайшего (оптимального) пути из начальной вершины в целевую при условии, что он проходит через вершину v. Введем в рассмотрение следующие оценочные функции: $\hat{g}(v)$ — оценка стоимости кратчайшего пути из начальной вершины в вершину v,

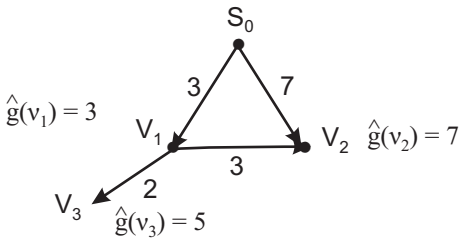
$\hat{h}(v)$ — оценка стоимости кратчайшего пути из вершины v до ближайшей целевой.

Тогда функция $\hat{f}(v) = \hat{g}(v) + \hat{h}(v)$ будет оценочной функцией функции $f(v)$, т. е. она является оценкой стоимости кратчайшего пути из начальной вершины в целевую при условии, что он проходит через вершину v.

В качестве оценочной функции $\hat{g}(v)$ естественно выбрать действительную стоимость пути от начальной вершины к вершине v найденного алгоритмом перебора к данному моменту. Заметим, что если граф состояний не имеет структуру дерева, то значение $\hat{g}(v)$ может меняться в процессе раскрытия вершин. Поясним это свойство на примере.

Пусть на шаге 1 поиска раскрыта вершина S_0 и порождены вершины v_1, и v_2 со значениями стоимостей $\hat{g}(v_1) = 3$, $\hat{g}(v_2) = 7$.

На шаге 2 раскрывается вершина v_1, и порождается вершина v_3 и снова вершина v_2, причем из v_1 в v_2 ведет путь стоимостью $g(v_1) = 3$. Очевидно, стоимость кратчайшего пути из S_0 в v_2 будет $g(v_2) = 6$.



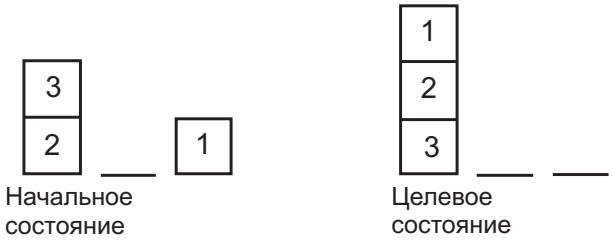
Необходимо заметить, что $\hat{g}(v) \geqslant g(v)$. Это видно и из примера. Эта оценочная функция легко вычисляется в процессе работы алгоритма.

Определение оценочной функции $\hat{h}(v)$ является более сложной задачей. При ее построении опираются на любую эвристическую информацию о решаемой задаче, поэтому $\hat{h}(v)$ называют эвристической функцией. Очевидно, если $\hat{h}(v) = 0$, то алгоритм перебора сводится к алгоритму равных цен, что соответствует полному отсутствию эвристической информации. Не существует каких-либо конструктивных рекомендаций к способам определения этой функции, поэтому рассмотрим ее смысл на примере. Конкретизируем задачу преобразования сцен (задачу о манипуляции предметами) следующим образом.

Пример

Пусть состояние, или, как принято говорить, мир задачи включает некоторое число площадок с расположенными на них кубиками (в реальной задаче вместо кубиков могут рассматриваться детали, из которых необходимо собрать изделие), а также робот манипулятор, перемещающий кубики с одной площадки на другую и устанавливающий их друг на друга. Текущее состояние такого мира можно интерпретировать как текущее расположение кубиков на площадках и положение манипулятора. Совокупность всех возможных состояний образует множество S. Очевидно, способы перемещения кубиков роботом, т. е. переход от одного состояния к другому, должны быть подчинены некоторым требованиям. Так, может быть ограничено число кубиков на конкретной площадке. Для перемещения естественно выбирать кубики, лежащие сверху, и т. д. На основе таких требований строится

множество F допустимых операторов преобразования мира, которые фактически являются совокупностью разрешенных действий робота. Желаемое расположение кубиков на площадках есть целевое состояние мира, решением задачи является последовательность действий робота (цепочка операторов $F_1, F_2, \ldots, F_i$), с помощью которой можно переставить кубики из некоторого начального расположения (начальное состояние мира) в желаемое, т. е. целевое состояние.



Три пронумерованных кубика располагаются на трех площадках. Манипуляционный робот может перемещать с одной площадки на другую по одному кубику. Кубики могут устанавливаться друг на друга. Для перемещения разрешается брать только верхний кубик. Заданы начальное и целевое расположение кубиков. Задача заключается в определении плана перемещения кубиков, позволяющего за минимальное число шагов (шаг — одно перемещение кубика) перейти от начального расположения к целевому.

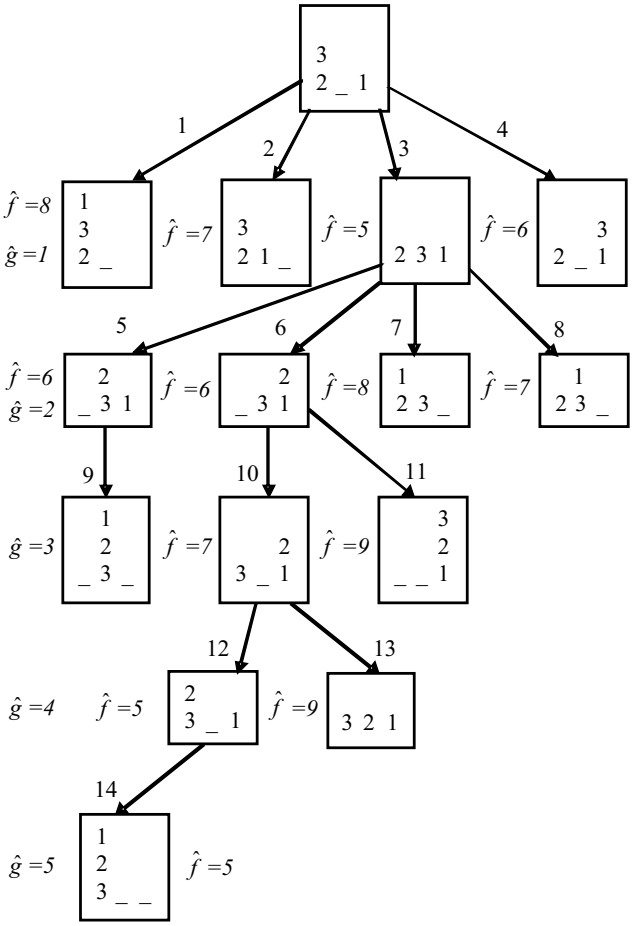


Рис. 2.6 – Граф решения задачи преобразования сцен.

В качестве эвристической функции $\hat{h}(v)$ примем сумму числа кубиков, расположенных не на своем месте, и общего числа кубиков, препятствующих установлению каждого из них на свое место. Препятствующими считаются кубики, расположенные сверху какого-либо кубика, который нужно переставить, и кубики, которые занимают «чужие» места в целевой конфигурации, т. е. места, в которые должны быть установлены кубики, расположенные не на своем месте.

Граф решения задачи с помощью алгоритма упорядоченного перебора при указанной эвристической функции приведен на рис. 2.6.

. . . . . .

Рассмотрим некоторые свойства эвристических алгоритмов упорядоченного перебора, использующих оценочную функцию $\hat{f}(v) = \hat{g}(v) + \hat{h}(v)$. Они являются гарантирующими, если для всех вершин v выполняется условие $\hat{h}(v) \leqslant h(v)$ и стоимости всех дуг превосходят некоторое минимальное число.

Для сравнения эффективности алгоритмов перебора используется понятие эвристической силы. Пусть A_1 и A_2 — произвольные гарантирующие алгоритмы перебора, а $f_1 = g(v) + \hat{h}_1(v)$ и $f_2 = g(v) + \hat{h}_2(v)$ — используемые ими оценочные функции. Алгоритм A_1 называют эвристически более сильным, чем алгоритм A_2, если на всех вершинах, кроме целевой, выполняется неравенство $\hat{h}_1(v) > \hat{h}_2(v)$. Эвристически более сильный алгоритм A_1 раскрывает меньшее число вершин при поиске минимального пути, чем алгоритм A_2.

Обозначим B множество гарантирующих алгоритмов, которые можно использовать для решения данной задачи. Алгоритм $A^* \in B$ называется оптимальным, если он не раскрывает большее число вершин, чем любой другой алгоритм $A \in B$.

Оптимальность в указанном смысле не является исчерпывающей характеристикой эффективности алгоритма, т. к. не учитывает сложность вычисления оценки эвристической функции $\hat{h}(v)$. В большинстве практических задач более объективной характеристикой является оценка объема вычислений, необходимых для определения значений $\hat{h}(v)$ на всех шагах к целевой вершине.

Поиск решений при сведении задач к подзадачам

Поиск при редукции задач. Поиск решения при данном типе представления задачи опирается на граф редукции задачи, который является графом типа И-ИЛИ. Неявно граф И-ИЛИ определяется посредством описания операторов порождения подзадач. Оператор преобразует исходное описание задачи во множество дочерних подзадач. Это преобразование должно быть таким, чтобы решение всех дочерних подзадач обеспечивало решение исходной задачи. Если множество дочерних задач состоит из одного элемента, то производится замена одной задачи другой, эквивалентной ей. Для конкретной задачи может существовать множество операторов преобразования. Применение каждого оператора порождает альтернативное множество подзадач, что определяет существование отношения ИЛИ на графе редукции.

Построение графа редукции задачи аналогично построению графа поиска решения в пространстве состояний. Цель поиска — показать, что начальная вершина разрешима. Процедуру поиска можно интерпретировать как построение дерева решения. Дерево решения — это поддерево (подграф) графа редукции задачи с корнем в начальной вершине, состоящее из разрешимых вершин.

Поиск на графе редукции задачи отличается от поиска на графе состояний тем, что он включает процедуры проверок разрешимости и неразрешимости вершин вместо процедуры проверок соответствия состояния целевому.

Вершина неразрешима, если она является тупиковой или если неразрешимы все ее дочерние вершины, когда они связаны отношением ИЛИ, или хотя бы одна из дочерних вершин, когда они связаны отношением И. Редукция (раскрытие вершин) заканчивается, если устанавливается разрешимость или неразрешимость начальной вершины.

Как и при поиске в пространстве состояний различают методы слепого и упорядоченного перебора на графе редукции [4, 7].

Алгоритм полного перебора. В методе полного перебора вершины раскрываются в том порядке, в котором они строятся. Алгоритм полного перебора при поиске решающего графа имеет специфику, обусловленную процедурами проверок, являются ли вершины разрешимыми или неразрешимыми. Он формируется следующим образом:

1) поместить начальную вершину S_0 в список ОТК;
2) взять первую вершину из списка ОТК и поместить ее в список ЗКР; обозначить эту вершину через v;
3) раскрыть вершину v и поместить все ее дочерние вершины в конец списка ОТК и провести от них указатели к вершине v. Если дочерних вершин не оказалось, то поместить вершину как неразрешимую и перейти к следующему шагу; в ином случае перейти к шагу 7;
4) применить к дереву поиска процедуру разметки неразрешимых вершин;
5) если начальная вершина помечена как неразрешимая, то выдать результат о неудаче; в ином случае перейти к следующему шагу;
6) изъять из списка ОТК все вершины, имеющие неразрешимые предшествующие им вершины, и перейти к шагу 2;
7) если дочерние вершины являются связанными, то перейти к следующему шагу; в ином случае перейти к шагу 9;
8) если все дочерние вершины являются заключительными, то поместить v как разрешимую и перейти к 10, в ином случае — как неразрешимую и идти к шагу 4;
9) если хотя бы одна дочерняя вершина является заключительной, то поместить как разрешимую и идти к 10, иначе — как неразрешимую и перейти к 4;
10) произвести разметку разрешимых вершин;
11) если начальная вершина помечена как разрешимая, то выдать дерево решения; в ином случае перейти к следующему шагу;
12) изъять из списка ОТК все вершины, являющиеся разрешимыми или имеющие разрешимые предшествующие им вершины, и перейти к шагу 2.

Рисунок 2.7 — это пример поискового дерева редукции задачи, на вершинах которого приведены цифры, указывающие последовательность раскрытия вершин при использовании алгоритма полного перебора. Заключительные вершины помечены буквой Z. Дерево решения выделено жирной линией. Вершины A и B не

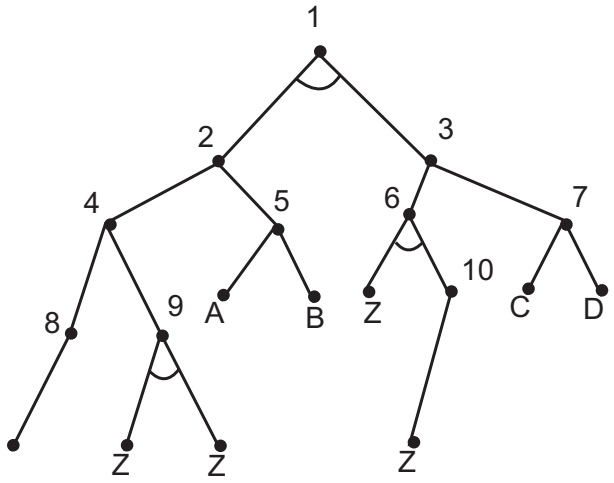


Рис. 2.7 – Дерево состояний алгоритма полного перебора.

раскрыты, т. к. являются тупиковым. Вершины *C* и *D* могут быть не тупиковыми и не заключительными, но они не раскрыты, т. к. по мере раскрытия вершины 10 найдено решающее дерево и алгоритм полного перебора прекращает работу.

Алгоритм перебора в глубину. При методе перебора в глубину ищется дерево решения в пределах заданной граничной глубины, и каждый раз раскрывается та из вершин с глубиной, строго меньшей граничной глубины, которая построена последней. Если решающее дерево содержит вершины с глубиной, превышающей граничную глубину, естественно, что решение не будет найдено. Алгоритм перебора в глубину включает такую последовательность шагов:

1) поместить начальную вершину S_0 в список ОТК;
2) взять первую вершину из списка ОТК и поместить ее в список ЗКР; обозначить эту вершину через *v*;
3) если глубина вершины *v* равна граничной глубине, то отметить вершину *v* как неразрешимую и перейти к шагу 5; в ином случае перейти к следующему шагу;
4) раскрыть вершину *v*. Поместить все дочерние вершины (в произвольном порядке) в начало списка ОТК и провести от них указатели к вершине *v*. Если дочерних вершин не оказалось, то пометить вершину *v* как неразрешимую и перейти к следующему шагу, в ином случае перейти к 8;
5) применить к дереву поиска процедуру разметки неразрешимых вершин;
6) если начальная вершина помечена, как неразрешимая, то выдать результат о неудаче; в ином случае перейти к следующему шагу;
7) изъять из списка ОТК все вершины, имеющие неразрешимые предшествующие им вершины. Перейти к шагу 2;
8) если все дочерние вершины являются связанными, то перейти к следующему шагу; в ином случае перейти к шагу 10;
9) если все дочерние вершины являются заключительными, то пометить *v* как разрешимую и перейти к 11, иначе — как неразрешимую и идти к 5;

10) если хотя бы одна дочерняя вершина является заключительной, то пометить v как разрешимую и идти к 11, иначе — как неразрешимую и перейти к 5;
11) применить к дереву перебора процедуру разметки разрешимых вершин;
12) если начальная вершина помечена как разрешимая, то выдается дерево решения; в ином случае перейти к следующему шагу;
13) изъять из списка ОТК все вершины, являющиеся разрешимыми или имеющие разрешимые предшествующие им вершины, и перейти к шагу 2.

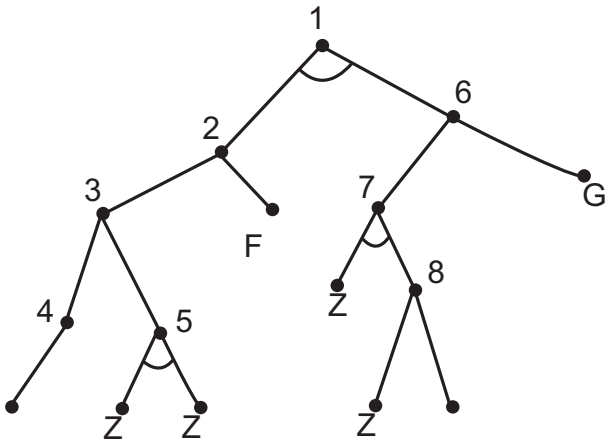


Рис. 2.8 – Дерево состояний алгоритма перебора в глубину.

На рис. 2.8 приведено поисковое дерево редукции задачи, которое строится при переборе в глубину с граничной глубиной, равной 4. Жирной линией выделено решающее дерево. Цифры на вершинах указывают на последовательность, в которой раскрываются вершины. Вершина F не раскрыта, хотя она не является ни тупиковой, ни заключительной и не достигла граничной глубины, т. к. до наступления момента ее раскрытия становится известным, что предшествующая ей вершина 2 разрешима и в соответствии с шагом 11 она изымается из списка ОТК. Вершина G не раскрыта, т. к. решающее дерево найдено.

При переходе на дерево И-ИЛИ возникает особенность при появлении повторяющихся вершин, т. е. вершин, эквивалентных (определяющих одну и ту же подзадачу) ранее построенным. В модифицированных методах полного перебора и перебора в глубину на дереве состояний повторяющиеся вершины вновь не строятся. При переборе на дереве типа И-ИЛИ этого делать нельзя (рис. 2.9).

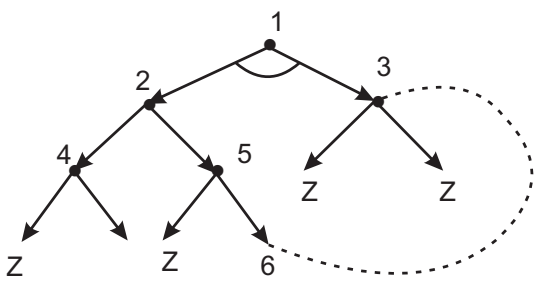


Рис. 2.9 – Пример поискового дерева типа И-ИЛИ.

Пример поискового дерева типа И-ИЛИ, построенного в процессе полного перебора. На этом дереве вершина 6 является повторяющейся — она эквивалентна вершине 3. Очевидно, повторяющуюся вершину 6 не строить нельзя, т. к. в ином случае не удалось бы получить решающее дерево. Если в процедуре предусмотрена процедура установления эквивалентности различных вершин, то в данном примере повторяющуюся вершину 6 раскрывать не обязательно, т. к. к моменту ее раскрытия известно, что эквивалентная ей вершина 3 разрешима.

Эвристические методы поиска (перебора). Для упорядочивания раскрываемых вершин на дереве типа И-ИЛИ используется так называемое оптимальное потенциальное дерево решения, для выделения которого применяется эвристическая функция. При поиске решения в пространстве состояний эвристическая функция определялась как оценка стоимости оптимального пути от заданной вершины до целевой. В деревьях типа И-ИЛИ для ее определения используется оценка стоимости дерева решения.

Стоимость дерева решения. Различают суммарную стоимость, представляющую собой сумму стоимости всех дуг в дереве решения и максимальную стоимость, равную стоимости пути между двумя вершинами дерева решения, имеющего максимальную стоимость. Стоимость пути определяется как сумма стоимостей дуг, входящих в этот путь. Эти определения можно пояснить на примере дерева решения (жирная линия), где цифры около дуг указывают на стоимости. В этом примере суммарная стоимость равна 20, максимальная стоимость — 15.

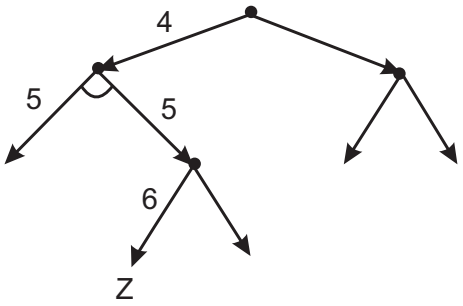


Рис. 2.10 – Дерево решения редукционного графа.

Если стоимость дуг равна единице, то суммарная стоимость равна числу дуг в дереве решения, а максимальная стоимость — числу дуг в пути между двумя самыми отдаленными вершинами дерева решения.

Дерево решения, имеющее минимальную стоимость (суммарную или максимальную в зависимости от того, какая из них принята за критерий оптимальности), называется оптимальным.

Эвристическая функция. Так называется функция h, являющаяся оценкой минимальной стоимости $\hat{h}(v)$. Рассмотрим, как может быть построена эвристическая функция. Дерево поиска, которое строится в ходе процесса перебора на каждом этапе задачи, пока не закончено построение дерева редукции, обладает вершинами, не имеющими построенных дочерних вершин. Такие вершины называются *концевыми*. Они могут быть заключительными, тупиковыми и нераскрытыми, т. е. вершинами, для которых на рассматриваемом этапе еще не построены дочерние вершины. Если концевая вершина v — заключительная вершина, то $h(v) = 0$; если

v — нераскрытая вершина, то $h(v)$ строится как оценка функции $\hat{h}(v)$ на основе эвристической информации, связанной с v исходной задачей.

Для не концевых вершин эвристическая функция строится так:

- если v имеет несвязанные дочерние вершины $v_1, \ldots, v_k$, то

$$\hat{h}(v) = \min_{i \in (l,\ldots,k)} \left[c(v, v_i) + \hat{h}(v_i) \right],$$

- если v имеет связанные дочерние вершины $v_1, \ldots, v_k$, то эвристические функции для суммарной и максимальной стоимостей определяются соответственно формулами:

$$\hat{h}(v) = \sum_{i=1}^{k} \left[c(v, v_i) + \hat{h}(v_i) \right],$$
$$\hat{h}(v) = \max_{i \in (l,\ldots,k)} \left[c(v, v_i) + \hat{h}(v_i) \right].$$

Для тупиковых вершин функция $\hat{h}(v)$ не определена.

Эвристический алгоритм упорядоченного перебора. В дереве поиска содержится множество поддеревьев с корнем в начальной точке, каждое из которых могло бы быть начальной частью дерева решения. Эти поддеревья называют потенциальными деревьями решения. Потенциальное дерево решений D_0 назовем оптимальным в том случае, когда оно обладает следующим свойством: если у вершины v, входящей в дерево D_0, в дереве перебора имеются связанные дочерние вершины, то все они входят в дерево D_0; если у этой вершины в дереве перебора имеются несвязанные дочерние вершины $v_i, (i = l, \ldots, k)$, то в дерево D_0 входит та из них, для которой значение $\left[c(v, v_1) + \hat{h}(v_1) \right]$ минимально.

При эвристическом методе упорядоченного перебора предполагается, что оптимальное потенциальное дерево решений является начальной частью оптимального дерева решения и его концевые нераскрытые вершины раскрываются в первую очередь. Для того чтобы в ходе перебора можно было извлечь из дерева поиска дерево D_0, необходимо после очередного раскрытия вершины вычислять эвристическую функцию $\hat{h}$ для каждой из вновь построенных вершин, предшествующих только что раскрытой вершине. Для остальных вершин значение $\hat{h}$ не меняется.

Эвристический алгоритм упорядоченного перебора можно сформулировать следующим образом.

1) Поместить начальную вершину в список ОТК и вычислить $\hat{h}(S_0)$.
2) Выделить оптимальное потенциальное дерево решений D_0.
3) Выбрать некоторую концевую вершину дерева D_0, которое имеется в списке ОТК, и поместить в список ЗКР. Обозначить эту вершину через v.
4) Если v — заключительная вершина, то пометить ее как разрешимую и перейти к следующему шагу. В ином случае перейти к шагу 8.
5) Применить к дереву D_0 процедуру разметки разрешимых вершин.
6) Если начальная вершина помечена как разрешимая, то выдать дерево D_0 в качестве дерева решения; иначе перейти к следующему шагу.

7) Изъять из списка ОТК все вершины, имеющие разрешимые предшествующие им вершины, и перейти к шагу 2.
8) Раскрыть вершину v. Если дочерних вершин она не имеет, то пометить ее как неразрешимую и перейти к следующему шагу; в ином случае перейти к шагу 12.
9) Применить к дереву D_0 процедуру разметки неразрешимых вершин.
10) Если начальная вершина помечена как неразрешимая, то выдать результат о неудаче; в ином случае перейти к следующему шагу.
11) Изъять из списка ОТК все вершины, имеющие неразрешимые предшествующие им вершины, и перейти к шагу 2.
12) Поместить все дочерние вершины в список ОТК и провести от них указатели назад к вершине S. Вычислить $\hat{h}$ для этих вершин. Пересчитать величины $\hat{h}$ для вершины V и для предшествующих ей вершин.
13) Перейти к шагу 2.

2.3.2 Поиск в иерархических и альтернативных пространствах

Методы поиска в одном пространстве не позволяют решать сложные задачи, так как с увеличением размера пространства время поиска экспоненциально растет. При большом размере пространства поиска можно попробовать разбить общее пространство на подпространства и осуществить поиск сначала в них. Пространство поиска представлено иерархией пространств.

Методы поиска решения в иерархических пространствах обычно делятся:

1) на поиск в факторизованном пространстве;
2) поиск в фиксированном множестве пространств;
3) поиск в изменяющемся пространстве множеств.

Поиск в факторизованном пространстве. Во многих приложениях требуется найти все решения. Например — постановка диагноза. Пространство называется факторизованным, если оно разбивается на непересекающиеся подпространства (классы) частичными (неполными) решениями. Причем по виду частичного решения можно определить, что оно не приведет к успеху, т. е. что все полные решения, образованные из него, не приведут к целевым решениям. Поиск в факторизованном пространстве осуществляется на основе метода «иерархическая генерация-проверка». Если пространство поиска удается факторизовать, то поиск даже в очень большом пространстве можно организовать эффективно.

Поиск в фиксированном множестве пространств. Применение метода факторизации пространства ограничено тем, что для ряда областей не удается по частичному решению сделать заключение о его непригодности. Например задачи планирования и конструирования. В этих случаях могут быть применены методы поиска, использующие идею абстрактного пространства. Абстракция должна подчеркнуть важные особенности рассматриваемой задачи, позволить разбить задачу на более простые подзадачи и определить последовательность подзадач (план решения), приводящую к решению основной задачи.

Поиск в изменяющемся множестве иерархических пространств. В ряде приложений не удается все решаемые задачи свести к фиксированному набору подзадач.

План решения задачи в данном случае должен иметь переменную структуру и не может быть сведен к фиксированному набору подзадач. Для решения подобных задач может быть использован метод нисходящего уточнения. Этот метод базируется на следующих предположениях:

1) возможно осуществить частичное упорядочение понятий области, приемлемое для всех решаемых задач;
2) решения, принимаемые на верхних уровнях, нет необходимости отменять на более нижних.

Поиск в альтернативных пространствах. Рассмотренные выше методы поиска исходят из молчаливой предпосылки, что знания о предметной области и данные о решаемой задаче являются точными и полными и для них справедливо следующее:

1) все утверждения, описывающие состояние, являются истинными;
2) применение оператора к некоторому состоянию формирует некоторое новое состояние, описание которого состоит только из истинных фактов.

Однако при решении любых практических задач и особенно при решении неформализованных задач распространена обратная ситуация. Эксперту приходится работать в условиях неполноты и неточности знаний (данных) и, как правило, в условиях дефицита времени. Когда эксперт решает задачу, он использует методы, отличающиеся от формальных математических рассуждений. В этом случае эксперт делает правдоподобные предположения, которые он не может доказать; тем самым вопрос об их истинности остается открытым. Все утверждения, полученные на основе этих правдоподобных предположений, также не могут быть доказаны.

Выводы

Итак, для того чтобы система могла делать умозаключения, основанные на здравом смысле, при работе с неполными (неточными) данными и знаниями, она должна быть способна делать предположения, а при получении новой информации, показывающей ошибочность предположений, отказываться как от сделанных предположений, так и от умозаключений, полученных на основе этих предположений.

Мнение системы о том, какие факты имеют место, изменяется в ходе рассуждения, т. е. можно говорить о ревизии мнений. Таким образом, даже если рассматривать проблемную область как статическую, неполнота (и неточность) знаний и данных влечет за собой рассмотрение этой области при различных (и даже противоположных) предположениях, что, в свою очередь, приводит к представлению области в виде альтернативных пространств, соответствующих различным, возможно, противоречивым и (или) взаимодополняющим предположениям и мнениям.

Все неудачи, возникшие при поиске в одном направлении, не запоминаются при переходе к поиску в другом направлении. Та же самая причина неудачи может заново обнаруживаться и на новом направлении.

Осуществлять возврат целесообразно не к состоянию, непосредственно предшествующему данному, а к тому состоянию, которое является причиной возникновения неудачи. В используемых нами терминах причиной неудач являются предположения, т. е. недоказуемые утверждения. Поэтому при обнаружении неудачи необходимо возвращаться в состояние, где это предположение было сделано, и испытывать другое предположение.

Этот метод поиска называют поиском, направляемым зависимостью.

Поиск с использованием нескольких моделей. Все методы поиска, рассмотренные до сих пор, использовали при представлении проблемной области какую-то одну модель, т. е. рассматривали область с какой-то одной точки зрения. При решении сложных задач в условиях ограниченных ресурсов использование нескольких моделей может значительно повысить мощность системы. Объединение в одной системе нескольких моделей дает возможность преодолеть следующие трудности.

1) Переход с одной модели на другую позволяет обходить тупики, возникающие при поиске в процессе распространения ограничений.
2) Использование нескольких моделей позволяет в ряде случаев уменьшить вероятность потери хорошего решения (следствие неполного поиска, вызванного ограниченностью ресурсов) за счет конструирования полного решения из ограниченного числа частичных кандидатов путем их расширения и комбинации.
3) Наличие нескольких моделей позволяет системе справляться с неточностью (ошибочностью) данных.

Контрольные вопросы и задания к главе 2

1) Опишите классификацию задач искусственного интеллекта по общим признакам.
2) Представьте классификацию задач искусственного интеллекта по типу решаемой задачи.
3) Перечислите способы представления задач.
4) Поясните, какие два этапа необходимы для процесса решения задач.
5) Расскажите о формах описания состояний и операторов при поиске решения задачи.
6) Какие два типа структур взаимосвязи задач вы знаете?
7) Приведите примеры деревьев и преобразованных деревьев редукции задачи.
8) Какие особенности предметной области и требования позволяют правильно выбрать метод решения задачи?
9) Перечислите методы решения задач в одном пространстве.
10) Опишите поиск решения в пространстве состояний и приведите пример.

11) Опишите алгоритмы полного перебора (в ширину), в глубину, упорядоченного перебора решения задач в пространстве состояний.
12) Приведите примеры решения задач по разным алгоритмам в пространстве состояний.
13) Опишите поиск решения задач при редукции задачи на подзадачи.
14) Чем отличается поиск решений на графе редукции задачи от поиска на графе состояний?
15) Сформулируйте алгоритмы поиска решений при редукции задачи.
16) Представьте поисковое дерево редукции задачи для выбранного вами примера.
17) Как определяется эвристическая функция при поиске решения в пространстве состояний?
18) Как определяется эвристическая функция при поиске решения при редукции задачи на подзадачи?
19) Определите суммарную и максимальную стоимость, т. е. стоимость дерева решения в И-ИЛИ графе.
20) Рассмотрите пример дерева решения редукционного графа и определите оптимальное дерево решения для начальной вершины.
21) Перечислите методы поиска решения задачи в иерархических пространствах.
22) Дайте характеристику факторизованного пространства.
23) Опишите поиск решения задачи в фиксированном множестве пространств.
24) На чем базируется поиск решений задачи в изменяющемся множестве иерархических пространств.
25) Опишите поиск решения задачи в альтернативных пространствах.
26) Проанализируйте преимущества поиска решения задачи с использованием нескольких моделей.

Литература к главе 2

[1] Алиев Р. А. Производственные системы с искусственным интеллектом / Р. А. Алиев, Н. М. Абдикеев, М. М. Шахназаров. — М. : Радио и связь, 1990. — 246 с.

[2] Вагин В. Н. Дедуктивный вывод на знаниях // Искусственный интеллект : справочник / под ред. Д. А. Поспелова. — М. : Радио и связь, 1990. — Кн. 2. : Модели и методы — С. 89–105.

[3] Назаров В. М. Техническая имитация интеллекта: учеб. пособие для вузов / В. М. Назаров, Д. П. Ким, И. М. Макрова. — М. : Высш. шк., 1998. — 144 с.

[4] Нильсон Н. Искусственный интеллект. Методы поиска решений: пер. с англ. / Н. Нильсон. — М. : Мир, 1973. — 270 с.

[5] Нильсон Н. Проблемы искусственного интеллекта: пер. с англ. / Н. Нильсон. — М. : Радио и связь, 1985. — 280 с.

[6] Нильсон Н. Принципы искусственного интеллекта: пер. с англ. / Н. Нильсон. — М. : Радио и связь, 1985. — 375 с.

[7] Попов Э. В. Алгоритмические основы интеллектуальных роботов и искусственный интеллект / Э. В. Попов, Г. Р. Фирдман. — М. : Наука, 1976. — 456 с.

[8] Тельнов Ю. Ф. Интеллектуальные информационные системы в экономике / Ю. Ф. Тельнов. — М. : Московский государственный университет экономики, статистики и информатики, 1998. — 174 с.

[9] Newell A., Siwon H. GPS: A Program that Simulates Human Thought / Ed. by Feigenbaum E. A. and Feldman J. // Computers and Thought. — №4: McGraw-Hill, 1963.

[10] Allen J. AI Growing up / J. Allen // AI MAGAZINE. — 1998. — V. 19. — №4. — P. 13–23.

[11] Russell S. L. Artificial intelligence: a modern approach / S. L. Russell, P. Norvig. — Upper Saddle River, New Jersey: Prentice — Hall Inc., 1995. — 905 p.

Глава 3

ОСНОВНЫЕ ВИДЫ ЛОГИЧЕСКИХ ВЫВОДОВ

Формальная логика — наука о законах выводного значения, т. е. знания, полученного из ранее установленных и проверенных истин, без обращения в каждом конкретном случае к опыту, а только в результате применения законов и правил мышления.

Формальная логика включает: традиционную логику; математическую логику.

Традиционная логика при получении новых (выводных) знаний использует следующие логические виды, методы.

Анализ — логический метод расчленения целого на отдельные элементы с рассмотрением каждого из них в отдельности.

Синтез — объединение всех данных, полученных в результате анализа. Синтез на простое суммирование результатов анализа. Его задача состоит в мысленном воспроизведении основных связей между элементами анализируемого целого.

Индукция — процесс движения мысли от частного к общему, от ряда факторов к закону. Индуктивный прием обычно используется в тех случаях, когда на основе частного факта можно сделать вывод, установить взаимосвязь между отдельными явлениями и каким-либо законом.

Дедукция — это процесс движения мысли от общего к единичному, от закона к отдельным его проявлениям.

Абстрагирование — способность отвлечься от всей совокупности факторов и сосредоточить внимание на каком-либо одном вопросе.

Аналогия (традукция) — прием, в котором из сходства двух явлений в одних условиях делается вывод о сходстве этих явлений в других условиях. В логике аналогия рассматривается как форма получения выводного знания, как

умозаключение, в котором на основании сходства предметов в одних признаках делается вывод о сходстве этих предметов в других признаках. Метод аналогии широко используется в моделировании, так как модель — аналог объекта, изучаемого посредством моделирования.

Сравнение — установление сходства или различия явлений, процессов и объектов в целом или в каких-либо признаках. Сравнение — метод, позволяющий обнаружить тенденции общего хода процесса развития, вскрыть изменения, происходящие в развитии явления.

Математическая логика возникла в результате применения к проблемам формальной логики строгих методов, сходных с теми, которые используются в математике. С помощью специального языка формул достигается адекватное описание логической структуры доказательства и осуществляется построение строгих логических теорий. Математическая логика базируется на логике высказываний (описание суждений) и ее расширение - логике предикатов (описание умозаключений).

3.1 Дедуктивный вывод и автоматическое доказательство теорем

К настоящему времени в области ИИ известно большое количество систем логического вывода. Все они отличаются друг от друга заложенными в них моделями знаний, видами логического вывода, способами реализации, методами логического вывода (рис. 3.1) [11].

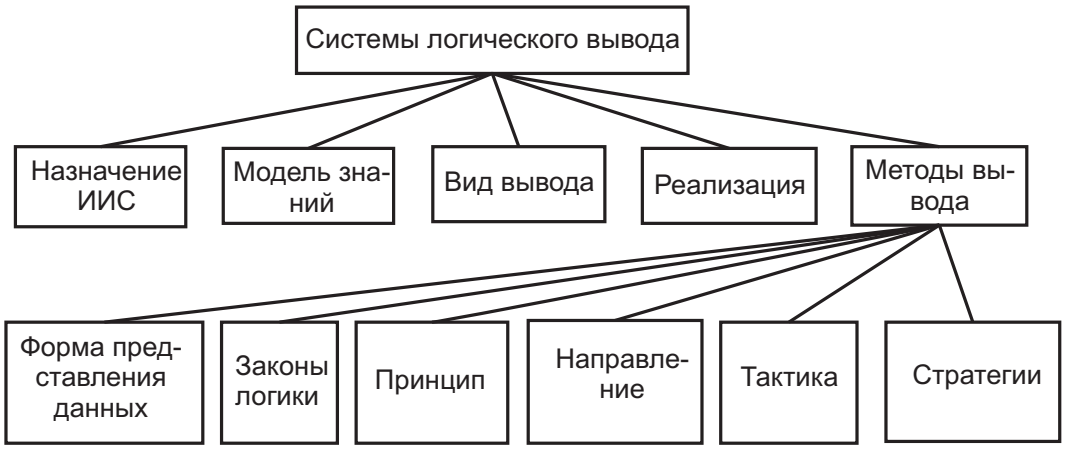


Рис. 3.1 – Характеристики систем логического вывода.

3.1.1 Рассуждения и принципы дедуктивного вывода

С возникновением интеллектуальных систем различного назначения и перенесением центра тяжести на модели и методы представления и обработки знаний существенно изменяется аппарат формальных рассуждений, комбинирующий средства достоверного и правдоподобного выводов. На наш взгляд, логика есть наука о рассуждениях и от разработки формальных моделей различных форм рассуждений зависит успех создания действительно интеллектуальных систем.

Неотъемлемой частью интеллектуальной системы является рассуждатель, интеллектуальной системы, состоящий из генератора гипотез, доказателя теорем и вычислителя [4].

Под рассуждением понимается построение последовательности аргументов, вынуждающих принятие некоторого утверждения, которое и является целью рассуждения.

Особенностями рассуждения, отличающими его от логического вывода, и в частности, от доказательства, в стандартном понимании являются:

- открытость множества возможных аргументов;
- использование метатеоретических средств, с помощью которых осуществляется управление логическими выводами, применяемыми в процессе рассуждения;
- использование правил не только достоверного вывода, но и правдоподобного вывода.

Очевидно, что логический вывод в стандартном понимании математической логики — частный случай рассуждений, когда множество аргументов фиксировано, нетривиальные средства (например, проверка на непротиворечивость) не используются и применяются только правила достоверного вывода, по которым из истинных аргументов (посылок) можно получить лишь истинные заключения.

В широком смысле к достоверному выводу и относится дедуктивный вывод, который в настоящее время хорошо изучен и исследован. В классической логике дедуктивный вывод рассматривается как вывод от общего к частному. Дедукция — в высшей степени идеализированная и ограниченная форма рассуждений, и если мы хотим моделировать некоторые аспекты человеческих рассуждений (здравый смысл, неопределенность, противоречивость информации и т. п.), то дедукции будет совершенно недостаточно, и нужно привлекать недедуктивные или правдоподобные формы рассуждений, такие как абдукция и индукция.

Дедукция (от deducere — выводить) — термин современной логики, обозначающий выведение одной мысли из другой, делаемое на основании логических законов. Большинство логиков под словом дедукция разумеют выведение частного из общего: такое ограничение, однако, не имеет основания. Дедукция получила значение термина лишь в новой логике, главным образом благодаря трудам английских мыслителей, рассматривающих дедукцию в противоположность индуктивному методу. Понятие дедукция встречается уже у Аристотеля. Латинская форма, deductio, впервые встречается в сочинениях Боэция; но как у Аристотеля, так и у Боэция дедукция не противополагается индукции, а обозначает собой понятие, тождественное с силлогизмом и с доказательством. В средневековой, схоластической логике слово дедукция не играет роли термина. В знаменитой пор-рояльской логике Арно («Logique ou l'art de penser») дедукция как термин тоже не встречается, нет его еще и в логике Канта.

Традиционной логикой называют формальную систему, предложенную Аристотелем более 2500 лет назад. Аристотель стремился установить и формально записать способы, с помощью которых можно было бы установить правильность

рассуждений в разумной полемике. Множества правил, определяющих, какие умозаключения могут быть получены из множества суждений, он назвал силлогизмом. В данном случае под суждением понимается законченная мысль, которую, можно выразить на естественном языке в одной из следующих четырех форм:

1) все X являются $P - (\forall X)P(X)$;
2) никакой X не является $P - (\forall X) \sim P(X)$;
3) некоторые из X являются $P - (\exists X)P(X)$;
4) некоторые из X не являются $P - (\exists X) \sim P(X)$.

Каждое правило силлогизма определяет переход от предпосылок к заключению, являющемуся интуитивно очевидным. Силлогизм изначально предназначен для управления разумной дискуссией на естественном языке. Как формальная теория, он чрезмерно сложен и неполон. Силлогизм можно назвать логикой классов, которая не содержит понятия дополнения класса. Например, определив класс A (допустим, субъекты, являющиеся водителями автомобилей), мы не можем построить его дополнение (т. е. установить всех лиц, которые не являются водителями).

Будучи тесно связанным с естественным языком, силлогизм иногда приводит к абсурдным результатам, в частности, не допуская логическую эквивалентность двух способов выражения одной и той же мысли. Силлогизм неполон в том смысле, что он не позволяет осуществлять логические выводы, в которых затрагиваются вопросы существования элемента некоторого класса.

Последователи Аристотеля, основываясь на силлогизме, сформулировали принципы дедуктивного вывода для высказываний, которые находятся на более высоком уровне абстракции по сравнению с суждениями. Наиболее известными из этих правил являются следующие.

Modus Ponendo Ponens: «Если истинна импликация $A \to B$ и A истинно, то B истинно».

Modus Tollendo Tollens: «Если истинна импликация $A \to B$ и B ложно, то A ложно».

Modus Ponendo Tollens: «Если A истинно и конъюнкция $A \wedge B$ имеет результатом ложь, то B ложно».

Modus Tollendo Ponens: «Если A ложно и дизъюнкция $A \vee B$ истинна, то B является истиной».

1) $\dfrac{A \to B, A}{B}$, 2) $\dfrac{A \to B, \sim B}{\sim A}$, 3) $\dfrac{\sim (A \wedge B), A}{\sim B}$, 4) $\dfrac{A \vee B, \sim A}{B}$.

На основе этих правил сформулировано правило «цепного заключения», весьма удобное для вывода в системе исчисления высказываний.

Цепное заключение: «Если истинна импликация $A \to B$ и истинна импликация $B \to C$, то импликация $A \to C$ является истинной».

Пример

Рассмотрим пример вывода с применением этих правил. Пусть заданы следующие посылки.

1) $P \to Q$. Если растут мировые цены на топливно-энергетические ресурсы, то увеличиваются поступления в бюджет.
2) $(R \vee Q) \to (R \vee S)$. Если наблюдается рост производства или увеличиваются поступления в бюджет, то следует увеличение производства или укрепление рубля.
3) $(P \to Q) \to \big((P \vee Q) \to (R \vee Q)\big)$. Если растут мировые цены на топливно-энергетические ресурсы, то увеличиваются поступления в бюджет, из чего следует, что при росте цен на сырье или при увеличении поступлений в бюджет происходит рост производства или увеличение бюджета.

Используя Modus Ponendo Ponens, из посылок 1 и 3 можно вывести следующее заключение.

4) $(P \vee Q) \to (R \vee Q)$. Если растут мировые цены на топливно-энергетические ресурсы или увеличиваются поступления в бюджет, то происходит рост производства или увеличение бюджета.

Из посылок 4 и 2 с помощью цепного заключения можно получить посылку 5.

5) $(P \vee Q) \to (R \vee S)$. Если растут мировые цены на топливно-энергетические ресурсы или увеличиваются поступления в бюджет, то происходит рост производства или укрепление рубля.

Выводы

Таким образом, применяя правила логического вывода, получаем новые логические формулы на основании исходных. При значительном количестве исходных данных возможно получение большого количества цепочек вывода, результаты которых могут противоречить друг другу. Подобные проблемы должны быть корректно решены в конкретных интеллектуальных системах при организации управления стратегией логического вывода.

Проблема доказательства в логике состоит в нахождении истинностного значения заключения B, если предполагается истинность исходных посылок $A_1, A_2, \ldots, A_n$, что записывается в виде: $A_1, A_2, \ldots, A_n \vdash B$. Знак $\vdash$ в доказательствах и выводах следует читать «верно, что» или «можно вывести».

Следовательно, термин *дедукция* составляет принадлежность логики XIX века. Еще и в настоящее время в некоторых сочинениях по логике дедукцию отождествляют с силлогизмом и считают его единственным правомерным способом умозаключения. Отождествлению дедукции с силлогизмом мешает, однако, то обстоятельство, что силлогизм есть лишь форма дедукции, а не самый процесс. От узкого значения термина дедукции следует отличать более широкое: совокупность процессов научного мышления (разделение и определение понятий, доказательство положения) за вычетом индукции. Понимаемая в таком смысле, дедукция оказывается процессом, прямо противоположным индукции; эту противоположность видят как в исходных точках, так и в способах перехода от одной мысли к другой и, наконец, в конечных целях. Такое воззрение защищал в русской литературе М. П. Владиславлев, стоявший в данном случае под влиянием Милля, Бэна и др. Нельзя отрицать различия между дедукцией и индукцией, но противоположение их не имеет никакого основания. Человеческое мышление одно; как бы ни были разнообразны предметы, направление и цель мышления, одни и те же законы управляют мыслью. Противополагать дедукцию во всем индуктивному мышлению — значит вносить дуализм в человеческое сознание. Различие дедукции от индукции получило характер противоположности вследствие развития опытных наук в новое время. Успехи опытного знания повлекли за собой подробное исследование методов его, причем иногда забывали, что в индукции имеется дело с тем же самым мышлением, в применении его к фактам внутреннего и внешнего мира. Неудачи спекулятивной философии, пользовавшейся по преимуществу дедукцией, способствовали расширению пропасти между индукцией и дедукцией. Между тем, легко заметить сродство индукции с дедукцией; не говоря о так называемой полной индукции (умозаключении от всех членов известной группы к самой группе), которая представляет собою пример совершенно правильного силлогизма, т. е. дедукция — и так называемая неполная индукция, т. е. заключение от частного к общему, имеет своим основанием закон тождества, ибо в неполной индукции от некоторых случаев мы заключаем ко всем на том основании, что рассматриваем эти некоторые случаи как типичные представители всей группы. Д. С. Милль свел индуктивные методы исследования к четырем основным: метод различия, согласия, остатков и сопутствующих изменений. Рассматривая их, легко убедиться, что они представляют собой не что иное, как различные способы умозаключения, основанные на законе тождества. Метод остатков, например, представляет собой чистый случай определения путем исключения, т. е. умозаключение разделительное. В превосходном труде Каринского «Классификация выводов» (СПб., 1880) есть множество доказательств тому, что противоположность индукции и дедукции в той форме, в которой оно обыкновенно делается, неосновательно и что поэтому нельзя делить все выводы на *индуктивные* и *дедуктивные*. С другой стороны, некоторые силлогистические выводы представляют собой пример умозаключений от частного к частному, на что впервые обратил внимание Дж. С. Милль.

Выводы

Таким образом, отождествляя дедукцию с силлогизмом, нельзя в то же время утверждать, что дедукция есть всегда умозаключение от общего к частному, а следует дать более общее определение.

Полное определение понятия дедукции требует, помимо указания отношения ее к индукции, еще рассмотрения отношения дедукции к анализу. Анализом называется прием мышления, посредством которого разлагается на составные элементы то, что в сознании дано как нечто целое; анализ противополагается синтезу; но и в дедукции выводится из известной мысли с необходимостью другая, которая была заключена implicite в первой; отсюда сходство дедукции и анализа очевидно. Если, однако, допустить, что форма дедукции — силлогизм, то придется сказать, что анализ — более общее понятие, чем дедукция. Всякая дедукция есть анализ, ибо разъясняет данное положение, выводя из него другое, заключенное в нем; но не всякий анализ есть дедукция. В состав каждого процесса дедукции входят следующие элементы: положение, из которого делается вывод и которое в таком случае называется основанием; самый процесс выведения из основания мысли, в нем заключенной, и, наконец, вывод или мысль, добытая из основания и поставленная как отдельное положение. Положения, из которых делаются выводы, могут быть чрезвычайно разнообразны, но, в конце концов, сводятся к двум родам: самоочевидные истины (аксиомы) и обобщения, добытые из опыта. Процесс выведения не меняет характера основания, из которого получается вывод, т. е. вывод из аксиомы сам получает аксиоматический характер; вывод из эмпирического положения есть факт, могущий быть проверенным на опыте. Самый процесс дедукции основан на законе тождества. Частное подводится под общее на том основании, что оно по содержанию тождественно с общим; то же самое положение можно заметить и в заключении от частного к частному.

Итак, дедукция — это совокупность процессов научного мышления (разделения (анализ) и определения понятий, доказательство положений), т. е. выведение столь же достоверного заключения из достоверных посылок с логической необходимостью на основе закона тождества, где частное подводится под общее.

3.1.2 Методы доказательства в логике

Существуют два основных метода решения проблемы доказательства в логике: семантический и синтаксический [2].

Семантический метод состоит в следующем. Можно перечислить все атомы, входящие в формулы $A_1, A_2, \ldots, A_n, B$, и составить таблицу истинности для

всевозможных комбинаций значений этих атомов. Затем следует осуществить просмотр полученной таблицы, чтобы проверить, во всех ли ее строках, где формулы $A_1, A_2, \ldots, A_n$ имеют значения «истина», формула B также имеет значение «истина». Этот метод применим всегда, но может оказаться слишком трудоемким.

При *синтаксическом* методе доказательства сначала записывают посылки и, применяя к ним правила вывода, стараются получить из них другие истинные формулы. Из этих формул и исходных посылок выводят последующие формулы, и этот процесс продолжают до тех пор, пока не будет получено требуемое заключение (заметим, что это не всегда возможно). Этот процесс, по сути дела, и является логическим выводом, он часто применяется при доказательстве теорем в математике.

Иногда значение конкретной логической формулы не зависит от значений входящих в них атомов. Правильно построенные логические формулы, значением которых будет «истина» при любых значениях входящих в них атомов, называются *тавтологиями*. Тавтологии, или теоремы логики, обладают следующим свойством: если вместо всех вхождений некоторого атома в тавтологию подставить произвольную логическую формулу, то снова будет получена истинная формула. Эта новая формула называется частным случаем исходной формулы, или результатом подстановки.

Правило подстановки. Если $C(A)$ — тавтология и вместо всех вхождений формулы A в C подставить формулу B, то $C(B)$ — тавтология. Для обозначения тавтологий используется символ $\models$, который читается «общезначимо» или «всегда истинно».

Примеры тавтологий:

$$\models A \wedge (A \to B) \to B;$$
$$\models \sim B \wedge (A \to B) \to \sim A;$$
$$\models (A \to B) \to \big((A \vee C) \to (B \vee C)\big).$$

Докажем то, что первая приведенная тавтология (Modus Ponendo Ponens) всегда будет иметь значение «истина», для чего построим сокращенную таблицу истинности.

Таблица 3.1 – Таблица истинности для доказательства тавтологии.

A	$A \to B$	$A \wedge (A \to B)$	$A \wedge (A \to B) \to B$
T	B	B	T
T	B	B	T
F	T	F	T
F	T	F	T

В приведенной таблице T — истина, F — ложь. Значение импликации совпадает со значением второго аргумента в том случае, если первый аргумент имеет значение T, поэтому в первых двух строках второго столбца присутствует второй аргумент — B, значение которого может быть произвольным. Конъюнкция $A \wedge (A \to B)$ при истинном A также будет иметь результат, совпадающий со значением B. Последний столбец таблицы комментариев не требует, так как приведенные в нем результаты очевидны.

Попробуем установить тавтологичность этой же формулы синтаксическим методом. Для этого раскроем все скобки по распределительному закону и убедимся, что результат упрощается до формул с очевидными значениями истинности:

$$\begin{aligned}\big(A \wedge (A \to B)\big) \to B &=\sim \big(A \wedge (\sim A \vee B)\big) \vee B = \big(\sim A \vee (A \wedge \sim B)\big) \vee B = \\ &= \big((\sim A \vee A) \wedge (\sim A \vee \sim B)\big) \vee B = \big(T \wedge (\sim A \vee \sim B)\big) \vee B = \\ &=\sim A \vee (\sim B \vee B) =\sim A \vee T = T\end{aligned}$$

Помимо использования тавтологий и подстановок полезным средством для вывода является эквивалентность. Необходимо уметь правильно осуществлять замену взаимно эквивалентных формул. Например, можно подставить $Q \vee P$ вместо $P \vee Q$, так как $Q \vee P \leftrightarrow P \vee Q$.

Эквивалентность $A \leftrightarrow B$ можно заменить двусторонней импликацией $(A \to B) \wedge$ $\wedge (B \to A)$, так как эти выражения имеют одну и ту же таблицу истинности. Отсюда можно сделать вывод, что логическая эквивалентность — это импликация в обоих направлениях. Эквивалентность можно привести в конъюнктивную нормальную форму, которая имеет вид $(\sim A \vee B) \wedge (\sim B \vee A)$. Если в этом выражении раскрыть скобки, получим дизъюнктивную нормальную форму: $(\sim A \wedge \sim B) \vee (A \wedge B)$. Другими словами, если A и B эквивалентны, то они либо оба истинны, либо оба ложны. Примеры тавтологий с эквивалентностями:

$$\begin{aligned}&\vDash (A \to B) \leftrightarrow (\sim B \to \sim A);\\ &\vDash (A \vee B) \leftrightarrow (B \vee A);\\ &\vDash A \wedge (A \vee B) \leftrightarrow A;\\ &\vDash \big(X \to (Y \to Z)\big) \leftrightarrow (X \wedge Y \to Z).\end{aligned}$$

В процедурах логического вывода эквивалентности можно использовать двумя способами:

1) расписывать в виде двух отдельных импликаций;
2) использовать при заменах.

При этом важно понимать отличие замены от подстановки, которое состоит в следующем: если $A \leftrightarrow B$, то A можно заменить на B в любом вхождении в формулу C, не меняя ее значения, причем замену не обязательно осуществлять во всех вхождениях.

В противовес тавтологиям в логике существуют противоречия формулы; значением которых всегда будет «ложь» независимо от значений входящих в них атомов. Примером является выражение $B \wedge \sim B = F$ при любом значении B.

Множество ППФ для некоторой предметной области называется теорией заданной области знаний, а каждая отдельная ППФ именуется аксиомой. Цель построения теории заключается в описании нужных знаний наиболее экономичным способом. Если удается построить теорию, которая адекватно описывает заданную область знаний, то все истинные факты из области интерпретации будут следствиями аксиом этой теории, другими словами, их можно будет вывести из множества ППФ. Соответственно ложные факты не будут следствиями теории, следовательно, их нельзя будет получить путем логического вывода на основании аксиом данной

теории. Теорию называют полной, если все истинные факты являются следствиями этой теории. Это означает, что каждая истинная для данной теории ППФ может быть доказана на основании ее аксиом.

Про теорию говорят, что она является синтаксически последовательной (непротиворечивой), если из аксиом теории невозможно вывести противоречие. Теория, в которой можно доказать и P, и $\sim P$, непоследовательна.

В качестве примера рассмотрим построение теории и ее проверку на полноту, и непротиворечивость для следующего фрагмента знаний:

«Если у Вас нет дома,
Пожары ему не страшны,
И жена не уйдет к другому,
Если у Вас нет жены».

Постараемся обойтись средствами логики высказываний и начнем с формирования области интерпретации. Для этого введем обозначения:

A — «У Вас есть дом»
B — «Дом может сгореть»
C — «У Вас есть жена»
D — «Жена может уйти к другому»

На базе этих четырех высказываний построим логические формулы:

$$\sim A \rightarrow \sim B;$$
$$\sim C \rightarrow \sim D;$$

Кроме логических формул, выражающих связь фактов, любая теория содержит истинные факты, на основе которых становится возможным конкретная интерпретация ППФ. Допустим, что в нашей теории такими фактами являются A, B, $\sim C$, $\sim D$.

Полученная теория является полной и последовательной, поскольку каждый факт этой теории выводится из остальных аксиом, при этом не возникнет противоречия. Доказать это несложно как семантическим, так и синтаксическим способом, однако для более сложных областей знаний необходимо использовать определенные стратегии доказательства, позволяющие преодолеть хаотичность процессов логического вывода.

Кратко рассмотрим три основные стратегии доказательства:

- доказательство с введением допущения;
- доказательство методом «от противного» (приведение к противоречию);
- доказательство методом резолюции.

Доказательство с введением допущения. Для доказательства импликации вида $A \rightarrow B$ допускается, что левая часть импликации истинна, т. е. A принимается в качестве дополнительной посылки, после чего делают попытки доказать правую часть B. Такая стратегия доказательства часто применяется в геометрии при доказательстве теорем.

Приведение к противоречию. Этот метод доказательства, предложенный К. Робинсоном в 1960-х гг., вывел исследования в области искусственного интеллекта на новый уровень. Метод обусловил появление обратных выводов и эффективных способов выявления противоречий. Суть его состоит в следующем. Например, требуется доказать $A \rightarrow B$. Вместо этого можно попытаться доказать $\sim B \rightarrow \sim A$, используя эквивалентность. Такое доказательство можно провести двояко, а именно: или

допустить A и доказать затем B (это будет прямой вывод), или сделать допущение о том, что B — ложно, после чего сделать попытку опровергнуть посылку A (обратный вывод). Приведение к противоречию — комбинация прямого и обратного вывода, т. е. для доказательства $A \to B$ можно одновременно допустить A и $\sim B$ (посылка истинна, а заключение — ложно):

$$\sim (A \to B) = \sim (\sim A \vee B) = A \wedge \sim B$$

В процессе доказательства можно двигаться по пути, который начинается от A или от $\sim B$. Если B выводимо из A, то, допустив истинность A, мы доказали бы B. Поэтому, сделав допущение $\sim B$, получим противоречие. Если вывод приведет к успеху (т. е. противоречие не будет получено), это будет свидетельствовать о несовместимости либо противоречивости исходных посылок. Мы также не получим противоречия, если доказываемое предложение $A \to B$ является ложным.

Доказательство методом резолюции. Этот метод считается более трудным для понимания, однако имеет важное преимущество перед другими: он легко формализуем. В основе метода лежит тавтология, получившая название «правило резолюции».

3.1.3 Представление и решение задач в виде теорем

При представлении задач в форме доказательства теорем возможные состояния задачи, включая начальное и целевые состояния, рассматриваются как правильно построенные формулы исчисления предикатов, которое будет рассмотрено позднее. Операторы, отображающие одно состояние в другое, рассматриваются как правила, которые выводят одно правильно построенное выражение из другого. Процесс решения задачи в этом случае протекает следующим образом:

- формулируется множество исходных истинных утверждений (аксиом) относительно условия задачи, записываемых в виде формул языка исчисления предикатов;
- строится гипотеза относительно результата решения задачи, которая также представляется на языке исчисления предикатов и называется целевым утверждением;
- проверяется, не содержит ли множество новых утверждений целевое утверждение или его отрицания; если содержит, то теорема считается доказанной и процесс решения задачи заканчивается; если целевое утверждение или его отрицание не входят во множество вновь сформулированных утверждений, то полученное множество утверждений объединятся с множеством исходных аксиом и процесс повторяется.

Этот метод полного перебора гарантирует нахождение решения, если он существует. Для управления поиском используются различные стратегии выбора пар аксиом или промежуточных утверждений, участвующих в выводе, что сокращает время решения и объем памяти ЭВМ.

Формулировка задачи дедуктивного вывода. Пусть задача описывается на языке исчисления предикатов. Если обозначить через Φ_0 множество исходных утверждений и через F_g целевое утверждение, то задача, представленная в виде

доказательства теоремы, формально может быть записана в виде [9, 13]:

$$\Phi_0 \Rightarrow F_g$$

Иными словами, необходимо доказать, что формула F_g логически следует из множества Φ_0.

Формула F_g логически следует из Φ_0, если каждая интерпретация, удовлетворяющая Φ_0, удовлетворяет F_g. Когда множество Φ_0 представлено формулами $F_1, F_2, \ldots, F_n$, то задачей дедуктивного вывода исчисления предикатов является выяснение общезначимости формулы:

$$F_1 \wedge F_2 \wedge \ldots \wedge F_n \rightarrow F_g$$

В ходе вывода часто используют метод доказательства от обратного, то есть устанавливают не общезначимость приведенной выше формулы, а невыполнимость формулы:

$$\sim (F_1 \wedge F_2 \wedge \ldots \wedge F_n \rightarrow F_g) \quad \text{или} \quad F_1 \wedge F_2 \wedge \ldots \wedge F_n \wedge \sim F_g$$

Иными словами, доказывают, что объединение $\Phi_0 U \sim F_g$ невыполнимо. Процесс установления невыполнимости некоторого множества формул называют опровержением.

Разработаны эффективные методы доказательства теорем (методы автоматизации дедуктивного вывода): процедура вывода Эрбрана, принцип резолюции, вывод на графе связей, вывод на графе дизъюнктов. Мы рассмотрим один подход (принцип резолюции), разработанный в 1965 году Дж. Робинсоном, позволяющий автоматизировать процесс опровержения и являющийся теоретической базой для построения многих систем автоматического доказательства теорем.

Стандартизация предикатных формул. Для определения невыполнимости и выводимости формулы ее удобно представить в виде дизъюнктов (предложений). Всякую формулу можно представить в виде дизъюнктов, применив к ней последовательность приведенных ниже простых операций.

1) Переименование переменных. Выполняется такая замена переменных, что все переменные, связанные кванторами, становятся различными. Например, $\forall x R(x) \vee \forall x S(x)$ переписывается в виде $\forall x R(x) \vee \forall y S(y)$.
2) Исключение знака импликации. Всякий раз, когда встречается $\rightarrow$, делается замена $(A \rightarrow B)$ на $\big((\sim A) \vee B\big)$.
3) Уменьшение области действия связки $\sim$. Везде, где возможно, делаются замены:

   Заменяется $\sim\sim A$ на A.

   Заменяется $\sim (A \vee B)$ на $\sim A \wedge \sim B$.

   Заменяется $\sim (A \wedge B)$ на $\sim A \vee \sim B$.

   Заменяется $\sim (\forall x A)$ на $\exists x (\sim A)$.

   Заменяется $\sim (\exists x A)$ на $\forall x (\sim A)$.

В конце концов получается формула, где связка встречается непосредственно перед атомной формулой.

1) Исключение кванторов существования. Вычеркиваются поочередно кванторы существования. При этом каждая переменная y, связанная квантором существования, заменяется на $g(x_1, \ldots, x_n)$, где g — символ новой (отличной от имеющихся в формуле) функции, а $x_1, \ldots, x_n$ — все переменные, встречающиеся в кванторах всеобщности, области действия которых содержат вычеркиваемый квантор существования. Если таких переменных нет, то y заменяется на новую константу. Функция g называется функцией Сколема.
2) В предваренной нормальной форме все кванторы общности переносятся влево в начало формулы, так что формула принимает вид $\forall x_1, \forall x_2 \ldots \forall x_n A$, где A не содержит кванторов.
3) Приведение к конъюнктивной нормальной форме. Приведение осуществляется заменой, пока это возможно, $(A \wedge B) \vee C$ на $(A \vee C) \wedge (B \vee C)$.

В результате применения шагов 1–6 получаем выражение $\forall x_1, \ldots, \forall x_n (A_1 \wedge \ldots \wedge \wedge A_n)$, или формулу, которая называется формой, где A_i, имеет вид $(l_1^i \vee \ldots \vee l_r^i)$, а l_j^i есть атомная формула или ее отрицание. Атомную формулу или ее отрицание будем называть литерой. Если А-атомная формула, то A и $\sim A$ будем называть дополнительными (комплиментарными, контрарными) литерами.

1) Исключение кванторов всеобщности. Так как все переменные связаны квантором всеобщности, а порядок расположения кванторов безразличен, то не будем указывать кванторы явным образом. Будем называть этот вид представления бескванторной нормальной формой.
2) Исключение связок $\wedge$. Исключаем связку $\wedge$, заменяя $A \wedge B$ на две формулы A, B. В результате многократной замены получим множество формул, каждая из которых представляет собой дизъюнкцию литер, называемую предложением (дизъюнктом).

Можно показать, что исходное множество формул A является невыполнимым тогда и только тогда, когда невыполнимо множество A', полученное из A применением указанных восьми операций.

Множество дизъюнктов невыполнимы тогда и только тогда, когда пустой дизъюнкт (обозначается $\square$) является логическим следствием из него. Таким образом, невыполнимость множества дизъюнктов S можно проверить, порождая логические следствия из S до тех пор, пока не получится пустой дизъюнкт.

Для порождения логических следствий будет использоваться очень простая схема рассуждений. Пусть A, B и X — формулы. Предположим, что две формулы $(A \vee X)$ и $(B \vee \sim X)$ — истинны. Если X тоже истинна, то отсюда можно заключить, что A истинна. В обоих случаях $(A \vee B)$ истинна, т. е. является тавтологией в исчислении высказываний. Если S содержит $\square$, то оно невыполнимо; если не содержит — то выводятся новые дизъюнкты до тех пор, пока не будет получен $\square$ (это всегда имеет место для невыполнимого S).

Принцип резолюции для логики высказываний. Основная идея принципа резолюции заключается в проверке, содержит ли множество дизъюнктов S пустой (ложный) дизъюнкт $\square$. Если это так, то S невыполнимо. Если S не содержит $\square$, то следующие шаги заключаются в выводе новых дизъюнктов до тех пор, пока не будет получен $\square$ (что всегда будет иметь место для невыполнимого S). Таким образом, принцип резолюции рассматривается как правило вывода, с помощью

которого порождаются новые дизъюнкты из S. По существу принцип резолюции является расширением однолитерального правила Девиса и Патнема. Действительно, используя это правило, например для дизъюнктов типа P и $\sim P \vee Q$, мы получаем дизъюнкт Q. Согласно этому правилу в двух дизъюнктах, один из которых состоит из одной литеры, а второй содержит произвольное число литер (в том числе и одну), находится контрарная пара литер (например P и $\sim P$, которая затем вычеркивается, и из оставшихся частей дизъюнктов формируется новый дизъюнкт (Q в нашем примере).

Ничего нового по сравнению с известным правилом вывода modus ponens здесь нет, так как $\sim P \vee Q$ равносильно $P \to Q$ и из выводимости P и $P \to Q$ следует выводимость Q.

Однолитеральное правило было расширено Дж. Робинсон на произвольных дизъюнктах с любым числом литер. Если в любых двух дизъюнктах $C1$ и $C2$ имеется контрарная пара литер (L и $\sim L$), то, вычеркивая ее, мы формируем новый дизъюнкт из оставшихся частей дизъюнктов. Этот вновь сформированный дизъюнкт будем называть резольвентой дизъюнктов C_1 и C_2 [5, 6].

Пример

Пусть

$$C_1: P \vee \sim Q \vee \sim R;$$
$$C_2: Q \vee P.$$

Тогда резольвента $C: P \vee \sim R$.

Обоснованность получения таким образом резольвенты вытекает из простой теоремы, которая гласит: резольвента C, полученная из двух дизъюнктов C_1 и C_2, является логическим следствием этих дизъюнктов.

Если в процессе вывода новых дизъюнктов мы получим два однолитеральных дизъюнкта, образующих контрарную пару, то резольвентой этих двух дизъюнктов будет пустой дизъюнкт □.

Таким образом, выводом пустого дизъюнкта □ из множества дизъюнктов S называется конечная последовательность дизъюнктов $C_1, C_2, \ldots, C_k$, такая, что любой C_i, является или дизъюнктом из S, или резольвентой, полученной принципом резолюции, и C_k = □.

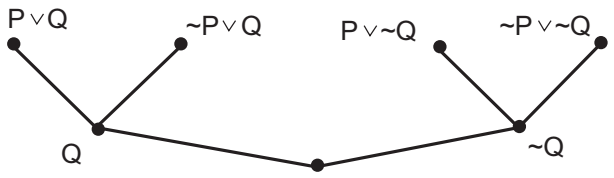


Рис. 3.2 – Пример вывода формулы.

Вывод пустого дизъюнкта может быть наглядно представлен с помощью дерева вывода, вершинами которого являются или исходные дизъюнкты, или резольвенты, а корнем — пустой дизъюнкт (рис. 3.2).

Пример

Пусть $S = \{P \vee Q, \sim P \vee Q, P \vee \sim Q, \sim P \vee \sim Q\}$
Тогда

1) $P \vee Q$;
2) $\sim P \vee Q$;
3) $P \vee \sim Q$;
4) $\sim P \vee \sim Q$;
5) $Q(1,2)$;
6) $\sim Q(3,4)$;
7) $\square\,(5,6)$.

В заключение приведем пример вывода формулы из множества посылок принципом резолюции. Напомним, что доказательство вывода формулы G из множества посылок $F_1, F_2, \ldots, F_n$ сводится к доказательству противоречивости формулы $F_1 \wedge F_2 \wedge \ldots \wedge F_n \wedge \sim G$ (процедура опровержения).

Пример

Если команда А выигрывает в футбол, то город А′ торжествует, а если выигрывает команда В, то торжествовать будет город В′. Выигрывает или А, или В. Однако если выигрывает А, то город В′ не торжествует, а если выигрывает В, то не будет торжествовать город А′. Следовательно, город В′ будет торжествовать тогда и только тогда, когда не будет торжествовать город А1.

Напишем посылки и заключение на языке логики высказываний и приведем их к форме дизъюнктов.

1) $A \to A'$
2) $B \to B'$
3) $A \vee B$
4) $A \to \sim B'$
5) $B \to \sim A'$
6) $B' \leftrightarrow \sim A'$

$\sim (B' \leftrightarrow \sim A') = \sim \big((\sim B' \vee \sim A') \wedge (B' \vee A')\big) = \sim (\sim B' \vee \sim A') \vee \sim (B' \vee A') =$
$= (B' \wedge A') \vee (\sim B' \wedge \sim A') = (A' \vee \sim B') \wedge (\sim A' \vee B')$.

1) $\sim A \vee A'$
2) $\sim B \vee B'$
3) $A \vee B$
4) $\sim A \vee \sim B'$
5) $\sim B \vee \sim A'$
6) $A' \vee \sim B'$
7) $\sim A' \vee B'$
8) $A' \vee B(1,3)$
9) $\sim B \vee A'(2,6)$
10) $A'(8,9)$
11) $A \vee \sim A'(3,5)$;
12) $\sim A \vee \sim A'(4,7)$;
13) $\sim A'(11,12)$;
14) $\square\,(10,13)$.

Принцип резолюции для логики предикатов первого порядка. Если в логике высказываний нахождение контрарных пар не вызывает трудностей, то для логики предикатов это не так. Действительно, если мы имеем дизъюнкты типа:

$$C_1: P(x) \vee \sim R(x);$$
$$C_2: \sim P\big(g(y)\big) \vee Q(z),$$

то резольвента может быть получена только после применения к C подстановки $g(y)$ вместо x.

Имеем	Однако для случая
$C_1': P\big(g(y)\big) \vee \sim R\big(g(y)\big);$	$C_1: P\big(f(x)\big) \vee \sim R(x);$
$C_2': \sim P\big(g(y)\big) \vee Q(z);$	$C_2: \sim P\big(g(x)\big) \vee Q(y),$
$C: \sim R\big(g(y)\big) \vee Q(z).$	

очевидно, никакая подстановка не применима и никакая резольвента не образуется. Отсюда следует определение того, что является подстановкой.

Подстановкой будем называть конечное множество вида $\{t_1/x_1, t_2/x_2, \ldots, t_n/x_n\}$, где любой t_i, — терм, а любая x_i — переменная $(1 \leqslant i \leqslant n)$, отличная от t_i.

Подстановка называется фундаментальной, если все t_i, $(1 \leqslant i \leqslant n)$ являются фундаментальными термами. Подстановка, не имеющая элементов, называется пустой подстановкой и обозначается через ε.

Пусть $\Theta = \{t_1/x_1, t_2/x_2, \ldots, t_n/x_n\}$ — подстановка и W — выражение. Тогда $W\Theta$ будем называть примером выражения W, полученного заменой всех вхождений в W переменной x_i, $(1 \leqslant i \leqslant n)$ на вхождение терма t_i. $W\Theta$ будет называться фундаментальным примером выражения W, если Θ — фундаментальная подстановка.

Например, применив к $W = \Big\{\big(P(x), f(y)\big) \vee \sim Q(z)\Big\}$ фундаментальную подстановку $\Theta = \big\{a/x, g(b)/y, f(a)/z\big\}$ получим фундаментальный пример $W\Theta = \Big\{P\Big(a, f\big(g(b)\big)\Big) \vee \sim Q\big((f(a))\big)\Big\}$.

Пусть $\Theta = \{t_1/x_1, t_2/x_2, \ldots, t_n/x_n\}$ и $\lambda = \{u_1/y_1, u_2/y_2, \ldots, u_m/y_m\}$ — две подстановки. Тогда композицией $\Theta \circ \lambda$ двух подстановок Θ и λ называется подстановка, состоящая из множества $\{t_1\lambda/x_1, \ldots, t_n\lambda/x_n, u_1/y_1, u_2/y_2, \ldots, u_m/y_m\}$ в котором вычеркиваются $t_i\lambda/x_i$, в случае $t_i\lambda/x_i$ и u_i/y_i если y_j находится среди $x_1, x_2, \ldots, x_n$.

Пример

$\Theta = \{g(x, y/x, z/y)\}$ и $\lambda = \{a/x, b/y, c/w, y/z\}$.
$\Theta \circ \lambda = \big\{g(a, b)/x, y/y, a/x, b/y, c/w, y/z\big\} = \big\{g(a, b)/x, c/w, y/z\big\}$.

Можно показать, что композиция подстановок ассоциативна, т. е. $(\Theta \circ \lambda) \circ \mu = \Theta \circ (\lambda \circ \mu)$.

Подстановку Θ будем называть унификатором для множества выражений $\{W_1, W_2, \ldots, W_k\}$, если $W_1\Theta = W_2\Theta = \ldots = W_k\Theta$. Будем говорить, что множество выражений $\{W_1, W_2, \ldots, W_k\}$ унифицируемо, если для него имеется унификатор. Унификатор σ для множества выражений $\{W_1, W_2, \ldots, W_k\}$ называется наиболее общим унификатором (НОУ) тогда и только тогда, когда для каждого унификатора Θ для этого множества выражений найдется подстановка λ, такая, что $\Theta = \sigma \circ \lambda$.

Пример

$W = \left\{P\left(x, a, f\left(g(a)\right)\right), P\left(z, y, f(u)\right)\right\}$. Тогда $\sigma = \{z/x, a/y, g(a)/u\}$ есть НОУ, а $\Theta = \{b/x, a/y, b/z, g(a)/u\}$ есть унификатор.

Опишем теперь алгоритм унификации, который находит НОУ, если множество выражений W унифицируемо, и сообщает о неудаче, если это не так.

1) Установить $k = 0$, $W_k = W$ и $\sigma_k = \varepsilon$. Перейти к пункту 2.
2) Если W_k не является одноэлементным множеством, то перейти к пункту 3. В противном случае принять $\sigma_k = \sigma$ и окончить работу.
3) Каждая из литер W_k рассматривается как цепочка символов, и выделяются первые подвыражения, не являющиеся одинаковыми у всех элементов W_k, т. е. образуется так называемое множество рассогласований типа $\{x_k, t_k\}$. Если в этом множестве x_k — переменная, а t_k — терм, отличный от x_k, то перейти к пункту 4. В противном случае окончить работу: W не унифицируемо.
4) Пусть $\sigma_{k+1} = \sigma_k \circ \{t_k/x_k\}$ и $W_{k+1} = W_k\{t_k, x_k\}$.
5) Установить $k := k + 1$ и перейти к пункту 2.

Пример

Найти НОУ для $W = \left\{P\left(y, g(z), f(x)\right), P\left(a, x, f\left(g(y)\right)\right)\right\}$.

1) $\sigma_0 = \varepsilon$ и $W_0 = W$.
2) Так как W_0 не является одноэлементным множеством, то перейти к пункту 3.
3) $\{y, a\}$, т. е. $\{a/y\}$.
4) $\sigma_1 = \sigma_0 \circ \{a/y\} = \varepsilon_0\{a/y\} = \{a/y\}$;
   $W_1 = W_0\{a/y\} = \left\{P\left(a, g(z), f(x)\right), P\left(a, x, f\left(g(a)\right)\right)\right\}$.
5) Так как W_1 опять неодноэлементно, то множество рассогласований будет $\{g(z), x\}$, т. е. $\{g(z)/x\}$.

6) $\sigma_2 = \sigma_1 \circ \{g(z)/x\} = \{a/y, g(z)/x\}$;
$W_2 = W_1\{g(z)/x\} = \left\{P\left(a, g(z), f\left(g(z)\right)\right), P\left(a, g(z), f\left(g(a)\right)\right)\right\}$.

7) Имеем $\{z, a\}$, т. е. $\{a, z\}$.

8) $\sigma_3 = \sigma_2 \circ \{a/z\} = \{a/y, g(a)/x, a/z\}$;
$W_3 = W_2\{a/z\} = \left\{P\left(a, g(a), f\left(g(a)\right)\right), P\left(a, g(a), f\left(g(a)\right)\right)\right\} = \left\{P\left(a, g(a), f\left(g(a)\right)\right)\right\}$;
$\sigma_3 = \{a/y, g(a)/x, a/z\}$ есть НОУ для W.

Можно показать, что алгоритм унификации всегда завершается на шаге 2, если множество W унифицируемо, и на шаге 3 — в противном случае.

Если две или более одинаковые литеры (одного и того же знака) дизъюнкта C имеют НОУ σ, то C_σ называется фактором дизъюнктора C.

Пусть C_1 и C_2 — два дизъюнкта, не имеющие общих переменных (это всегда можно получить переименованием переменных). И пусть L_1 и $L_2 =\sim L_1$ — литеры в дизъюнктах C_1 и C_2 соответственно, имеющие НОУ σ. Тогда бинарной резольвентой для C_1 и C_2 является дизъюнкт вида $C = (c_1\sigma - L_1\sigma)U(c_2\sigma - L_2\sigma)$. Бинарная резольвента может быть получена одним из четырех способов:

1) резольвента для C_1 и C_2;
2) резольвента для C_1 и фактора дизъюнкта C_2;
3) резольвента для фактора дизъюнкта C_1 и дизъюнкта C_2;
4) резольвента для фактора дизъюнкта C_1 и фактора дизъюнкта C_2.

Принцип резолюции обладает важным свойством — полнотой, которое устанавливается следующей теоремой (Дж. Робинсон): множество дизъюнктен S невыполнимо тогда и только тогда, когда существует вывод пустого дизъюнкта из S.

К сожалению, в силу неразрешимости логики предикатов первого порядка для выполнимого множества дизъюнктов S в общем случае процедура, основанная на принципе резолюции, будет работать бесконечно долго.

Приведем два примера, иллюстрирующие принцип резолюции для логики предикатов первого порядка.

Пример

Существуют студенты, которые любят всех преподавателей. Ни один из студентов не любит невежд. Следовательно, ни один из преподавателей не является невеждой.

Запишем эти утверждения на языке логики предикатов и приведем их к стандартному виду.

$$\exists x\Big(C(x) \wedge \forall\big(P(y) \to L(x,y)\big)\Big).$$
$$\forall x\Big(C(x) \to \forall\big(H(y) \to\sim L(x,y)\big)\Big).$$
$$\forall\big(P(y) \to\sim H(y)\big).$$

1) $C(a)$;
2) $\sim P(y) \vee L(a,y)$;
3) $\sim C(x) \vee \sim H(y) \vee \sim L(x,y)$; $\quad \sim \forall y\big(\sim P(y)\big) \vee \big(\sim H(y)\big) = \exists y\big(P(y) \wedge H(y)\big)$;
4) $P(b)$;
5) $H(b)$;
6) $L(a,b)$ $\quad (2,4)\sigma = \{b/y\}$;
7) $\sim H(y) \vee \sim L(a,y)$ $\quad (1,3)\sigma = \{a/x\}$;
8) $\sim L(a,b)$ $\quad (5,7)\sigma = \{b/y\}$;
9) $\square\,(6,8)$.

Пример

Все люди животные. Следовательно, голова человека является головой животного.

Пусть $L(x)$, $A(x)$ и $G(x,y)$ означают «x — человек», «x — животное» и x — является головой y».

Тогда $\forall y\big(L(y) \to A(y)\big)$

$$\sim \forall x\Big(\exists y\big(L(y) \wedge G(x,y)\big) \to \exists z\big(A(z) \wedge G(x,z)\big)\Big).$$
$$\sim \forall x\Big(\exists y\big(L(y) \wedge G(x,y)\big) \to \exists z\big(A(z) \wedge G(x,z)\big)\Big) =$$
$$\sim \forall x\Big(\exists y\big(L(y) \wedge G(x,y)\big) \vee \exists z\big(A(z) \wedge G(x,z)\big)\Big) =$$
$$\sim \forall x\Big(\forall y\big(\sim L(y) \vee \sim G(x,y)\big) \vee \exists z\big(A(z) \wedge G(x,z)\big)\Big) =$$
$$\sim \forall x \forall y \exists z\big(\sim L(y) \vee \sim G(x,y) \vee A(z) \wedge G(x,z)\big).$$
$$\sim \exists x \exists y \forall z\Big(L(y) \wedge G(x,y) \wedge \big(\sim A(Z) \vee \sim G(x,z)\big)\Big).$$

1) $\sim L(y) \vee A(y)$;
2) $L(b)$;
3) $G(a,b)$;
4) $\sim A(z) \vee \sim G(a,z)$;
5) $A(b)$; $(1.2)\sigma = \{b/y\}$;
6) $\sim G(a,b)$ $(4.5)\sigma = \{b/z\}$;
7) $\square$ (3.6).

. . . . . .

Принцип резолюции является более эффективной процедурой вывода, нежели процедура Эрбрана. Но и он страдает существенным недостатком, заключающимся в формировании всевозможных резольвент, большинство из которых оказываются излишними и поэтому ненужными. С 1965 года и по сей день появляются всевозможные модификации принципа резолюции, направленные на нахождение более эффективных стратегий поиска нужных дизъюнктов.

Дедуктивный вывод используется на всех моделях знаний.

Стратегии поиска. Для решения проблемы «комбинаторного взрыва», возникающей при непосредственном применении принципа резолюций, необходимо осуществлять целенаправленный подбор предложений, участвующих в процессе унификации. Выбор должен осуществляться так, чтобы пустое предложение достигалось как можно скорее. Конечно, на каждом этапе опровержения выбор зависит от текущей ситуации. Однако имеются общие стратегии поиска, которые, вне зависимости от контекста задачи, позволяют сократить число резолюций. Ниже рассмотрим следующие четыре стратегии: стратегию унитарности, стратегию опорного множества, стратегию опорных данных, линейную стратегию.

Стратегия унитарности. Согласно этой стратегии предпочтение отдается тем бинарным резолюциям, где одно из предложений содержит единственный литерал. Такая стратегия гарантирует уменьшение длины результирующих предложений. Стратегия унитарности упорядочивает выбор пар предложений, упрощая последующие этапы доказательства невыполнимости исходного множества дизъюнктов.

Стратегия опорного множества. Более эффективной может оказаться попытка исключить из рассмотрения сразу несколько предложений на основе понятия опорного множества. Невыполнимое множество предложений S можно разбить на два подмножества $S1$ и $S2$: подмножество $S1$ состоит из аксиом, а $S2$ — из отрицаний тех предложений, которые необходимо доказать.

Очевидно, что подмножество $S1$ не содержит противоречий, поэтому нахождение резольвент предложений из $S1$ не допускается. Опорным называется множество предложений, состоящих из дизъюнктов, входящих в $S2$, и резольвент тех пар предложений, из которых хотя бы одно принадлежит опорному множеству. Стратегия опорного множества допускает выполнение тех резолюций, в которых, по крайней мере, одно из предложений входит в опорное множество. Если опорное множество относительно небольшое, то это значительно сокращает пространство поиска. Достоинство стратегии опорного множества также состоит в том, что формируемое дерево опровержения легко интерпретируется человеком, так как оно управляется целью.

Если исходное множество предложений невыполнимо, то рассмотренная стратегия приводит к пустому предложению, то есть она является полной. Стратегия опорного множества является специальным случаем более общей семантической резолюции.

Линейная резолюция. Сначала находится следствие двух предложений исходного множества S. На следующих этапах в резолюции участвуют резольвента C_1, полученная на предыдущем шаге (называемая центральным дизъюнктом), и дизъюнкт B (называемый боковым), являющийся либо одним из дизъюнктов исходного множества S, либо центральным дизъюнктом C_j, который предшествует в выводе дизъюнкту C_i, то есть $j < i$. Линейная стратегия является полной и одной из наиболее эффективных в реализации.

Стратегия входных данных. Такая стратегия соответствует линейной резолюции с одним ограничением: в качестве боковых дизъюнктов выбираются только дизъюнкты исходного множества S. Для примера (рис. 3.3) приведем изображение дерева опровержения невыполнимого множества дизъюнктов $S = \{R(x) \vee Q(x), \sim Q(f(z), \sim R(w) \vee P(b), \sim P(y))\}$, являющихся формулами исчисления предикатов. В дереве указаны дизъюнкты, получающиеся в результате соответствующих подстановок.

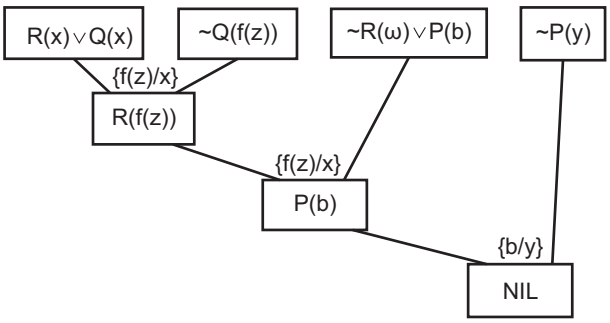


Рис. 3.3 – Дерево опровержения.

Входная резолюция является полной, только если предложения множества S являются дизъюнктами Хорна.

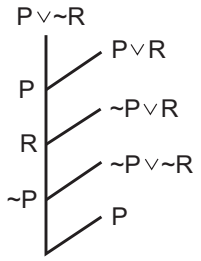
Рассмотренные стратегии поиска можно комбинировать. Например, стратегии унитарности и опорного множества. Кроме этого, можно упрощать множество исходных предложений S. Для этого исключаются тавтологии, предложения, содержащие уникальные литералы, подслучаи. Например, если множество S содержит дизъюнкт как $P(x)$, то не имеет смысла включать в состав S дизъюнкты вида $P(a)$ или $P(a) \vee Q(b)$, так как они представляют подслучаи $P(x)$.

Приведем примеры метода резолюций при рассмотрении логики высказываний и предикатов.

Пример

Пусть $S = \{P \vee R, \sim P \vee R, P \vee \sim R, \sim P \vee \sim R\}$. Тогда линейный вывод пустого дизъюнкта из S представлен ниже. Отметим, что из четырех боковых дизъюнктов

три принадлежат S и только один P является центральным дизъюнктом.



Рассуждения по поводу знаний. В большинстве встречающихся в ИИ систем относящиеся к области экспертизы знания делятся на 2 категории: факты и правила. Факты — это данные (представлены предикатами), касающиеся области экспертизы. Например, данные о персонале некоего университета составляют множество фактов [1].

Факт 1: Иванов — профессор факультета информатики, Проф(инфо, Иванов1)

Факт 2: Мария — студентка математического факультета, Студ(мат, Мария4).

Правила — это данные, представленные с помощью импликаций (или в иной эквивалентной логической форме). Они представляют собой обобщающие знания об области экспертизы.

Правило 1. Если y — профессор факультета x и w — студентка факультета z, при $x \neq z$, то y может служить внешним экзаменатором для w,

$\text{Проф}(x, y) \wedge \text{Студ}(z, w) \wedge \text{Равно}(x, z) \rightarrow \text{экзам}(y, w)$.

Задача доказательства (обоснования) теоремы состоит в установлении выводимости из фактов и правил некой формулы (предложения — цели, заключения), представляющей некоторый вопрос.

Предложение — цель (1).

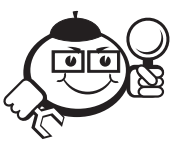
Может ли Иванов служить внешним экзаменатором для Марии?

Экзам(Иванов1, Мария4).

Можно ли применить различные приемы для выработки стратегий доказательства теоремы? Да.

3.1.4 Прямой и обратный дедуктивный вывод

Системы прямой дедукции. В системах прямой дедукции новые знания получают, применяя выводы к фактам и правилам. Алгоритм завершает работу при получении некоторого знания, эквивалентного цели (или непосредственно влекущего ее). Для иллюстрации системы прямой дедукции обратимся опять к этому примеру.



Пример

Докажем теорему:

$\big(\text{факт}(1) \wedge \text{факт}(2) \wedge \text{Правило}(1)\big) \rightarrow \text{Цель}(1)$.

Этап(1): Преобразуем факт(1) и Правило(1) в дизъюнкты, чтобы применить затем метод резолюций. С помощью резолюции выводим новое Правило(2), используя обозначение:

Факт(1)	Правило(1)
Проф(инфо, Иванов1)	~ Проф$(x, y)\vee$ ~ Студ$(z, w) \vee$ Равно$(x, z) \vee$ Экзам(y, w)

Правило(2): ~ Студ$(z, w) \vee$ Равно(Инфо, $z) \vee$ Экзам(Иванов1, $w)$.

Этап(2): Из факта(2) и Правила(2) резолюцией выводим новый факт(3):

Факт(2)	Правило(2)
Студ(мат, Мария4)	~ Студ$(z, w) \vee$ Равно(инфо, $z) \vee$ Экзам(Иванов1, $w)$

Факт(3)Экзам(Иванов1, Мария4) $\vee$ Равно(инфо, мат) = $\square$.

Отношение Равно(инфо, мат) = должно быть явно указано в БД.

Этап(3): Факт(3) соответствует Цели(1). Следовательно, она подтверждена. Аналогично выведем утверждение из факта(3) и отрицания Цели(1):

Факт(3)	~ Цель(1)
Экзам(Иванов1, Мария4)	~ Экзам(Иванов1, Мария4)

Систему прямой дедукции можно толковать как систему, которой применяется теорема о прямой дедукции: если $F_1, F_2, \ldots, F_n, G$ — логические выражения, то G является логическим следствием из $F_1, \ldots, F_n$ тогда и только тогда, когда логическое выражение $(F_1 \wedge F_2 \wedge \ldots \wedge F_n \wedge \sim G)$ тождественно ложно, т. е. невыполнимо. Правила вывода и стратегии, используемые в системах прямой дедукции, графически представлены И-ИЛИ-деревьями.

. . . . . .

Системы обратной дедукции.

В системах обратной дедукции выводы применяют к цели и к правилам, чтобы построить новые частные цели. Алгоритм завершает работу, когда все частичные цели соответствуют фактам. Такую систему можно толковать с логической точки зрения как систему, в которой применяется теорема об обратной дедукции, которая гласит:

если $F_1, F_2, \ldots, F_n G$ — логиические выражения, то G является логическим дизъюнктом из $F_1, F_2, \ldots, F_n$ тогда и только тогда, когда логическое выражение $(\sim F_1 \vee \vee \sim F_2 \vee \ldots \vee \sim F_n \wedge G)$ тождественно истинно, т. е. общезначимо.

В системах обратной дедукции правила и цели преобразуют в конъюнкты, чтобы применить затем правило согласия. Оно двойственно правилу резолюций. Если его применить к конъюнкциям $p \wedge c_1$, и $\sim p \wedge c_2$, то получим конъюнкцию $c_1 \wedge c_2$. Эта операция была введена в рамках булевой алгебры.

Пример

Этап(1): Из Цели(1) и отрицания Правила(1), используя правило согласия, выводим новую Цель(2):

Цель(1)	~ Правило(1)
Экзам(Иванов1, Мария4)	Проф(x, y) ∧ Студ(z, w)∧ ~ Равно(x, z)∧ ~ Экзам(y, w)

Цель(2): Проф$(x,$ Иванов 1$)$ ∧ Студ$(z,$ Мария4$)$∧ ~ Равно(x, z).

Этап(2): Из Цели(2) и отрицания факта(1) с помощью правила согласия выводим Цель(3).

Цель(2)	~ Факт(1)
Проф$(x,$ Иванов1$)$ ∧ Студ$(z,$ Мария4$)$∧ ~ Равно(x, z)	~ Проф(Инфо, Иванов1)

Цель(3): студ$(z,$ Мария4$)$∧ ~ Равно$($Инфо$, z)$.

Этап(3): из Цели(3) и отрицания факта(2) выводим теорему:

Цель(3)	Факт(2)
Студ(z, Мария4)∧ ~Равно(Инфо, z)	~Студ(мат, Мария4)

~ Равно(Инфо, мат) = И

Практическим применением принципа резолюции в системе ИИ являются: информационный поиск, планирование перемещения робота, автоматическое написание программ, логическое программирование (Пролог), экспертные системы и т. д.

3.2 Абдуктивный вывод

Абдукция — это процесс формирования объясняющей гипотезы. Точнее, при заданной теории и наблюдении, предложенном для объяснения, абдуктивный вывод должен определить одно или более наилучших объяснений наблюдения на основе заданной теории [4].

Простейший пример абдуктивного вывода. Пусть имеется правило «Все шары в этой коробке белые» и результат наблюдения «Эти шары белые», тогда по абдукции можно заключить, что «Данные шары взяты из этой коробки» (причина — объяснение). Чаще всего абдукция используется в задаче диагностики для обнаружения причины наблюдаемого неправильного поведения системы. В приложениях систем искусственного интеллекта абдукция используется в задачах:

- понимания Е-Я;
- планирования;
- распознавания плана;
- накопления и усвоения знаний.

Абдукция предназначена для объяснения или причин наблюдаемых явлений и факторов. Форма абдуктивного вывода по Пирсу:

- наблюдается удивительный факт C, не следующий из наших знаний о мире;
- если бы A было истинным, то C могло бы произойти (A — объясняющая гипотеза);
- следовательно, можно предположить, что A истинно.

Таким образом, абдуктивный вывод представляется схемой: даны правило и результат, требуется вывести частный случай — причину. Если правило выражается через связь — импликация (не обязательно), то правило силлогизма можно представить следующим образом:

$$\frac{A \to C, C}{A},$$

где A может быть причиной C или доводом в пользу того, чтобы C было истинным.

В искусственном интеллекте под абдуктивным выводом понимается вывод наилучшего абдуктивного объяснения. Чаще всего критерием является степень правдоподобия. Абдуктивные решения входят как в класс строгих так и в класс эвристических решений и отличаются большой неопределенностью. Они представляют процесс выявления наиболее вероятных исходных утверждений (посылок, причин и т. д.) из некоторого заключительного утверждения на основе обратных преобразований. Абдуктивные решения строятся на широком использовании прошлого опыта. Пусть, например, между некоторым множеством посылок $x_i (x_i \in X)$ и следствий $y_i (y_i \in Y)$ обнаружена причинно-следственная связь R, тогда наиболее вероятной причиной, появления некоторого нового следствия $y_j (y_j \in Y)$ является посылка $x_j = R^{-1}(y_j)$, причем $x_j \in X$.

Абдуктивные решения встречаются очень часто в науке и повседневной жизни: анализ хозяйственной деятельности предприятия, изучение космических излучений. Важно отметить, что принимающий абдуктивное решение может не осознавать неопределенность своего решения.

Приведем простейший пример:

Y: Сократ — смертен;

R: Все люди смертны;

X: Сократ человек (но Сократ может быть и собакой)

Методы вероятностных абдуктивных рассуждений в сложноструктурированных проблемных областях представлены у В. Н. Вагина, но мы рассматривать их не будем.

Лукасевич все логические выводы поделил на два класса:

Дедукция $\left(\frac{A \to B, B}{B}\right)$ и редукция $\left(\frac{A \to B, B}{A}\right)$.

Дедукция позволяет сделать явным все истинные утверждения, определяемые заданным множеством утверждений, а редукция позволяет обнаруживать объяснения для утверждений, описывающих, например, новые факты. Редуктивный вывод не сохраняет истину, но с его помощью можно получать новую информацию. К редукции можно отнести как индуктивный, так и абдуктивный выводы.

3.3 Индуктивный вывод

Кроме абдуктивного и дедуктивного выводов, в системах искусственного интеллекта широко используются индуктивные схемы вывода. Такие схемы вывода позволяют порождать обобщения имеющихся частных утверждений. Способность к обобщению является важной функцией естественного интеллекта и используется им для получения новых знаний. Поэтому индуктивные схемы вывода являются элементом обучающихся подсистем искусственного интеллекта. Так как при обучении из совокупности имеющихся фактов формируются путем обобщения новые понятия и факты, то индуктивные схемы вывода находят широкое применение при автоматизации процесса приобретения знаний.

Общая проблема индукции состоит в выяснении того, «на что мы опираемся, когда переходим от фактов к закону, который выражает некоторые регулярности природы».

3.3.1 Виды индукции

Под классической индукцией понимается способ движения мысли от частного к общему, от знания менее общего к знанию более общему. Исторически сложилось три различных вида индукции [3, 7, 8, 10, 12, 13]:

1) Перечислительная или энумеративная индукция (полная и неполная).
2) Элиминативная индукция (схемы установления причинно-следственных связей между явлениями).
3) Индукция как обратная дедукция (рассуждения от следствия к основаниям).

Впервые логически обосновали индукцию древнегреческие философы: индукцию как обратную дедукцию — Сократ и Платон, перечислительную индукцию — Аристотель, элиминативную — эпикурийцы. Схематически эти виды индукции представлены на рисунке 3.4:

1. *Энумеративную индукцию* Ф. Бекон назвал «индукция через простое перечисление, в котором не встречается противоречащегося случая».

Приведем два примера:

Пример 1: Первая ворона — черная, вторая ворона — черная, . . .

Все наблюдаемые вороны черные (полная индукция).

Пример 2: Первая ворона — черная, вторая ворона — черная, . . .

Все вороны черные (неполная индукция).

Полная индукция через простое перечисление выводится на основании изучения всех имеющихся фактов. Это возможно лишь в том случае, когда количество возможных фактов конечно и невелико. Только полную индукцию рассматривал Аристотель, как формально законный вид вывода, и называл он ее силлогистической.

Если заключения выводятся на основании изучения лишь части фактов, то индукция называется неполной. Неполную индукцию анализировал Б. Рассел, которому принадлежит простое и изящное доказательство ее несостоятельности в качестве чисто логического принципа. Заключения, выполненные с помощью неполной

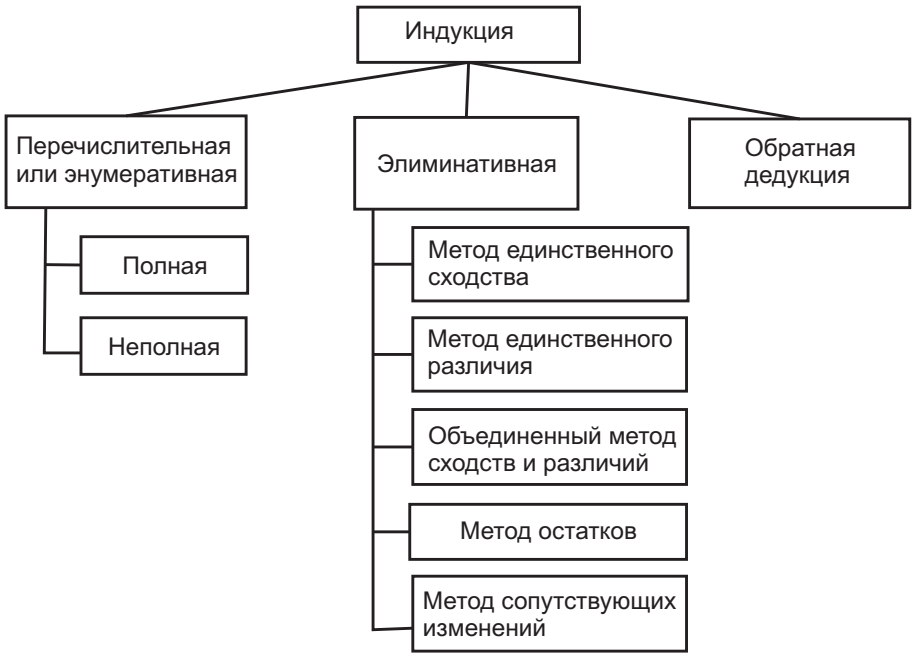


Рис. 3.4 – Виды индукции.

индукции, являются правдоподобными, так как из истинности частного не обязательно следует истинность общего.

В основе индуктивных выводов лежит правило индуктивного обобщения

$$\frac{A, B \Rightarrow F}{B},$$

где A — множество известных фактов, B — понятие (гипотеза, иногда обозначается H). Смысл этого правила заключается в следующем. Пусть имеется множество фактов A. Если ввести некоторую гипотезу B и показать, что из B выводим любой факт A, то гипотеза B верна. Особенностью правила индуктивного обобщения является то, что множество объектов, описываемых гипотезой B, шире, чем множество, соответствующее фактам A. Это может приводить к тому, что из гипотезы B будут выводиться и другие факты. Следовательно, есть риск совершить ошибку. Поэтому при выборе гипотезы B необходимо стремиться к минимальному обобщению.

Очень часто в современной литературе неполную индукцию через простое перечисление рассматривают как способ формирования некоторой гипотезы. Заключение (обобщение) в виде общего суждения берется как гипотеза, догадка, которую в ходе последующего исследования надлежит или более строго обосновать, или оценить, или опровергнуть. Как можно оценить гипотезы?

Один из самых распространенных способов — это применение методов математической статистики («вероятностная индукция»), другой — с использованием логик («структурная индукция»).

Теперь обсудим роль контрпримеров для обобщений (гипотез), сделанных в результате исследований. Представим ситуацию, когда первый эксперимент показал результат, который гипотеза считала невозможным. Что делают в этом случае?

Существует точка зрения, что не согласующийся с гипотезой эксперимент опровергает ее.

Остановимся на анализе примера, приведенного Д. Юмом. В результате обобщения фактов посредством индукции через простое перечисление было сформировано суждение: все лебеди — белые (1). В дальнейшем оказалось, что это обобщение оказалось неверным: в Австралии были обнаружены черные лебеди. Таким образом, было опровергнуто суждение (1). Это, однако, не означало, что наша обобщающая деятельность, выразившаяся в формулировании (1), была бесполезна. Обнаружение черных лебедей заставило к числу известных ранее шести видов лебедей добавить новый вид. Открытие черных лебедей, следовательно, привело в логическом смысле к расщеплению одного понятия на два.

Обобщая изложенное, можно сказать, что опровержение противоречащими примерами общих утверждений, даже если они получены индукцией через простое перечисление, не означает их конкретизацию — внесение различий в тождественное.

2. *Элименативная индукция.* Указание причины явления играет решающую роль для перевода гипотезы в разряд правдоподобных. Приведем пример. Классический образец абсурда обычно иллюстрируется высказыванием: «В огороде — бузина, а в Киеве — дядька». Попытаемся из этого высказывания сформулировать некоторую гипотезу о том, что всегда имеют место одновременно два события: 1) в огороде цветет бузина, 2) конкретный «дядька» находится в Киеве. Свяжем оба события такой информацией: «Дядька — житель Киева, пенсионер, все свое время проводит на даче, кроме того периода, когда цветет бузина. Почему? Потому что у него аллергия к цветам бузины». Теперь видно, что классический образец абсурда оказался вполне правдоподобной гипотезой.

Таким образом, мы вплотную подошли к элиминативной индукции, ибо, по словам Д. С. Милля, понятие о причине есть корень всей теории индукции.

Д. С. Милль предложил следующие методы (модусы) индуктивных выводов.

Метод сходства. Если два или более случая подлежащего исследованию явления имеют общим лишь одно обстоятельство, то это обстоятельство, в котором только и согласуются все эти случаи, есть причина (или следствие) данного явления.

Метод различия. Если случай, в котором данное явление наступает, и случай, в котором оно не наступает, сходны во всех обстоятельствах, кроме одного, встречающегося лишь в первом случае, то это обстоятельство, в котором они только и разнятся, есть следствие, или причина, или необходимая часть причины.

Метод сопутствующих изменений. Всякое явление, изменяющееся определенным образом всякий раз, когда некоторым особенным образом изменяется другое явление, есть либо причина, либо следствие этого явления, либо соединено с ним какой-либо причинной связью.

Метод остатков. Если из явления вычесть ту его часть, которая, как известно из прежних индукций, есть следствие некоторых определенных предыдущих, то остаток данного явления должен быть следствием остальных предыдущих.

Соединительный метод сходства и различия. Если два или более случая возникновения явления имеют общим лишь одно обстоятельство и два или более случая возникновения того же явления имеют общим только отсутствие того же самого обстоятельства, то это обстоятельство, в котором только и разнятся оба

ряда случаев, есть или следствие, или причина, или необходимая часть причины изучаемого явления.

Не возражая вообще против выбора случаев, в чем-то сходных, представители элиминативной индукции между тем настаивают на такой выборке, которая предполагала бы это сходство в условиях максимального разнообразия и варьируемости элементов исследуемого класса.

Тогда то общее, что сохраняется в максимально отличных друг от друга проявлениях одного и того же случая, исчезает с исчезновением этого случая или изменяется с его изменением, только и может стать объектом индуктивного обобщения. При таком подходе в начале исследования, естественно, берегся несколько конкурирующих между собой гипотез, которые в процессе индуктивного анализа постепенно исключаются (элиминируются) в пользу одной какой-либо гипотезы.

3. Теперь рассмотрим третий тип индукции — *индукцию как обратную дедукцию.* Решающий вклад здесь внесли Ньютон, Джевонс. Исследуя соотношение индукции и дедукции, Джевонс писал: «В дедукции мы имеем дело с развитием выводов из закона. Индукция же есть совершенно обратный процесс. Здесь даются известные результаты, и требуется открыть общий закон, из которого они вытекают».

Н. Ньютон, Ст. Джевонс трактуют индукцию, прежде всего, как метод открытия общих законов и принципов, объясняющих известные факты и менее общие законы и принципы. При этом все они сознавали неоднозначный (и в этом случае не чисто логический) характер движения познающей мысли при индуктивном восхождении от фактов к объясняющим их законам и принципам. Вместе с тем индуктивное восхождение не является с их точки зрения и чисто произвольным, полностью интуитивным. Индуктивное восхождение является «правильным» тогда и только тогда, когда факты (основа, исходный пункт индукции) могут быть чисто логически (дедуктивно) выведены из предложенного для их объяснения закона или принципа.

3.3.2 Индукция как вывод и индукция как метод

Следует различать индукцию как метод и вывод. С одной стороны, Аристотель трактует индукцию как метод опытного назначения, как путь движения мысли от чувственно-воспринимаемого единичного к общим понятиям и утверждениям. С другой стороны, он впервые исследовал индуктивный способ движения мысли со стороны его логической формы, а именно как вывод, как специфический тип рассуждений.

Индукцию как вывод делят на материальную и формальную. Формальная индукция это индукция как вывод только с точки зрения ее логической формы, без всякой связи с содержанием посылок индукции, от которого полностью отвлекаются. Материальная индукция — это рассуждение (вывод) от частного к общему с учетом не только его логической формы, но и его содержания.

Когда в науке говорят об индуктивных выводах, то, как правило, имеют в виду именно материальную индукцию как логический способ движения научно-исследовательской мысли от опыта к теории как при получении новых научных законов путем обобщения опытных данных, так и при обосновании имеющегося научного знания эмпирическим путем.

Необходимо понять, что доказательность любых дедуктивных выводов не зависит от содержания посылок, а только от их логической формы. Для многих индуктивных выводов имеет место обратное. Здесь учет содержания посылок и, в частности, учет релевантности и существенности заключенной в них информации имеет первостепенное значение для оценки правильности и доказательности сделанных индуктивных заключений.

Приведем некоторые примеры формализации индукции как вывода. Рассматривая интерпретацию законов Д. С. Милля в рамках исчисления высказываний, можно получить следующие правила вывода:

Метод сходства Принцип единственного сходства

$$\begin{array}{l} a \wedge x \Rightarrow y \\ b \wedge x \Rightarrow y \\ \_\_\_\_\_ \\ x \Rightarrow y \end{array} \qquad n\begin{cases} a,b,c \Rightarrow d \\ a,b,c \Rightarrow d \\ \_\_\_\_\_ \\ a,b,c \Rightarrow d \end{cases} \qquad n\begin{cases} a,b \Rightarrow d \\ a,b \Rightarrow d \\ \_\_\_\_\_ \\ a,b \Rightarrow d \end{cases} \qquad n\begin{cases} a \Rightarrow d \\ a \Rightarrow d \\ \_\_\_\_ \\ a \Rightarrow d \end{cases}$$

Метод различия Принцип единственного различия

$$\begin{array}{l} a \wedge x \Rightarrow y \\ b \wedge \sim x \Rightarrow \sim y \\ \_\_\_\_\_ \\ x \Rightarrow y \end{array} \qquad n\begin{cases} a,b,c \Rightarrow d \\ a,b,c \Rightarrow d \\ \_\_\_\_\_ \\ a,b,c \Rightarrow d \end{cases} \qquad n\begin{cases} b,c \not\Rightarrow d \\ b,c \not\Rightarrow d \\ \_\_\_\_\_ \\ b,c \Rightarrow d, \text{т. е. } a \Rightarrow d \end{cases}$$

Метод остатков Принцип единственного остатка

$$\begin{array}{l} z \wedge x \Rightarrow y \wedge w \\ x \Rightarrow w \\ \_\_\_\_\_ \\ x \Rightarrow y \end{array} \qquad n\begin{cases} a,b,c \Rightarrow d,e \\ a,b,c \Rightarrow d,e \\ \_\_\_\_\_ \\ a,b,c \Rightarrow d,e \end{cases} \qquad n\begin{cases} b,c \Rightarrow e \\ b,c \Rightarrow e \\ \_\_\_\_\_ \\ b,c \Rightarrow e, \text{т. е. } a \Rightarrow d \end{cases}$$

Здесь a, b, x, y, z, w могут обозначать не только переменные, но и любые формулы.

Эти правила могут приводить к противоречию, но подобные противоречия являются отражением реальных противоречий, существующих в экспериментальных данных. Все выводы, полученные по этим правилам, есть гипотезы, достоверность которых должна быть оценена.

Интерпретация законов Д. С. Милля в рамках расширенного языка логики предикатов привела к созданию одного из наиболее ярких и интересных методов — ДСМ — метода автоматического порождения гипотез в экспертной системе с неполной информацией.

Реализация метода, предложенного Д. С. Миллем в интеллектуальных системах, представлена ниже.

Индуктивные решения (от латинского слова induction — наведение, побуждение) входят в класс эвристических решений, отличаются неопределенностью, представляют процесс выявления наиболее вероятных закономерностей, механизм действия, вытекающий из сопоставления исходных утверждений. Иными словами, индуктивные решения выявляют гипотезу A или оператор R по входным Xi и выход-

ным *Yi* сигналам. Индуктивные решения наиболее свойственны мышлению. Например, ребенок, поставив перед собой задачу, построить домик из кубиков *Yi*, предпринимает такие действия *R*, чтобы добиться желаемого исхода из наличия имеющегося у него материала *Xi* (кубиков). Если сложнее, то это действия врача при лечении больного, действия руководителя предприятия при организации выполнения плана и т. д.

Отметим, что выявление этим способом оператора (гипотезы) *R*, который уменьшается по мере накопления опыта и рассмотрения решения на нескольких уровнях, является неоднозначным. Живые существа осуществляют индукцию именно этим способом, а «эвристические способности человека являются результатом одновременного обобщения данного события на разных уровнях». При этом решения, полученные на более высоком уровне, доминируют над решением более низкого уровня, отбрасывая его, если оно оказывается неверным.

Так, естественным предположением о нахождении следующей цифры в последовательности $1, 2, 3$ может явиться цифра 4, которая в сочетании с остальными образует упорядоченное множество целых чисел $1, 2, 3, 4, \ldots$.

Сообщение о том, что предположение неверно, приводит к новой гипотезе — гипотезе последовательности простых чисел, в результате чего называется цифра 5 и т. д.

Пример

Приведем наш пример с индуктивным типом решения и основными параметрами: x, y (дано), R (найти).

x: Сократ — человек;

y: Сократ — смертен;

R: Все люди смертны, где вывод представляет одну из гипотез.

ДСМ-метод. Сущность ДСМ-метода заключается в следующем. Пусть задано множество причин $A = \{A_b A_2, \ldots, A_p\}$ множество следствий $B = \{B_u B_2, \ldots, B_m\}$ и множество оценок $Q = \{q_1, q_2, \ldots, q_r\}$. Выражение вида $A_i \Rightarrow B_j$ называется положительной гипотезой, выражающей утверждение «A_i является причиной B_j с оценкой достоверности q_k». Отрицательной гипотезой называется выражение $A_i \Rightarrow B_j$, которое формируется «A_i — не является причиной B_j с оценкой достоверности q_k». Положительные гипотезы будем обозначать $h^+_{l,j,k}$, отрицательные — $h^-_{l,j,k}$. Среди значений выделим два специальных, которые можно интерпретировать как «ложь» (0) и «истина» (1). Гипотезы с этими оценками можно рассматривать как явления, истинность или ложность которых твердо установлена. Остальные значения между 0 и 1 будем обозначать рациональными числами k/n, где $k = 1, \ldots, n-1$, а n характеризует число примеров.

Обобщенный алгоритм ДСМ-метода включает следующие шаги [12].

На основе исходного множества положительных и отрицательных примеров (наблюдений) формируется набор гипотез, которые записываются в матрицы M^+

и M^-. Гипотезы формируются на основе выявления сходства и различия в примерах. Матрицы имеют вид:

$$M^+ = \begin{array}{cccc} & B_i & \dots & B_w \\ A_l & h^+_{lik} & \dots & h^+_{lwm} \\ \dots & \dots & \dots & \dots \\ A_r & h^+_{ris} & \dots & h^+_{rwd} \end{array}$$

$$M^- = \begin{array}{cccc} & B_j & \dots & B_v \\ A_x & h^-_{xjk} & \dots & h^-_{xvf} \\ \dots & \dots & \dots & \dots \\ A_z & h^-_{zim} & \dots & h^-_{zvf} \end{array}$$

К исходному множеству примеров добавляются новые наблюдения, которые могут либо подтверждать выдвинутые гипотезы, либо опровергать их, при этом оценки гипотез изменяются следующим образом. Если некоторая гипотеза h_{ijk} имела оценку $q_k = k/n$, то при появлении нового примера $(n + 1)$ проводится проверка на подтверждение этой гипотезы. В случае положительного ответа оценка $q_k = (k + 1)/(n + 1)$, иначе $q_k = (k - 1)/(n + 1)$. В процессе накопления информации оценки выдвинутых гипотез могут приближаться к 1 или 0. Изменение оценок может также иметь колебательный характер, что, как правило, ведет к исключению таких гипотез из множеств M^+ или M^-.

1) Циклическое добавление примеров, сопровождающееся изменением оценок достоверности гипотез с периодическим изменением множеств M^+ или M^-.
2) Завершение процесса индуктивного вывода при выполнении условий окончания цикла. В качестве таких условий могут использоваться меры близости значений q_i к 0 или 1, а также дополнительные условия, которые могут быть связаны с ограничением времени (количества новых примеров) вывода и т. п.

В современных модификациях ДСМ-метода используются выводы по аналогии, проводится учет контекста реализации причинно-следственных отношений, применяются нечеткие описания фактов и т. д.

Контрольные вопросы и задания к главе 3

1) Опишите характеристики систем логического вида.
2) Определите понятие «рассуждение».
3) Сформулируйте отличия логического вывода от рассуждений.
4) Определите термин «дедукция» как термин современной логики.
5) Представьте четыре формы, выраженные на естественном языке и представляющее собой суждения.
6) Запишите четыре принципа (модуса) дедуктивного вывода для высказываний.

7) Охарактеризуйте правило цепного заключения и приведите пример вывода с применением правил вывода.
8) Поясните смысл полного определения понятия «дедукция».
9) Охарактеризуйте процесс дедукции.
10) Опишите два основных метода решения проблемы доказательства в логике.
11) Приведите пример тавтологии и докажите ее истинность.
12) Приведите пример противоречия.
13) Приведите пример построения теории и ее проверку на полноту и непротиворечивость.
14) Рассмотрите три основных стратегии доказательства ППФ.
15) Расскажите о представлении и решении задач в виде теорем.
16) Представьте формулировку задачи дедуктивного вывода.
17) Приведите тождественные преобразования формул в дизъюнкты.
18) Сформулируйте принцип резолюции для логики высказываний и приведите пример.
19) Сформулируйте принцип резолюции для логики предикатов первого порядка и приведите пример.
20) Рассмотрите стратегии поиска предложений при доказательстве теорем.
21) Чем отличаются прямой и обратный дедуктивные выводы?
22) Дайте определение и характеристики абдуктивного вывода.
23) Приведите пример абдуктивного вывода.
24) Охарактеризуйте три различных вида индукции.
25) Какие методы (модусы) индуктивного вывода предложил Д. С. Миль?
26) Представьте формальные правила индуктивного вывода Д. С. Миля.
27) Запишите правило индуктивного обобщения и поясните смысл этого правила.

Литература к главе 3

[1] Алиев Р. А. Производственные системы с искусственным интеллектом / Р. А. Алиев, Н. М. Абдикеев, М. М. Шахназаров. — М. : Радио и связь, 1990. — 246 с.

[2] Андрейчиков А. В. Интеллектуальные информационные системы : учебник / А. В. Андрейчиков, О. Н. Андрейчикова. — М. : Финансы и статистика, 2006. — 424 с.

[3] Бардзинь Я. М. Некоторые правила индуктивного вывода и их применение // Семантика и информатика. — 1982. — Вып. 19. — С. 59–89.

[4] Достоверный и правдоподобный вывод в интеллектуальных системах / В. Н. Вагин [и др.] ; под ред. В. Н. Вагина и Д. А. Поспелова. — М. : Физматлит, 2004. — 704 с.

[5] Змитрович А. И. Интеллектуальные информационные системы / А. И. Змитрович. — Минск: НТООО «ТетраСистемс», 1997. — 368 с.

[6] Искусственный интеллект : справочник: в 3 кн. / под ред. Д. А. Поспелова. — М. : Радио и связь, 1990.

[7] Миль Д. С. Система логики силлогистической и индуктивной : пер. с англ. / Д. С. Миль. — М. : Книжное дело, 1990.

[8] Михенкова М. А. Правдоподобные рассуждения с информацией о ситуации / М. А. Михенкова, В. К. Финн // Труды VII Национальной конференции по искусственному интеллекту с международным участием. — М. : Физматлит, 2000. — С. 192–198

[9] Павлов С. Н. Интеллектуальные информационные системы : учеб. пособие / С. Н. Павлов. — Томск: Томский межвузовский центр дистанционного образования, 2004. — 328 с.

[10] Поспелов Д. А. Моделирование рассуждений. Опыт анализа мыслительных актов / Д. А. Поспелов. — М. : Радио и связь, 1989. — 184 с.

[11] Страбыкин Д. А. Логический вывод в системах обработки знаний / Д. А. Страбыкин. — СПб.: СПГЭТУ, 1998. — 164 с.

[12] Финн В. К Правдоподобные рассуждения в интеллектуальных системах типа ДСМ // Итоги науки и техники. Сер. Информатика. — М. : ВИНИТИ. — 1991. — Т. 15 : Интеллектуальные информационные системы. — С. 54–101

[13] Чень Ч. Математическая логика и автоматическое доказательство теорем : пер. с англ. / Ч. Чень, Р. Ли. — М. : Наука, 1983. — 358 с.

Глава 4

НЕОПРЕДЕЛЕННОСТЬ ЗНАНИЙ И СПОСОБЫ ИХ ОБРАБОТКИ

4.1 Виды неопределенности описания задачи

При формировании задачи происходит отображение реальной задачи на некоторый формальный язык, а в общем случае — на профессиональный язык ЛПР, который представляет модель реального объекта.

Пусть $O_З$ — множество объектов отображаемой задачи, под которыми понимаются ее элементы и их взаимосвязи;

$O_Я$ — множество объектов языка (понятия, отношения, имена и т. д.).

Рассмотрим следующие отображения $\Phi: O_З \to O_Я$. В случае, когда отображение Φ устанавливает взаимно однозначное соответствие элементов множеств $O_З$ и $O_Я$, имеет место задача в условиях определенности. Пусть $O_1 \in O_З, O_2 \in O_Я$ и выполняются хотя бы одно из условий

$$(\exists O_1)\,|\Phi(O_1)| > 1; \qquad (\exists O_2)\,\left|\Phi^{-1}(O_2)\right| > 1;$$
$$(\exists O_1)\Phi(O_1) = \varnothing; \qquad (\exists O_2)\Phi^{-1}(O_2) = \varnothing,$$

где $|X|$ — мощность множества X; $\varnothing$ — пустое множество.

Тогда имеем задачу в условиях неопределенности и соответственно явления синонимии, полисемии, недостаточности или избыточности языка описания. Наиболее важные для задачи виды неопределенности описания можно представить с помощью дерева (рис. 4.1).

Первый уровень данного дерева образован терминами, качественно характеризующими количество отсутствующей информации об элементах задачи. Подчеркнем, что расстановка терминов на дереве условна. Здесь более существенна структура.

В ситуации неизвестности информация о задаче практически отсутствует (начальная стадия изучения задачи). В процессе сбора информации на определенном этапе может оказаться, что:

- собрана еще не вся возможная (неполнота) или не вся необходимая (недостаточность) информация;
- для некоторых элементов определены не их точные описания, а лишь множества, которым эти описания принадлежат (недоопределенность);
- ряд элементов задачи временно описан лишь по аналогии с уже решавшимися задачами, имеется лишь «замещающее» описание (неадекватность).

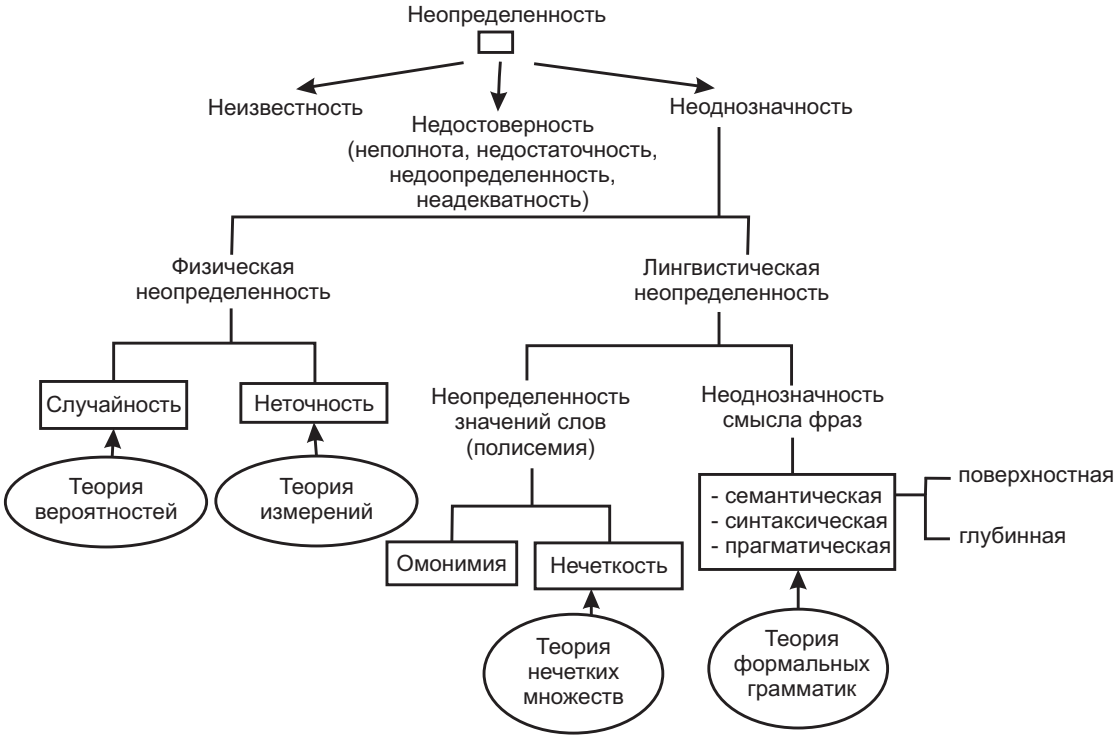


Рис. 4.1 – Неопределенность описания задач.

Важно, что наличие данных видов неопределенности (недостоверности) связанно либо с тем, что процесс сбора информации временно приостановлен, либо с нехваткой ресурсов, выделенных для сбора информации.

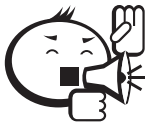
Дальнейшее изучение может привести либо к ситуации определенности, в которой все элементы описаны однозначно (классический пример — транспортная задача линейного программирования), либо к ситуации неоднозначности. Для последней предполагается, что вся возможная информация о задаче собрана, но полностью определенное описание не получено и не может быть получено. Второй уровень дерева описывает источники (причины) возможной неоднозначности описания, которыми являются внешняя среда (физическая неопределенность) и используемый ЛПР профессиональный язык (лингвистическая неопределенность).

Физическая неопределенность описывает неопределенность объектов реального мира с точки зрения наблюдателя. Так, неточность связанна с возможностями измерительного оборудования, т. е. с неточностью измерений вполне определенной величины, выполняемых физическими приборами. С объектами, измеряемыми в различных шкалах, мы должны работать по-разному. Например, для ранговых или номинальных шкал арифметические операции (включая вычисление средних значений) не имеют смысла. Изучение подобных вопросов составляет предмет исследования теории измерений.

Физическая неопределенность может быть связанна и с наличием во внешней среде нескольких возможностей, каждая из которых случайным образом становится действительностью (ситуация случайности, или стохастической неопределенности).

Теория вероятностей представляет, пожалуй, наиболее развитую теорию, ориентированную на обработку неопределенности. Однако эта теория базируется на ряде предположений и гипотез, без проверки которых для данной конкретной проблемы мы не можем гарантировать адекватность выводов, полученных в рамках анализа модели, реальным объектам или процессам. Примерами таких требований могут быть:

- повторяемость событий;
- гарантии того, что наблюдаемые эффекты могут быть перенесены на все объекты или события данного типа (генеральную совокупность);
- независимость событий и т. д.



Лингвистическая неопределенность связанна с использованием естественного языка (в частом случае — профессионального языка ЛПР) для описания задачи. Эта неопределенность обуславливается необходимостью оперировать конечным числом слов и ограниченным числом структур фраз (предложений, абзацев, текстов) для описания за конечное время бесконечного множества разнообразных ситуаций, возникающих в процессе принятия решений.

Лингвистическая неопределенность порождается, с одной стороны, множественностью значений слов (понятий и отношений) языка, которую условно назовем полисемией, а с другой стороны, неоднозначностью смысла фраз.

Для наших целей достаточно выделить два вида полисемии: омонимию и нечеткость. Если отображаемые одним и тем же словом объекты задач существенно различны то соответствующую ситуацию отнесем к омонимии. Например: коса — вид побережья, сельскохозяйственный инструмент, вид прически. Если же эти объекты сходны, то ситуацию отнесем к нечеткости. Например, небольшой запас горючего

на складе, множество чисел, значительно больше ста. Теория нечетких множеств есть некоторый аппарат формализации одного из видов неопределенности, возникающей при моделировании (в широком смысле этого слова, не только математическом) реальных объектов. Нечеткость возникает всегда, когда мы используем слова естественного языка для описания объекта. Последнее возникает всегда, когда мы пытаемся применять информационные технологии в «нетрадиционных» или «гуманитарных» областях, таких, как медицина, экономика, управление, социология и др. В рамках теории нечетких множеств разработан аппарат формализации содержательно значимых понятий. Примеры: высокий уровень безопасности, устойчивая ситуация, нормальная температура и т. д.

Рассматривая источники неоднозначности смысла фраз, можно выделить синтаксическую, семантическую и прагматическую неоднозначности. В первом случае уточнение синтаксиса позволяет понять смысл фразы. Пример: «казнить, нельзя помиловать — казнить нельзя, помиловать»; во втором случае при поверхностной семантической неопределенности смысла фраз отдельные слова понятны, но не ясен смысл фразы. Пример: «голубые зеленые мысли яростно спят», при неоднозначности глубинной структуры (семантической неопределенности) не понятны и все отдельные слова или существует два или несколько прочтений. Пример: «глокая куздра штеко будланула бокра и курчатит бокренка» или «Шалтай-болтай свалился во сне». В последнем предложении два прочтения: или Шалтай-болтай действительно свалился во время сна, или же персонажу сказки приснилось, что он свалился; в третьем случае прагматическая неоднозначность проявляется (связывается) с неправильным использованием местоимений или каких-либо других отсылочных слов. Например: «Он ударил палкой по табуретке и сломал ее». Таким образом, возникает подмена понятия — сломал табуретку, а не палку.

Таким образом, прагматическая неопределенность связанна с неоднозначностью использования синтаксических и семантических понятий информации для достижения целей деятельности. Возникающие при понимании текстов проблемы детально анализируются в [6–11]. Итак, неопределенность смысла фраз исследуются теорией формальных грамматик.

Заканчивая рассмотрение видов неопределенности, необходимо отметить, что данные виды неопределенности могут накладываться друг на друга. Например, учет физической неопределенности может усложниться появлением лингвистической неопределенности в описании вероятностного распределения.

4.2 Особенности данных и знаний

При представлении знаний в БЗ различают, как правило, свойства данных и знаний по содержанию и истинности информации. Обычно знания в интеллектуальных системах неполны, противоречивы, немонотонны, неточны, неопределенны и нечетки. Остановимся на этих особенностях знаний [1].

Полнота обычно формулируется следующим образом: для множества формул с заданными свойствами исходная система аксиом и правил вывода должна обеспечить вывод всех формул, входящих в это множество.

Неполнота информации, которую приходится использовать, например, в экспертных системах машинного обучения, требует специальных подходов, отлич-

ных от методов классической логики. Это могут быть методы статистики, методы нечеткой логики, основанные на нечетких множествах Л. Заде, теория приближенных множеств (rough sets theory), предложенная в начале 80-х годов математиком Павлаком как новое средство работы с нечеткостями. Главное преимущество этой теории в том, что она не требует никакой предварительной или дополнительной информации о данных (информации о вероятностях или о степени принадлежности элемента множеству). Эта теория может быть использована для решения задач извлечения знаний из баз данных.

Непротиворечивость — свойство, которое сводится к тому, что исходная система аксиом и правил вывода не должна давать возможность выводить формулы, не принадлежащие заданному множеству формул с выбранными свойствами. Например, полные системы аксиом и правил вывода в классическом исчислении предикатов первого порядка позволяют получать любую общезначимую формулу из множества общезначимых формул и не дают возможности получить какие-либо формулы, не обладающие этим свойством. Однако на практике мы имеем дело со слабоформализованными данными и, следовательно, трудностями формулировки формул и ответов на запросы в вопрос-ответных системах. В отличие от простого добавления информации, как в базе данных, при добавлении новых знаний возникает опасность получения противоречивых выводов, т. е. выводы, полученные с использованием новых знаний, могут опровергать те, что были получены ранее. Еще хуже, если новые знания будут находиться в противоречии со старыми. Почти все экспертные системы первого поколения были основаны на модели закрытого мира с применением аппарата формальной логики. Модель закрытого мира предполагает заполнение БЗ только истинными понятиями, а остальные считаются ложными. Эта модель работает, на ней базируется язык PROLOG, но она должна быть полной, и выводы должны быть монотонными.

Монотонность — свойство логических выводов. Для полного набора знаний справедливость полученных выводов не нарушается с добавлением новых фактов, т. е. если $A_1, A_2, \ldots, A_n \vdash B_1$, то $A_1, A_2, \ldots, A_n, \{F\} \vdash B_1$, где $\{F\}$ — множество добавочных утверждений не отменяет ранее выведенного утверждения B_1.

В качестве средств формальной обработки неполных знаний, для которых необходимы немонотонные выводы, разрабатываются методы немонотонной логики: немонотонная логика Макдермотта и Доула, в которой вводятся условные логические операции, логика умолчания о замкнутости мира Рейтера, немонотонная логика Маккарти и т. д.

Неточность — относится к содержанию информации (или значению сущности) и наряду с неполнотой и противоречивостью должна обязательно учитываться при представлении знаний в БЗ. Возникновение неточных знаний связанно:

- с ошибками в данных;
- ошибками в правилах вывода;
- использованием неточных моделей.

Неточная информация может быть как непротиворечивой, так и противоречивой. Так, например, возраст студента Иванова, записанный в БЗ, равен 32 годам, хотя ему на самом деле 23. Этот пример непротиворечивой, хотя и не точной информации. Может быть и другая ситуация, когда по ошибке в БЗ ему записали

123 года, что противоречит действительности, так как возраст людей, как правило, колеблется от 0 до 100 лет.

К неточности будем относить также величины, значения которых могут быть получены с ограниченной точностью, не превышающей некоторый порог, определенный природой соответствующих параметров. Очевидно, что практически все реальные оценки являются неточными и сама оценка неточности также является неточной. Примеры неточности данных встречаются при измерении физических величин. В зависимости от степени точности измерительного прибора, от психологического состояния и здоровья человека, производящего измерения, получаемое значение величины колеблется в некотором интервале. Поэтому для представления неточности данных мы можем использовать интервал значений вместе с оценкой точности в качестве меры доверия к каждому значению.

Основной подход, связанный с теорией приближенных множеств, основан на идее классификации. Эта теория помогает решить проблему неточных знаний. Неточные знания или понятия могут быть определенны приближенно в рамках заданного обучающего множества, если использовать понятия верхнего и нижнего приближений и меры точности аппроксимации.

Разработаны алгоритм RS1 и модифицированный RS2 построения продукционных правил, основанные на теории приближенных множеств и обработки объектов с неопределенностью и различными степенями важности в информационной системе. Последний алгоритм позволяет обрабатывать информацию с неопределенностью двух видов: когда отсутствуют значения некоторых атрибутов у объектов и когда объекты представлены с некоторой степенью уверенности.

Неопределенность относится не к содержанию информации, а к ее истинности, понимаемой в смысле соответствия реальной действительности (степени уверенности знания). Иногда в литературе вводится термин «ненадежные знания», который определяется таким образом, что знание не может быть истинным или ложным, т. е. 0 или 1. Каждый факт реального мира связан с определенностью информации, которая указывает на степень этой уверенности.

Для понятий «определенность» и «уверенность» чаще всего используются количественные меры. Основная идея такой меры заключается во введении функции неопределенности $unc(p)$, понимаемой как определенность того, что высказывание p, содержащееся в БЗ, истинно, т. е. говорят, что утверждение p более определено, чем q, если $unc(p) \geqslant unc(q)$.

Традиционным подходом для представления неопределенности является теория Г. Шейфера, которая дает возможность с единых позиций подойти к вопросам формального представления неопределенности в математических моделях и тем самым приобретает большое значение как в фундаментальном, так и прикладном аспектах. Эта теория еще нуждается в глубокой проработке вопросов ее практического применения, тогда как теория возможностей и тем более теория вероятностей используются в инженерной практике.

Байесовский метод. При байесовском подходе степень достоверности каждого из фактов базы знаний оценивается вероятностью, которая принимает значения от нуля до единицы. Вероятности исходных фактов определяются либо методом статистических испытаний, либо опросом экспертов. Вероятность заключения определяют на основе правила Байеса для вычисления апостериорной условной

вероятности $\mathsf{P}(H|E)$ события (гипотезы) H при условии, что произошло событие (свидетельство) E [3]

$$\mathsf{P}(H|E) = \frac{\mathsf{P}(E,H)}{\mathsf{P}(E)},$$

где $\mathsf{P}(E)$ — безусловная (априорная) вероятность свидетельства E;

$\mathsf{P}(E,H)$ — совместная вероятность свидетельства E и гипотезы H.

Совместная вероятность $\mathsf{P}(E,H)$ равна произведению безусловной вероятности гипотезы $\mathsf{P}(H)$ на условную вероятность того, что свидетельство (факт) E имеет место, если наблюдается гипотеза H:

$$\mathsf{P}(E,H) = \mathsf{P}(H) \cdot \mathsf{P}(E|H).$$

Отсюда следует, что апостериорную вероятность $\mathsf{P}(H,E)$ можно вычислить с помощью формулы

$$\mathsf{P}(H|E) = \frac{\mathsf{P}(H) \cdot \mathsf{P}(E|H)}{\mathsf{P}(E)}. \tag{4.1}$$

Уточним практическое применение формулы (4.1) на простом примере. Пусть H обозначает некоторое заболевание, а E — симптом. Тогда априорная вероятность заболевания H может быть определена по формуле $\mathsf{P}(H) = N_H/N$, где N_H — количество жителей некоторого региона, имеющих заболевание H; N — количество всех жителей региона. Вероятность $\mathsf{P}(E)$ определяется аналогично $\mathsf{P}(E) = N_E/N$, где N_E — количество жителей, у которых наблюдается симптом E. Обычно значения вероятностей $\mathsf{P}(H)$ и $\mathsf{P}(E)$ выясняют у экспертов.

Вероятность $\mathsf{P}(E|H)$ соответствует наличию симптома E у больного с заболеванием H. Ее значение также определяется методом опроса экспертов.

Формула (4.1) справедлива для случая одного свидетельства E и одной гипотезы H. Она позволяет пересчитать значение вероятности гипотезы H в том случае, когда обнаружено свидетельство E в ее пользу, т. е. получать на основе априорной вероятности $\mathsf{P}(H)$ значение $\mathsf{P}(H|E)$ апостериорной вероятности. Если рассматривается полная группа несовместимых гипотез $H_1, H_2, \ldots, H_n$ с априорными вероятностями $\mathsf{P}(H_1), \mathsf{P}(H_2), \ldots, \mathsf{P}(H_n)$, то апостериорную вероятность каждой из гипотез при реализации свидетельства E вычисляют с помощью формулы

$$\mathsf{P}(H_i|E) = \frac{\mathsf{P}(E|H_i) \cdot \mathsf{P}(H_i)}{\sum\limits_{k=1}^{n} \mathsf{P}(E|H_k) \cdot \mathsf{P}(H_k)}, \tag{4.2}$$

которую называют теоремой гипотез Байеса.

Формула (4.2) позволяет упростить вычисление гипотезы H при реализации свидетельства E. Так, если рассматривать две несовместимые гипотезы H и $\sim H$ («не H», $\neg H$, $\bar{H}$), то

$$\mathsf{P}(H|E) = \frac{\mathsf{P}(E|H) \cdot \mathsf{P}(H)}{\mathsf{P}(E|H) \cdot \mathsf{P}(H) + \mathsf{P}(E| \sim H) \cdot \big(1 - \mathsf{P}(H)\big)}, \tag{4.3}$$

где $\mathsf{P}(E| \sim H)$ — вероятность свидетельства E при условии, что гипотеза H не наблюдается;

$1 - \mathsf{P}(H) = \mathsf{P}(\sim H)$ — априорная вероятность невыполнения гипотезы H. В формуле (4.3), в отличие от (4.1) , отсутствует априорная вероятность $\mathsf{P}(E)$, которая неудобна с точки зрения экспертной оценки.

Введем в рассмотрение отношение

$$D_E = \frac{\mathsf{P}(E|H)}{\mathsf{P}(E| \sim H)},$$

которые называют *отношением правдоподобия*. Оно характеризует отношение вероятности получения свидетельства E при условии, что гипотеза H верна, к вероятности получения этого же свидетельства при условии, что гипотеза H не верна. Поделив числитель и знаменатель (4.3) на $\mathsf{P}(E| \sim H)$, получим:

$$\frac{\mathsf{P}(H|E)}{1 - \mathsf{P}(H|E)} = D_E \frac{\mathsf{P}(H)}{1 - \mathsf{P}(H)}. \tag{4.4}$$

Отношения, записанные в выражении, называются шансами. Шансы — это иная шкала для представления вероятности. Так отношение

$$O(H) = \frac{\mathsf{P}(H)}{1 - \mathsf{P}(H)}$$

представляет априорные шансы в пользу гипотезы H, а

$$O(H|E) = \frac{(H|E)}{\big(1 - \mathsf{P}(H|E)\big)} —$$

апостериорные шансы. Например, если $\mathsf{P}(H) = 0.3$, то $O(H) = 3/7$ т. е. три случая «за» и семь «против».

Из (4.4) следует, что

$$O(H|E) = D_E \cdot O(H) \tag{4.5}$$

Таким образом, апостериорные шансы очень просто вычисляются через априорные шансы. Для этого необходимо знать отношение правдоподобия свидетельства E. Считается, что использование шансов в соответствии с формулой (4.5) для оценки правдоподобности гипотез проще, чем непосредственное вычисление вероятностей по формуле (4.4). Это объясняется следующим образом:

- для многих пользователей использование шкалы шансов выглядит проще;
- шансы обозначают целыми числами, а их проще вводить при ответах на вопросы системы.

Отметим, что можно получить формулу, симметричную (4.5), позволяющую вычислять апостериорные шансы в пользу гипотезы H, если заданно, что E заведомо ложно.

$$O(H| \sim E) = N_E \cdot O(H),$$

где N_E — отношение правдоподобия

$$\frac{\mathsf{P}(\sim E|H)}{\mathsf{P}(\sim E| \sim H)}$$

Следовательно, шансы в пользу некоторой гипотезы H можно вычислять, опираясь либо на истинность свидетельства E, либо на его ложность. Такой подход используется в экспертной системе PROSPECTOR, применяемой в геологии. При этом отношение правдоподобия D_E и N_E называют соответственно фактором достаточности и фактором необходимости. Факторы D_E и N_E связанны друг с другом простым соотношением

$$N_E = \frac{1 - D_E \cdot \mathsf{P}(E|H)}{1 - \mathsf{P}(E| \sim H)},$$

которое позволяет установить диапазон изменения их значений.

Можно показать, что значения D_E принадлежат диапазону $[1, \infty)$, а значения N_E — диапазону $[0, 1]$. С каждым правилом в системе PROSPECTOR связывают факторы D_E и N_E, значения которым могут назначаться независимо. Это иногда приводит к противоречиям. Чтобы разрешить такие противоречия, фактор D_E используется, когда свидетельство E истинно, а фактор N_E — когда ложно.

На практике гипотеза H может подтверждаться не одним свидетельством, а несколькими. Формула (4.5) легко обобщается на случай нескольких свидетельств. Например, если реализовались два свидетельства E_1 и E_2 в пользу гипотезы H, то ее апостериорные шансы можно вычислить по формуле

$$O(H|E_1, E_2) = D_{E_2} \cdot D_{E_1} \cdot O(H) \text{ или } O(H|E_1, E_2) = D_{E_1} \cdot O(H|E_1).$$

Так как обычно $D_E > 1$, то получение новых свидетельств в пользу гипотезы H позволяет увеличить ее правдоподобие.

В рассмотренных выше формулах предполагалось, что свидетельства $E_1, E_2, \ldots, E_n$ полностью определены, т. е. подтверждаются с вероятностью 1. Однако на практике факты, используемые в системе, могут подтверждаться с меньшей вероятностью. Это может происходить из-за неопределенных ответов пользователя, а также в результате логического вывода, когда заключения одних правил выступают в роли факторов (свидетельств) других правил. В этом случае необходимо при вычислении правдоподобия той или иной гипотезы учитывать ненадежность фактов. Пусть известно (например, из предыдущих выводов), что свидетельство E подтверждается с вероятностью $\mathsf{P}(E|E')$, где через E' обозначено некоторое свидетельство, подтверждающее E. Тогда апостериорную вероятность гипотезы H можно вычислить по формуле

$$\mathsf{P}(H|E') = \mathsf{P}(E|E') \cdot \mathsf{P}(H|E) + \big(1 - \mathsf{P}(E|E')\big) \cdot \mathsf{P}(H| \sim E).$$

Таким образом, из формулы следует:

1) если свидетельство E подтверждается с вероятностью $\mathsf{P}(E|E') = 1$, то $\mathsf{P}(H|E') = \mathsf{P}(H, E)$;
2) если $\mathsf{P}(E|E') = 0$ т. е. свидетельство E не подтверждается, то $\mathsf{P}(H|E') = \mathsf{P}(H, \sim E)$.

Кроме того, если $\mathsf{P}(E|E') = \mathsf{P}(E)$, т. е. свидетельство подтверждается с априорной вероятностью, то значение вероятности не должно измениться: $\mathsf{P}(H|E') = \mathsf{P}(H)$

Таким образом, значения $\mathsf{P}(H|E')$ лежат в диапазоне от $\mathsf{P}(H| \sim E)$ до $\mathsf{P}(H)$, если $0 \leqslant \mathsf{P}(E|E') \leqslant \mathsf{P}(E)$, и в диапазоне от $\mathsf{P}(H)$ до $\mathsf{P}(H|E)$ если $\mathsf{P}(E) \leqslant \mathsf{P}(E|E') \leqslant 1$.

Выполнив кусочно-линейную аппроксимацию зависимости $\mathsf{P}(H|E')$ от $\mathsf{P}(E|E')$, получим следующие формулы [54]:

$$\mathsf{P}(H|E') = \mathsf{P}(H|\sim E) + \frac{\mathsf{P}(H) - \mathsf{P}(H|\sim E)}{\mathsf{P}(E)} \cdot \mathsf{P}(E|E'),$$

если $0 \leqslant \mathsf{P}(E|E') \leqslant \mathsf{P}(E)$;

$$\mathsf{P}(H|E') = \mathsf{P}(H) + \frac{\mathsf{P}(H|E) - \mathsf{P}(H)}{1 - \mathsf{P}(H)} \cdot \big(\mathsf{P}(E|E') - \mathsf{P}(E)\big),$$

если $\mathsf{P}(E) \leqslant \mathsf{P}(E|E') \leqslant 1$.

Данные формулы позволяют выполнить коррекцию значения $\mathsf{P}(H, E)$ в зависимости от вероятности подтверждения свидетельства E в экспертной системе PROSPECTOR. При заполнении базы знаний пользователям разрешается вместо вероятности $\mathsf{P}(E|E')$ задавать коэффициенты (факторы) уверенности, лежащие в диапазоне от −5 до +5. В дальнейшем значения этих коэффициентов пересчитываются в вероятности $\mathsf{P}(E|E')$.

Значение апостериорной вероятности $\mathsf{P}(H|E')$, вычисленное с учетом вероятности подтверждения свидетельства E, определяет эффективное отношение правдоподобия D_E.

$$D'_E = \frac{\mathsf{P}(H|E')}{1 - \mathsf{P}(H|E')} \cdot \frac{1 - \mathsf{P}(H)}{\mathsf{P}(H)}.$$

Данное отношение позволяет выполнять коррекцию апостериорных шансов в пользу гипотезы H [10]:

$$O(H|E') = D_{E'} \cdot O(H).$$

Правила логического вывода допускают объединение свидетельств $E_1, E_2, \ldots, E_n$ с помощью логических связок И, ИЛИ, НЕ. В такой ситуации обработка каждого правила выполняется с учетом принципов нечеткой логики. Для свидетельств, связанных логической операцией И, выбирается минимальная из вероятностей. Для логической операции ИЛИ берется максимальная из вероятностей. Операция логического отрицания НЕ приводит к вычислению обратной вероятности.

При использовании байесовского подхода для обработки неточных знаний возникают две основные проблемы. Во-первых, должны быть известны все априорные условные вероятности свидетельств, а также априорные вероятности гипотез (или соответствующие отношения правдоподобия) и шансы. Во-вторых из условий теоремы Байеса следует, что все гипотезы, рассматриваемые системой, должны быть несовместны, а вероятности $\mathsf{P}(E|H_i)$ и $\mathsf{P}(E)$ — независимы. В ряде областей (например, в медицине) последнее требование не выполняется.

Кроме того, при введении новой гипотезы необходимо заново пересчитывать таблицы вероятностей, что также ограничивает применимость байесовского подхода. Тем не менее он применяется достаточно широко в системах искусственного интеллекта, благодаря хорошему теоретическому фундаменту.

Пример

Рассмотрим использование условной вероятности на примере правил, описывающих экспертную систему фондовой биржи (предположим, что эти правила относятся только к фондовой бирже) [5].

10 ЕСЛИ ПРОЦЕНТНЫЕ СТАВКИ = ПАДАЮТ, ТО УРОВЕНЬ ЦЕН = РАСТЕТ
20 ЕСЛИ ПРОЦЕНТНЫЕ СТАВКИ = РАСТУТ, ТО УРОВЕНЬ ЦЕН = ПАДАЕТ
30 ЕСЛИ ВАЛЮТНЫЙ КУРС ДОЛЛАРА = ПАДАЕТ, ТО ПРОЦЕНТНЫЕ СТАВКИ = РАСТУТ
40 ЕСЛИ ВАЛЮТНЫЙ КУРС ДОЛЛАРА = РАСТЕТ, ТО ПРОЦЕНТНЫЕ СТАВКИ = ПАДАЮТ

Надо определить вероятность повышения уровня цен. Цель примера не в описании реальной ситуации, а просто в иллюстрации подхода к решению задачи. Система, реализующая обратные рассуждения в части ТО правил, будет искать вывод УРОВЕНЬ ЦЕН = РАСТЕТ. Подойдет правило 10: УРОВЕНЬ ЦЕН = РАСТЕТ при условии, что ПРОЦЕНТНЫЕ СТАВКИ = ПАДАЮТ. Используя уравнение условной вероятности, можно оценить эти условия. Перейдем к переменным и заменим Н на STOK = РАСТЕТ и Е на INT = ПАДАЕТ, в результате получим уравнение:

P (STOK=РАСТЕТ) = P (STOK=РАСТЕТ | INT = ПАДАЕТ) * P (INT = ПАДАЕТ) +
+ P (STOK = РАСТЕТ | INT = НЕ ПАДАЕТ) * P (INT = НЕ ПАДАЕТ)

Для того чтобы определить, присвоено ли переменной INT значение ПАДАЕТ, надо вернуться к правилу 40:

40 ЕСЛИ DOLLAR = РАСТЕТ, ТО INT = ПАДАЕТ

Правило 40 преобразуется в уравнение:

P (INT = ПАДАЕТ) = P (INT = ПАДАЕТ | DOLLAR = РАСТЕТ) * P (DOLLAR =
= РАСТЕТ) + P (INT = ПАДАЕТ | DOLLAR = НЕ РАСТЕТ) * P (DOLLAR = НЕ
РАСТЕТ)

Поскольку ни в одном из правил в части ТО нет переменной DOLLAR, т. е. значение вероятности для нее определить нельзя, это значение должно быть введено пользователем. По этой же причине условную вероятность также должен задать пользователь (она не входит в часть ТО правил). Давайте установим вероятности появления некоторых событий, исходя из собственных обоснованных соображений:

P (DOLLAR = РАСТЕТ) = 0.6

Согласно теории вероятностей сумма вероятностей появления и не появления какого-либо события равна 1. В дальнейшем это свойство вероятностей будет часто

использоваться. Исходя из сказанного можно записать:

P (DOLLAR = НЕ РАСТЕТ) = 1 – P (DOLLAR = РАСТЕТ) = 1 – 0.6 = 0.4

Присвоим значения всем условным вероятностям:

P (INT = ПАДАЕТ | DOLLAR = РАСТЕТ) = 0.8
P (INT = ПАДАЕТ | DOLLAR = НЕ РАСТЕТ)= 0.1

Отметим, что сумма условных вероятностей для противоположностей не равняется 1. Противоположными в условной вероятности будут события DOLLAR = = РАСТЕТ и DOLLAR = НЕ РАСТЕТ. Подставив присвоенные значения в уравнение, получим:

P (INT = ПАДАЕТ) = 0.8 * 0.6 + 0.1 * 0.4 = 0.52

Из основного свойства вероятностей находим:

P (INT = НЕ ПАДАЕТ) = 1 – 0.52 = 0.48

Для того чтобы найти P (STOK=РАСТЕТ), пользователь должен задать значения условных вероятностей:

P (STOK = РАСТЕТ | INT=ПАДАЕТ) = 0.85
P (STOK = РАСТЕТ | INT = НЕ ПАДАЕТ) = 0.1

Вероятность P (STOK = РАСТЕТ) можно вычислить по первому уравнению:

P (STOK = РАСТЕТ) = 0.85 * 0.52 + 0.1 * 0.48 = 0.49 или 49 %

Получив все значения вероятностей, пользователь может определить свою политику на бирже. Интересно отметить, что в данном примере вероятность повышения уровня цен меньше 50 %. Это прямо обусловлено выбором вероятностей. Если пользователь повысит условную вероятность роста уровня цен на бирже при условии, что процентные ставки не будут падать, т. е. P (STOK = РАСТЕТ | INT = = НЕ ПАДАЕТ), повысится вероятность P (STOK = РАСТЕТ).

. . . . . .

Метод коэффициентов уверенности. Этот метод был впервые применен в ЭС MYCIN. В отличие от Байесовского подхода, который использует теорию вероятностей для подтверждения гипотез, метод системы MYCIN базируется на эвристических соображениях, которые были позаимствованы из практического опыта работы экспертов. Когда эксперт оценивает степень достоверности некоторого вывода, он использует такие понятия, как «точно», «весьма вероятно», «возможно», «ничего нельзя сказать» и т. д. Разработчики системы MYCIN решили отобразить эти размытые понятия на шкалу коэффициентов уверенности, изменяющихся в диапазоне от –1 до +1. Для этого были введены две оценки: *MB* и *MD*. Оценка

MB отражает степень истинности некоторого фактора (свидетельства) и принимает значение от 0 до +1. Оценка MD соответствует степени ложности некоторого фактора и принимает значения в диапазоне от −1 до 0. Коэффициент уверенности факта, обозначаемый CF, определяется как разность оценок MB и MD:

$$CF = MB - MD.$$

Здесь $0 < MB < 1$ (при $MD = 0$) и $-1 < MD < 0$ (при $MB = 0$). Если коэффициент уверенности принимает значение, равное +1, то факт считается истинным. Если $CF = -1$, то факт ложный. Шкала измерения значений коэффициентов уверенности изображена на рис. 4.2.

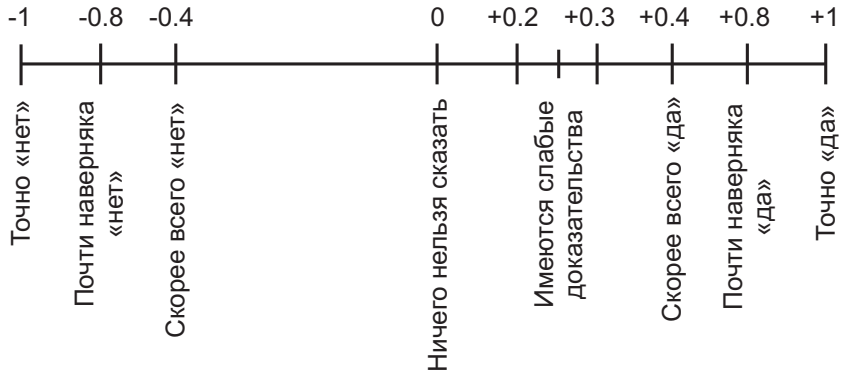


Рис. 4.2 – Шкала коэффициентов уверенности.

В ходе логического вывода над фактами, составляющими предпосылки правил, выполняются логические операции. В результате этого образуются сложные высказывания, коэффициенты уверенности которых вычисляются по следующим правилам:

1) при логической связи И между фактами P_1 и P_2

$$CF(P_1 \wedge P_2) = \min\big(CF(P_1), CF(P_2)\big);$$

2) при логической связи ИЛИ между фактами P_1 и P_2

$$CF(P_1 \vee P_2) = \max\big(CF(P_1), CF(P_2)\big)$$

В системе MYCIN коэффициенты уверенности приписываются не только фактам, но и правилам. Таким способом обеспечивается учет ненадежности правил, которые часто формируются на основе эвристических соображений. Обозначим коэффициент уверенности правила через CF_R. Коэффициент CF_R соответствует степени истинности заключения правила при истинных предпосылках. Ели предпосылки характеризуются коэффициентом уверенности $CF_{\text{ПРЕД}} \neq 1$, то коэффициент уверенности заключения $CF_{\text{ЗАКЛ}}$ вычисляется по формуле

$$CF_{\text{ЗАКЛ}} = CF_{\text{ПРЕД}} \cdot CF_R$$

В процессе вывода одно и то же заключение может подтверждаться различными правилами, каждое из которых приписывает заключению свой коэффициент

уверенности. Очевидно, что коэффициент уверенности, подтверждаемы несколькими правилами, должен увеличиваться. Чем больше будет иметься подтверждений в пользу некоторого заключения, тем ближе к 1 должен быть его коэффициент уверенности. В общем случае комбинация свидетельств в поддержку некоторого заключения выполняется по следующим формулам, используемым в системе EMYCIN:

$$CF = CF_1 + CF_2 - CF_1 \cdot CF_2, \text{ если } CF_1 > 0, CF_2 > 0;$$
$$CF = CF_1 + CF_2 + CF_1 \cdot CF_2, \text{ если } CF_1 < 0, CF_2 < 0;$$
$$CF = \frac{CF_1 + CF_2}{1 - \min(|CF_1|, |CF_2|)}, \text{ если } CF_1 \cdot CF_2 \leqslant 0, CF_1 \neq \pm 1, CF_2 \neq \pm 1;$$
$$\text{Если } CF_1 \neq \pm 1 \text{ и } CF_2 \neq \pm 1, \text{ то } CF = 1.$$

Коэффициент уверенности заключений, формируемых тремя или более правилами, можно вывести последовательно, применяя вышеприведенные формулы.

Метод коэффициентов уверенности благодаря своей простоте находит широкое применение во многих системах, поддерживающих вывод на ненадежных знаниях. Недостатки метода связанны с отсутствием теоретического фундамента, а также сложностью подбора коэффициентов уверенности.

Разберем пример:

Сформулируем общие принципы.

1) Выбрать минимальное значение коэффициента уверенности из коэффициентов уверенности всех условий правила, разделенных логическим оператором И.
2) Если в правиле есть оператор ИЛИ, выбрать минимальное значение из коэффициентов уверенности для всех условий правила, разделенных оператором И для всех условий, связанных оператором ИЛИ.
3) Умножить выбранный коэффициент уверенности на коэффициент уверенности правила.
4) Если существует несколько правил с одинаковым логическим выводом, выбрать из всех полученных коэффициентов уверенности максимальный.

Пример

Рассмотрим два правила с одним и тем же логическим выводом С:

$$\text{ЕСЛИ } A(\text{КУ} = 0.3) \text{ И } B(\text{КУ} = 0.6), \text{ ТО } C(\text{ТО} = 0.5),$$
$$\text{ЕСЛИ } D(\text{КУ} = 0.4) \text{ И } E(\text{КУ} = 0.7), \text{ ТО } C(\text{ТО} = 0.9),$$

где КУ — коэффициент уверенности.

В приведенных правилах коэффициент уверенности для логического вывода С подсчитывается следующим образом:

$$\text{КУ} = Maximum\Big(\big(Minimum(0.3, 0.6) \times 0.5\big), \big(Minimum(0.4, 0.7) \times 0.9\big)\Big) =$$
$$= Maximum\big((0.3 \times 0.5), (0.4 \times 0.9)\big) = Maximum(0.15, 0.36) = 0.36.$$

Приведем пример с использованием логического оператора ИЛИ:
ЕСЛИ А (КУ = 0.3) И
В (КУ = 0.6) ИЛИ
D (КУ = 0.5),
ТО С (КУ = 0.4)

В этом примере коэффициент уверенности для вывода С считается так:

$$\text{КУ} = Maximum\Big(\big(Minimum(0.3, 0.6), 0.5\big) \times 0.4\Big) =$$
$$= Maximum(0.3, 0.5) \times 0.4 = 0.5 \times 0.4 = 0.2.$$

. . . . . .

Теория свидетельств Демстера-Шефера. В теории Демстера-Шефера [14] требование к условию $\mathsf{P}(q) + \mathsf{P}(\sim q) = 1$ ослаблено, и вместо него имеет место $\mathsf{P}(q)+\mathsf{P}(\sim q) \leqslant 1$. Подход, принятый в этой теории, отличается от байесовского подхода и метода коэффициентов уверенности тем, что, во-первых, здесь используется не точная оценка уверенности (вероятность, коэффициент уверенности), а интегральная оценка. Такая оценка характеризуется нижней и верхней границами, что более надежно. Во-вторых, теория свидетельств позволяет исключить взаимосвязь между неопределенностью (неполнотой знаний) и недоверием, которая свойственна байесовскому подходу.

В рамках этой теории множеству высказываний A присваивается диапазон значений $\big[Bl(a), Pl(A)\big]$, в котором находятся степени доверия каждого из высказываний. Здесь $Bl(A)$ — степень доверия к множеству высказываний, изменяющая свое значение от 0 (нет свидетельств в пользу A) до 1 (множество высказываний A истинно); $Pl(A)$ — степень правдоподобия множества высказываний A, определяемая с помощью формулы:

$$Pl(A) = 1 - Bl(\sim A).$$

Значения степени правдоподобия также лежит в диапазоне от 0 до 1. Степень правдоподобия множества высказываний A — это величина обратная степени доверия к множеству противоречивых высказываний $\sim A$. Если имеются четкие свидетельства в пользу $\sim A$, то $Bl(\sim A) = 1$ и степень правдоподобия A равна нулю, т. е. $Pl(A) = 0$. При $Pl(A) = 0$ степень доверия $Bl(A)$ тоже равна нулю.

В случае отсутствия информации в поддержку той или иной гипотезы полагается, что диапазон изменения значений степеней Bl и Pl равен $[0, 1]$. По мере получения свидетельств указанный диапазон сужается.

Неопределенность описания знаний можно формально представить с помощью некоторого универсального множества X. Его элементами могут быть, например, исходы решения или эксперимента (бросание монеты, прогноз погоды, покупка автомобиля), диапазон изменения физической величины, возможные семантические значения фразы, факты, заключения. Неопределенность состоит в том, что точно не известно, какой элемент $x \in X$ в действительности имел, имеет или будет иметь место. Конкретная интерпретация неопределенности может быть самая различная в зависимости от физической природы универсального множества X: неабсолютная истинность некоторого высказывания, отличная от единицы вероятность некоторого события; неполная уверенность ЛПР в своих действиях; частичная принадлежность некоторому подмножеству, которая представляет интерес при использовании нечеткой информации.

В [14] для измерения степени определенности предполагается использовать некоторую единичную массу уверенности, распределенную между элементами множества X и его подмножествами. Если при этом вся масса уверенности приходится на некоторый элемент $x_0 \in X$, то можно говорить о ситуации полной определенности.

Если же она распределена хотя бы между двумя $x_1, x_2 \in X$, то уже возникает некоторая неопределенность.

Распределение уверенности может иметь различный вид. В [14] предложена следующая геометрическая интерпретация. Так, если множество X представить в виде набора точек, то уверенность можно отобразить в виде масс, закрепленных за этими точками (рис. 4.3)

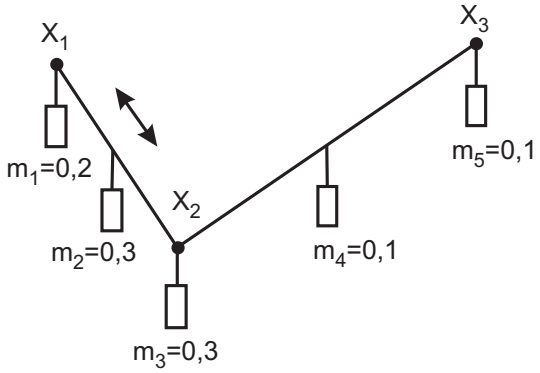


Рис. 4.3 – Геометрическая интерпретация распределения уверенности.

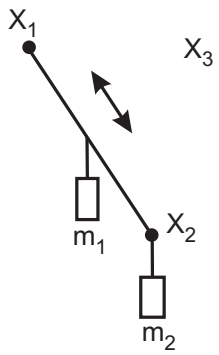


Рис. 4.4 – Геометрическая интерпретация распределения с неизвестными массами уверенности.

В простейшем случае массы закрепляются в точках жестко. На рис. 4.3 каждая из точек, составляющих множество $X = \{x_1, x_2, x_3\}$, обладает жестко закрепленной массой. Однако некоторая масса уверенности может быть не только жестко закрепленной за одной точкой, но и относиться сразу к нескольким точкам одновременно. Это соответствует массе, свободно перемещающейся между этими точками, т. е. масса относится одновременно к любой из них. Данный случай соответствует большей неопределенности, когда все множество X оказывается разбитым на классы эквивалентности (подмножества), возможно, пересекающиеся, между которыми распределяется уверенность.

Так, если мы на 90 % уверены в том, что проезжающий мимо нас на высокой скорости легковой автомобиль был марки «Жигули», то масса уверенности $m_1 = 0.9$

относится ко всем моделям «Жигули», а масса $m_2 = 0.1$ — ко всем прочим маркам легковых автомобилей, причем более детально ее распределение неизвестно.

В приведенной постановке вся неопределенность сводится, таким образом, к тому или иному способу распределения массы уверенности.

Заметим, что на рис. 4.3 все пять масс точно известны, но налицо неопределенность — неизвестно, за какими элементами множества X закреплены массы m_1, m_2. Первая из них закреплена за подмножествам $\{x_1, x_2\}$ вторая — за $\{x_2, x_3\}$.

Тем не менее в отдельных случаях определенные выводы можно сделать даже при неизвестных массах уверенности только на основании их распределения. Так, на рис. 4.4 массы m_1, m_2 неизвестны, но позволяют сделать вывод о том, что нет абсолютно никакой уверенности в x_3, тогда как определенные доводы в пользу x_1 и x_2 имеются, причем в пользу x_2 их больше из-за массы m_2, закрепленной непосредственно за x_2. Ситуация, представленная на рис. 4.4, относится к так называемому согласованному распределению уверенности. Только в подобных ситуациях можно сравнить степени уверенности в том или ином элементе множества X, не зная самих масс уверенности. Во всех остальных случаях массы уверенности должны быть известны, поскольку другие варианты равносильны полной неопределенности.

Распределением уверенности, согласно [14], называется функция вида $m: 2^x \to \to [0, 1]$, обладающая свойствами: $m(\varnothing) = 0$ и $\sum\limits_{A \subseteq X} m(A) = 1$. При этом $m(A)$ выражает степень уверенности, отнесенную к множеству A в целом; если за отдельными элементами или подмножествами множества A еще закреплены отдельные массы уверенности, то в $m(A)$ они не входят. Так, для примера, представленного на рис. 4.3:

A	$\varnothing$	$\{x_1\}$	$\{x_2\}$	$\{x_3\}$	$\{x_1, x_2\}$	$\{x_2, x_3\}$	$\{x_1, x_3\}$	X
$m(A)$	0	0.2	0.3	0.1	0.3	0.1	0	0

Общая степень уверенности в множестве $A \subseteq X$ может быть выражена с помощью так называемой функции уверенности $Bl: 2^x \to [0, 1]$, которая может быть получена из распределения уверенности: $Bl(A) = \sum\limits_{B \subseteq A} m(B)$ для всех $A \subseteq X$, т. е. суммированием масс уверенности $m(A)$, относящихся точно к множеству A, и масс уверенности $m(B)$, относящихся к его подмножествам $B \subset A$.

В [14] показано, что определенная таким образом функция удовлетворяет следующим трем свойствам:

1) $Bl(\varnothing) = 0$;
2) $Bl(X) = 1$;
3) для всех $A_1, \ldots, A_n \subseteq X, n > 0$,

$$Bl(A_1 \cup \ldots \cup A_n) \geqslant \sum_{\substack{I = \{1,2,\ldots,n\} \\ I \neq \infty}} (-1)^{|I|+1} Bl\left(\bigcap_{i \in I} A_i\right),$$

где $|I|$ — мощность множества I (индексной последовательности).

Для примера, представленного на рис. 4.3:

A	$\varnothing$	$\{x_1\}$	$\{x_2\}$	$\{x_3\}$	$\{x_1, x_2\}$	$\{x_2, x_3\}$	$\{x_1, x_3\}$	X
$Bl(A)$	0	0.2	0.3	0.1	0.8	0.5	0.3	1

Если $m(A) > 0$, то подмножество $A \subseteq X$ называется фокальным элементом распределения уверенности на множестве X. Совокупность всех фокальных элементов распределения уверенности называется его ядром. Так, для рис. 4.3 ядро распределения уверенности составляют $\{x_1\}, \{x_2\}, \{x_3\}, \{x_1, x_2\}, \{x_2, x_3\}, \{x_1, x_3\}$ а для рис. 4.4 — соответственно $\{x_2\}, \{x_1, x_2\}$.

Задав функцию уверенности, можно определить несколько вспомогательных характеристик:

- степень сомнения в множестве $A \subseteq X$

$$Dou(A) = Bl(\sim A);$$

- степень правдоподобия множества A

$$Pl(A) = 1 - Dou(A) = 1 - Bl(\sim A) = \sum_{\substack{A \cap B = \varnothing \\ B \subseteq X}} m(B).$$

В терминах массы введенные характеристики можно интерпретировать следующим образом:

$Bl(A)$ — степень уверенности в множестве A — общая масса, которая останется, если из множества X удалить все элементы, не входящие в A, вместе с закрепленными за ними массами;

$Pl(A)$ — степень правдоподобия множества A — общая масса, которую можно сдвинуть к элементам множества A.

Для примера, представленного на рис. 4.3:

A	$\varnothing$	$\{x_1\}$	$\{x_2\}$	$\{x_3\}$	$\{x_1, x_2\}$	$\{x_2, x_3\}$	$\{x_1, x_3\}$	X
$Pl(A)$	0	0.5	0.7	0.2	0.9	0.8	0.7	1

Заметим, что $Bl(A) \leqslant Pl(A)$ для всех $A \subseteq X$, т. е. $Bl(A)$ представляет нижнюю границу доверия к A, а $Pl(A)$ — верхнюю.

В [14] показано, что байесовское распределение уверенности обладает следующими свойствами, каждое из которых является его необходимым и достаточным условием:

- все фокальные элементы — точечные множества, т. е. в геометрической интерпретации все массы должны быть закреплены за точками и не имеют возможности перемещаться;
- $Bl(A) = \sum_{x_i \in X} \mathsf{P}(x_i)$ для всех $A \subseteq X$, где $\mathsf{P}(x_i) = m\big(\{x_i\}\big)$ — масса уверенности, закрепленная за элементом $x_i \in X, \mathsf{P}: X \to [0, 1], \sum_{x_i \in X} \mathsf{P}(x_i) = 1$;
- $Bl(A) = Pl(A)$.

Нетрудно заметить, что приведенная байесовская уверенность эквивалентна аксиоматическому определению вероятности. Таким образом, вероятность соответствует определенному (простейшему) распределению уверенности на множестве X. Отсюда следует, что теория вероятностей изучает один из частных видов неопределенности, когда все элементы множества X различимы — среди них нет хотя бы двух таких, к которым одновременно приложена одна и та же масса уверенности.

По аналогии с теорией вероятностей величины $Bl(A)$ и $Pl(A)$ получили соответственно названия: $\mathsf{P}_*(A)$ — нижней и $\mathsf{P}^*(A)$ — верхней вероятности множества $A \subseteq X$ в том смысле, что предполагается существование некоторой истинной вероятности $\mathsf{P}(A)$:

$$Bl(A) = P_*(A) \leqslant \mathsf{P}(A) \leqslant P^*(A) = Pl(A).$$

Тогда для байесовского распределения уверенности справедливо $\mathsf{P}(A) = \mathsf{P}_* = \mathsf{P}^*$.

Согласованное распределение уверенности и возможность. Рассмотрим еще один частный случай распределения уверенности. Распределение уверенности называется согласованным, если его фокальные элементы образуют вложенную последовательность. Следовательно, подмножества $A_1, \ldots, A_n \subseteq X$ тогда и только тогда образуют ядро согласованного распределения, когда $A_1 \subseteq \ldots \subseteq A_n$. В [14] приведено несколько необходимых и достаточных условий согласованности распределения уверенности, в частности:

$$\begin{aligned}
&\text{для всех } A, B \subseteq X && \mathsf{P}_*(A \cap B) = \min\left\{\mathsf{P}_*(A), \mathsf{P}_*(B)\right\};\\
&\text{для всех } A, B \subseteq X && \mathsf{P}^*(A \cup B) = \max\left\{\mathsf{P}^*(A), \mathsf{P}^*(B)\right\};\\
&\text{для всех } A \neq \varnothing, A \subseteq X && \mathsf{P}^*(A) = \max_{x \in A} \mathsf{P}^*(x).
\end{aligned}$$

Геометрическую интерпретацию согласованного распределения уверенности представим на рис. 4.5. Наглядно можно представить согласованное распределение уверенности на множестве X его контурной функцией $\pi: X \to [0,1]$, которая определяется через верхнюю вероятность P^* следующим образом: $\pi(x) = \mathsf{P}^*(x)$. В таком случае для всех $A \subseteq X$ $\mathsf{P}^*(A) = \max_{x \in A} \pi(x)$.

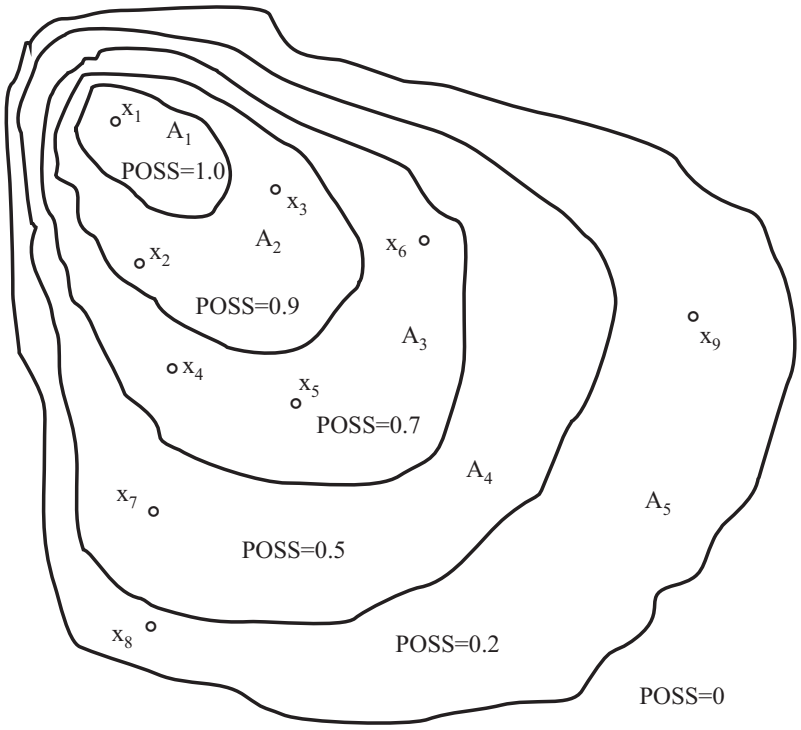


Рис. 4.5 – Геометрическая интерпретация согласованного распределения уверенности.

Можно показать, что приведенное определение согласованного распределения уверенности полностью соответствует определенной в [9] функции возможности

$$Poss: 2^x \to [0,1]$$

со свойствами:

$$Poss(\varnothing) = 0$$
$$Poss(X) = 1$$

для всех $A, B \subseteq X$

$$Poss(A \cup B) = \max\{Poss(A), Poss(B)\}$$

Действительно, функция возможности соответствует определению верхней вероятности P^*, которая однозначно задается с помощью контурной функции $\pi(x), x \in X$. Поэтому согласованное распределение уверенности можно назвать распределением возможностей на множестве X. Отсюда следует эквивалентность понятий фокального элемента для согласованного распределения уверенности и уровневого множества для распределения возможностей.

4.3 Нечеткие знания

Наконец, остановимся на проблеме представления нечетких знаний, являющейся ключевой при разработке интеллектуальных систем различного назначения. Нечеткие знания по своей природе разнообразны и могут быть условно разделены на следующие категории: неточность, недоопределенность, неоднозначность, т. е. любые нечеткости, между которыми нельзя провести четкой границы [7, 8].

Теория нечетких множеств — это, по сути дела, шаг на пути к сближению классической математики и всепроникающей неточности реального мира, к сближению, порожденному непрекращающимся стремлением человечества к лучшему пониманию процессов мышления и познания.

В настоящее время мы не способны сконструировать машины, которые могли бы соперничать с человеком в выполнении таких задач, как распознавание речи, перевод языков, понимание сущности, абстрагирование и обобщение, принятия решений в условиях неопределенности и тем более в задачах агрегирования информации.

Наша неспособность проектировать такие машины в значительной степени объясняется фундаментальным различием между человеческим разумом, с одной стороны, и «разумом» машины — с другой. Различие состоит в той способности человеческого мозга, которой в настоящем компьютеры не обладают: думать и делать заключения в неточных, неколичественных, нечетких терминах. Благодаря этой способности люди могут расшифровывать неразборчивый почерк, понимать искаженную речь, концентрировать внимание лишь на той информации, которая приводит к решению. И именно отсутствие этой способности делает даже самые сложные вычислительные машины непригодными к осуществлению контактов с человеком естественным образом, не прибегая к посредничеству искусственно созданных языков.

Множество или совокупность объектов — основное понятие в математике. Мы не очень быстро подошли к представлению о том, что многие, возможно, большинство человеческих знаний и связей с внешним миром включают такие построения, которые нельзя назвать множествами в классическом смысле. Их следует считать «нечеткими множествами» (или подмножествами), т. е. классами с нечеткими границами, когда переход от принадлежности к классу к непринадлежности происходит постепенно, не резко. По существу ставится под сомнение, что логика человеческого рассуждения основывается не на классической двузначной или даже многозначной логике, а на логике с нечеткими значениями истинности, с нечеткими связками и нечеткими правилами вывода.

В наших поисках точности мы пытались подогнать реальность, реальный мир под математические модели, которые не оставляют места нечеткости. Мы стремились выявить законы, управляющие поведением как отдельных людей, так и групп с помощью математических выражений, подобных тем, которые используются при анализе неодушевленных систем. Это, с нашей точки зрения, было и остается неправильно направленным усилием, подобным нашим давно забытым поискам перпеттум мобиле и философского камня.

Нам нужна новая точка зрения, новый комплекс понятий и методов, в которых нечеткость принимается как универсальная реальность человеческого существования, и, конечно, нам необходимо понять, как можно оперировать с нечеткими множествами внутри жестких рамок классической математики. Но, что намного важнее, мы должны разработать новые методы обращения с нечеткостями в систематическом (не обязательно количественном) смысле. Такие методы могут открыть много новых границ в психологии, экономике, лингвистике, операционных исследованиях, управлении и обеспечить основу для проектирования систем, разум которых значительно превосходит тот искусственный интеллект, который мы можем представить.

Заслуга Л. А. Заде заключается во введении понятия взвешенной принадлежности элемента к множеству. Элемент может принадлежать подмножеству в большей или меньшей степени, и отсюда появляется основное понятие — понятие нечеткости подмножества. С совершенно другой позиции, на основе n-местной логики, Пост, Лукашевич и Мойзил разработали общие теории, в которых могут найти место некоторые аспекты теории нечетких множеств.

В обработке информации человеком или с помощью компьютера числовые и нечисловые данные на входе, а иногда и на выходе — во многих случаях не могут быть ни четкими, ни даже вероятностными. Их можно поместить только в интервалах достоверности. В этом случае и работает теория нечетких множеств.

Что же предложил Заде? Во-первых, он расширил классическое канторовское понятие множества, допустив, что характеристическая функция (функция принадлежности элемента множеству) может принимать любые значения в интервале $(0, 1)$, а не только значения 0 или 1. Такое множество было названо нечетким (fuzzy). Заде определил также ряд операций над нечеткими множествами и предложил обобщение известных методов логического вывода modus ponens и modus tollens. Введя затем понятие лингвистической переменной и допустив, что в качестве ее значений (термов) выступают нечеткие множества, Л. Заде создал аппарат для описания процессов интеллектуальной деятельности, включая нечеткость

и неопределенность выражений. Математическая теория нечетких множеств позволяет описывать нечеткие понятия и знания, оперировать этими знаниями и делать нечеткие выводы. Основанные на этой теории методы построения компьютерных нечетких систем существенно расширяют области применения компьютеров. В последнее время нечеткое управление является одной из самых активных и результативных областей исследований применения теории нечетких множеств. Следует подчеркнуть активность публикаций по формированию качественных знаний и особенно аспекты, которые связанны с лингвистической неопределенностью при работе с экспертами на естественном языке в системах искусственного интеллекта и особенно в экспертных системах.

Приведем основные положения и определения нечетких множеств [2, 4, 6, 7, 12, 13].

4.3.1 Нечеткие множества

Понятие нечеткого множества. Пусть U — некоторое множество объектов (элементов, точек, обозначаемых через u и $\mu_A: U \to [0,1]$. Нечетким (расплывчатым) множеством A в U есть совокупность (график) упорядоченных пар

$$A\{u, \mu_A(u)\}; u \in U,$$

где $\mu_A(u)$ представляет собой принадлежности u к A, а $\mu_A: U \to [0,1]$ функция, отображающая U в пространство M, называемое пространством принадлежности.

Когда M содержит только две точки 0 и 1, A является нерасплывчатым (четким) и его функция принадлежности совпадает с характеристической функцией нерасплывчатого множества.

В последующем мы будем предполагать, что M есть интервал $[0,1]$ причем 0 и 1 представляют соответственно низшую и высшую степень принадлежности. В качестве множества принадлежностей рассматриваются и другие интерпретации: $[-1,1]$ в экспертной системе MYCIN, $[0,10]$, $[0,100]$, любое частично упорядоченное множество и, в частности, решетка. Таким образом, задание нечеткого множества A в U эквивалентно заданию его функции принадлежности $\mu_A(u)$ и, несмотря на нечеткость его границ, может быть точно определено путем сопоставления каждому элементу u числа, лежащего между 0 и 1, т. е. $\mu_A(u)$.

Приведем частные случаи нечетких множеств, которые используются в литературе: это S и π — функции принадлежности [4, 11]:

$$\pi(u,\beta,\gamma) = \begin{cases} S\left(u; \gamma-\beta, \gamma-\frac{\beta}{2}, \gamma\right) & \text{для } u \leqslant \gamma; \\ S\left(u; \gamma, \gamma+\frac{\beta}{2}, \gamma+\beta\right) & \text{для } u \geqslant \gamma. \end{cases}$$

$$S(u,\alpha,\beta,\gamma) = \begin{cases} 0 & \text{для } u \leqslant \alpha; \\ 2\left(\dfrac{u-\alpha}{\gamma-\alpha}\right)^2 & \text{для } \alpha \leqslant u \leqslant \beta; \\ 1-2\left(\dfrac{u-\alpha}{\gamma-\alpha}\right)^2 & \text{для } \beta \leqslant u \leqslant \gamma; \\ 1 & \text{для } u \geqslant \gamma. \end{cases}$$

Представим их графически (рис. 4.6)

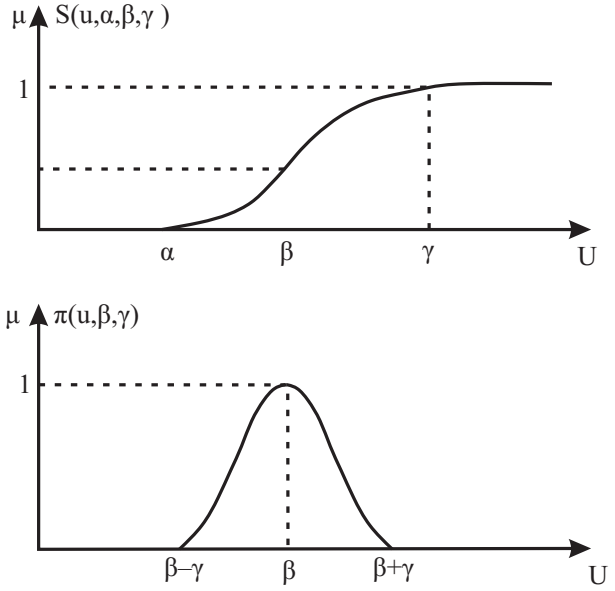


Рис. 4.6 – Частные случаи нечетких множеств.

Для частного случая, когда U является подмножеством числовой прямой, часто используются нечеткие множества (L-R)-типа (рис. 4.7). Функции принадлежности для таких множеств задаются с помощью функций L и R, удовлетворяющих следующим требованиям:

- $L(0) = R(0) = 1$;
- L и R — невозрастающие функции на множестве неотрицательных действительных чисел.

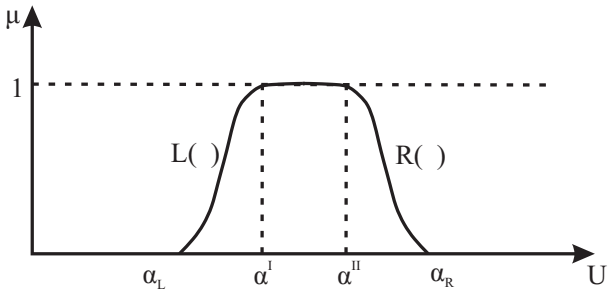


Рис. 4.7 – Нечеткое множество с функцией принадлежности (L-R)-типа.

Функция принадлежности нечеткого множества A, имеющая (L-R)-тип, задается следующим образом:

$$\mu_A(u) = \begin{cases} L\left(\dfrac{\alpha^I - u}{\alpha_L}\right) & \text{при } u \leqslant \alpha^I, \alpha_L > 0; \\ R\left(\dfrac{u - \alpha^{II}}{\alpha_R}\right) & \text{при } u \geqslant \alpha^{II}, \alpha_R > 0; \\ 1 & \text{при } u \in [\alpha^I, \alpha^{II}]. \end{cases}$$

Часто используются линейные функции (L-R)-типа (рис. 4.8)

$$\mu_A(u) = \begin{cases} 0 & \text{при } u \leqslant \alpha_L; \\ \dfrac{u - \alpha_L}{\alpha^I - \alpha_L} & \text{при } \alpha_L \leqslant u \leqslant \alpha^I; \\ 1 & \text{при } \alpha^I \leqslant u \leqslant \alpha^{II}; \\ \dfrac{\alpha_R - u}{\alpha_R - \alpha^{II}} & \text{при } \alpha^{II} \leqslant u \leqslant \alpha_R; \\ 0 & \text{при } u \geqslant \alpha_R \end{cases}$$

и частные случаи, трапециидальные при $\alpha^I < \alpha^{II}$ и треугольные при $\alpha^I = \alpha^{II} = \alpha$

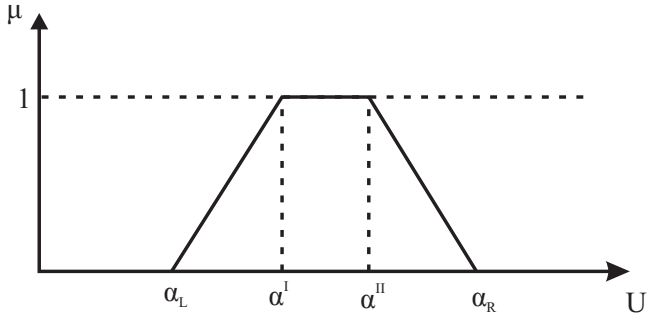


Рис. 4.8 – Линейная функция (L-R)-типа.

Множество нечетких подмножеств и его свойство.

Обозначим $P(U)$ множество всех подмножеств U. Например, $U = \{u_1, u_2, u_3\}$, тогда $P(U) = \{\varnothing, \{u_1\}, \{u_2\}, \{u_3\}, \{u_1, u_2\}, \{u_2, u_3\}, \{u_1, u_3\}, U\}$. Это множество состоит из $2^3 = 8$ элементов. В общем случае для множеств $U = \{u_1, u_2, \ldots, u_n\}$ можно определить 2^n элементов. Для нечетких подмножеств множество всех подмножеств или «множество нечетких подмножеств» определяется иначе. Выпишем множество $P(U)$ нечетких множеств U:

$$P(U) = \Big\{\{(x_1|0), (x_2|0)\}, \{(x_1|0), (x_2|0.5)\}, \{(x_1|0.5), (x_2|0)\}, \{(x_1|0.5), (x_2|0.5)\}, \{(x_1|0), (x_2|1)\}, \{(x_1|1), (x_2|0)\}, \{(x_1|1), (x_2|0.5)\}, \{(x_1|0.5), (x_2|1)\}, \{(x_1|1), (x_2|1)\}\Big\}.$$

В общем случае, если

$$card\, U = n \text{ и } card\, M = m,$$

где card означает «мощность», а в нашем случае число элементов множество, то $card\,P(U) = m^n$.

Простейшие операции над нечеткими множествами. Основные теорико-множественные операции над $P(U)$ в теории нечетких множеств.

Включение. Пусть U — множество принадлежностей и A и B — два нечетких подмножества U. Будем говорить, что A содержится в B, если $\forall u \in U: \mu_A(u) \leqslant \mu_B(u)$, и обозначать $A \subset B$

Пример

Пусть $U = \{u_1, u_2, u_3, u_4\}$, $M = [0, 1]$.

$$A = \{(x_1|0.4), (x_2|0.2), (x_3|0), (x_4|1)\}.$$
$$B = \{(x_1|0.3), (x_2|0), (x_3|0), (x_4|0)\}.$$

Имеем $A \subset B$, так как $0.3 < 0.4, 0 < 0.2, 0 = 0, 0 < 1$.

Равенство. Два нечетких подмножества A и B множества U равны $A = B$ тогда и только тогда, когда $\forall u \in U: \mu_A(u) \leqslant \mu_B(u)$.

Дополнение. Два нечетких подмножества A и B множества U дополняют друг друга: $B = \sim A$ или $\sim A = B$, если $\forall u \in U: \mu_B(u) = 1 - \mu_A(u)$, это обозначается $B = \sim A$ или $\sim A = B$.

Пример

Пусть $U = \{x_1, x_2, x_3, x_4, x_5, x_6\}$, $M = [0, 1]$.

$$A = \{(x_1|0.13), (x_2|0.61), (x_3|0), (x_4|0), (x_5|1), (x_6|0.03)\}.$$
$$B = \{(x_1|0.87), (x_2|0.39), (x_3|1), (x_4|1), (x_5|0), (x_6|0.97)\}.$$

Тогда, очевидно, $\bar{A} = B$.

Пересечение. Пересечение двух нечетких подмножеств A и B $(A \cap B)$ в множестве U определяется как наименьшее нечеткое подмножество, содержащееся одновременно в A и B:

$$\forall u \in U: \mu_{A \cap B}(u) = \min\big(\mu_A(u), \mu_B(u)\big).$$

Пример

Пусть $U = \{u_1, u_2, u_3, u_4, u_5\}$,

$$A = \{(x_1|0.2), (x_2|0.7), (x_3|1), (x_4|0), (x_5|0.5)\},$$
$$B = \{(x_1|0.5), (x_2|0.3), (x_3|1), (x_4|0.1), (x_5|0.5)\},$$
$$A \cap B = \{(x_1|0.2), (x_2|0.3), (x_3|1), (x_4|0), (x_5|0.5)\}.$$

Объединение. Объединение двух подмножеств A и B $(A \cup B)$ в множестве U определяется как наибольшее нечеткое подмножество, которое содержится как в A, так и B:

$$\forall u \in U: \mu_{A \cap B}(u) = \max\left(\mu_A(u), \mu_B(u)\right).$$

Алгебраическая (дизъюнктивная) сумма. Дизъюнктивная сумма двух нечетких подмножеств A и B $A \oplus B$ определяется в терминах объединений и пересечений следующим образом:

$$A \oplus B = (A \cap B) = (A \cap B) \cup (A \cap B) = \mu_A(u) + \mu_B(u) - \mu_A(u) \cdot \mu_B(u).$$

Алгебраическое произведение. Алгебраическое произведение двух нечетких множеств A и B $(A \cdot B)$ определяется следующим образом:

$$\forall u \in U: \mu_{A \cdot B}(u) = \mu_A(u) \cdot \mu_B(u).$$

Расстояние между нечеткими множествами. В математике под расстоянием $d(x, y)$, т. е. парой элементов множества U, будет величина, удовлетворяющая следующим условиям:

$\forall x, y, z \in U$:
1) $d(x, y) \geqslant 0$ — *неотрицательность*;
2) $d(x, y) = d(y, x)$ — *симметричность*;
3) $d(x, z) \leqslant d(x, y) * d(y, z)$ — *транзитивность*;
4) $d(x, z) = 0$.

Здесь $*$ оператор, связанный с понятием расстояния Хэмминга, — т. е. действительное расстояние, если оператор $* = +$, то обычная сумма.

Для двух обычных подмножеств A и B и конечного множества U

$A =$

u_1	u_2	u_3		...		u_7
1	0	0	1	0	1	0

$B =$

u_1	u_2	u_3		...		u_7
0	1	0	0	0	1	1

расстояние Хэмминга между A и B рассчитывается по формуле

$$d(A,B) = \sum_{i=0}^{n} \left|\mu_A(u_i) - \mu_B(u_i)\right|, \text{ т. е.}$$

$$d(A,B) = |1-0| + |0-1| + |0-0| + |1-0| + |0-0| + |1-1| + |0-1| =$$
$$= 1 + 1 + 0 + 1 + 0 + 0 + 1 = 4.$$

Для конечного множества U мощности n (т. е. n — число элементов в U) определим также относительное расстояние Хэмминга:

$$\delta(A,B) = \left(\frac{1}{n}\right) d(A,B).$$

Для нашего примера $\delta(A,B) = d(A,B)/7 = 4/7$.

Для обобщенного понятия «расстояние Хэмминга» для нечетких множеств рассмотрим три нечетких подмножества $A, B, C \subset U$, где U — конечное множество мощности n.

	u_1	u_2	u_3	…	u_n
$A =$	a_1	a_2	a_3	…	a_n
$B =$	b_1	b_2	b_3	…	b_n
$C =$	c_1	c_2	c_3	…	c_n

Предположим, что мы определили расстояние $D(a_i, b_i)$ между a_i и b_i для всех $i = 1, 2, \ldots, n$, а также для (b_i, c_i) и (a_i, c_i). Тогда для этих расстояний справедливо неравенство

$$\forall i = 1, 2, \ldots, n: D(a_i, c_i) \leqslant D(a_i, b_i) * D(b_i, c_i).$$

Теперь можно записать:

$$\sum_{i=0}^{n} D(a_i, c_i) \leqslant \sum_{i=0}^{n} D(a_i, b_i) + \sum_{i=0}^{n} D(b_i, c_i),$$

$$\sqrt{\sum_{i=0}^{n} D^2(a_i, c_i)} \leqslant \sqrt{\sum_{i=0}^{n} D^2(a_i, b_i)} + \sqrt{\sum_{i=0}^{n} D^2(b_i, c_i)}.$$

Эти две формулы дают две оценки расстояния между нечеткими подмножествами: первая — линейную оценку, а вторая — квадратичную.

Приведем окончательные известные расстояния между нечеткими множествами (табл. 4.1).

Понятие расстояния используется для измерения степени нечеткости множества. Мера нечеткости — это параметр оценки качества процедур и алгоритмов принятия решений, распознавания образов и т. д. Разработаны следующие методы оценки нечеткости: энтропийный, метрический, аксиоматический. Не будем останавливаться на этих методах, а покажем актуальность приведенных расстояний на простейшем примере определения обычного подмножества, ближайшего к нечеткому. Таким образом, четкое множество А должно быть расположено на наименьшем Евклидовом расстоянии от данного нечеткого подмножества (или иметь наименьшую норму).

Таблица 4.1 – Расчетные формулы расстояний для нечетких подмножеств на универсальном множестве U.

Вид расстояния	Тип универсального множества	Формула расстояния
Хэмминга	$\|U\| = n$	$d(A,B) = \sum_{i=0}^{n} \left\|\mu_A(u_i) - \mu_B(u_i)\right\|$
Хэмминга	$U \subseteq R^1$	$d(A,B) = \int_u \left\|\mu_A(u) - \mu_B(u)\, du\right\|$
Относительное Хэмминга	$U \subseteq R^1$	$\delta(A,B) = \frac{1}{\|U\|} \int_u \left\|\mu_A(u) - \mu_B(u)\, du\right\|$
Евклида	$\|U\| = n$	$l(A,B) = \sqrt{\sum_{i=0}^{n} \left(\mu_A(u_i) - \mu_B(u_i)\right)^2}$
Относительное Евклида	$\|U\| = n$	$\varepsilon(A,B) = \frac{1}{\sqrt{n}} \sqrt{\sum_{i=0}^{n} \left(\mu_A(u_i) - \mu_B(u_i)\right)^2}$
Относительное Хэмминга	$\|U\| = n$	$\delta(A,B) = \frac{1}{n} \sum_{i=0}^{n} \left\|\mu_A(u_i) - \mu_B(u_i)\right\|$
Евклида	$U \subseteq R^1$	$l(A,B) = \sqrt{\int_{i=0} \left(\mu_A(u) - \mu_B(u)\right)^2 du}$
Относительное Евклида	$U \subseteq R^1$	$\varepsilon(A,B) = \frac{1}{\sqrt{\|U\|}} \sqrt{\int_{i=0} \left(\mu_A(u) - \mu_B(u)\right)^2 du}$

Легко доказать, что это будет обычное множество, такое, что

$$\mu_A = \begin{cases} 0, & \text{если } \mu_A(u_i) < 0.5, \\ 1, & \text{если } \mu_A(u_i) > 0.5, \\ 0 \text{ или } 1, & \text{если } \mu_A(u_i) = 0.5, \end{cases}$$

где по определению пользователя мы принимаем, что $\mu_A(u_i) = 0$, если $\mu_A(u_i) = 0.5$.

Пример

Нечеткое множество	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8
	0.2	0.8	0.5	0.3	1	0	0.9	0.4
Четкое множество	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8
	0	1	0	0	1	0	1	0

Подмножество α-уровня. Пусть $\alpha \in [0,1]$; подмножеством α-уровня нечеткого подмножества A будем называть обычное нечеткое подмножество $A_\alpha = \{x | \mu_A(u) \geqslant \alpha\}$.

Пример

Пусть задано нечеткое подмножество

$$A = \begin{array}{|c|c|c|c|c|c|c|} \hline u_1 & u_2 & u_3 & & \ldots & & u_7 \\ \hline 0.8 & 0.1 & 1 & 0.3 & 0.6 & 0.2 & 0.5 \\ \hline \end{array}$$

Имеем подмножество α-уровня:

$$A_{0.3} = \begin{array}{|c|c|c|c|c|c|c|} \hline u_1 & u_2 & u_3 & & \ldots & & u_7 \\ \hline 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ \hline \end{array}$$

$$A_{0.55} = \begin{array}{|c|c|c|c|c|c|c|} \hline u_1 & u_2 & u_3 & & \ldots & & u_7 \\ \hline 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ \hline \end{array}$$

4.3.2 Нечеткие отношения

Понятие отношения (графа) играет важную роль в математике и в системах искусственного интеллекта: распознавании образов, проектировании сложных систем, выводах, системах формирования БЗ, анализе, управлении, моделировании, принятии решений и т. д. Их можно обобщить на случай нечетких подмножеств. При этом обнаруживаются некоторые новые свойства. Например, понятие класса эквивалентности заменяется понятием подобия, не таким жестким, но имеющим смысл для многих приложений. Предпорядок и порядок обобщаются аналогичным образом, определяются отношения сходства и несходства и т. д.

Пусть $U_1 = \{x\}$ и $U_2 = \{y\}$ обычные множества. Прямое произведение $U_1 \times$ $\times U_2$ множеств U_1 и U_2 есть множество упорядоченных пар (x, y), т. е. $U_1 \times U_2 =$ $= \{(x, y): x \in U_1, y \in U_2\}$.

Пусть M — множество принадлежностей. Тогда нечеткое множество R такое, что $\forall (x, y) \in U_1 \times U_2, \mu_R(x, y) \in M$ называется бинарным отношением R в $U_1 \times U_2$.

Пример

Пусть $U_1 = \{x_1, x_2, x_3\}, U_2 = \{y_1, y_2, y_3, y_4, y_5\}, M = [0, 1]$.

Тогда нечеткое бинарное отношение можно (субъективно) задать следующей таблицей:

Обобщая, получим нечеткое n-парное отношение, т. е. нечеткое множество в $P_n = U_1 \times U_2 \times \cdots \times U_n$.

Далее для обозначения экстремума будем использовать символы:

$\bigvee\limits_x$ — максимум относительно элемента или переменной x;

$\bigwedge\limits_x$ — минимум относительно элемента или переменной x.

	y_1	y_2	y_3	y_4	y_5
x_1	0	0	0.1	0.3	1
x_2	0	0.8	0	0	1
x_3	0.4	0.4	0.5	0	0.2

Так $\mu_1(x) = \bigvee\limits_y \mu(x,y) = \max\limits_y \mu(x,y)$ и $\mu_2(x) = \bigwedge\limits_y \mu(x,y) = \min\limits_y \mu(x,y)$.

Проекции нечеткого отношения. Первую проекцию R определяет функция принадлежности

$$\mu_R^{(1)}(x) = \bigvee_y \mu_R(x,y).$$

Вторую проекцию R определяет функция принадлежности

$$\mu_R^{(2)}(x) = \bigwedge_y \mu_R(x,y).$$

Вторая проекция первой проекции (или наоборот) будет называться глобальной проекцией нечеткого отношения и обозначается $h(R)$:

$$h(R) = \bigvee_x \bigvee_y \mu_R(x,y) = \bigvee_y \bigvee_x \mu_R(x,y).$$

Пример

Зададим матрицу R и рассчитаем первую, вторую и глобальные проекции нечеткого отношения (рис. 4.9).

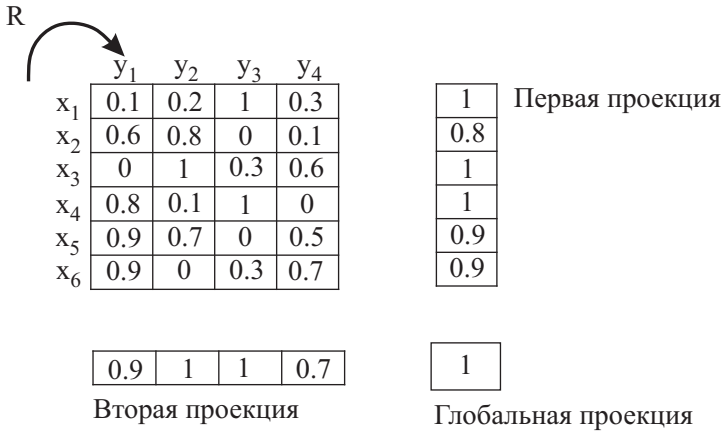


Рис. 4.9 – Расчет проекций нечеткого отношения.

Носитель нечеткого отношения. Носителем нечеткого отношения R называется обычное множество упорядоченных пар (x,y), для которых функция принадлежности положительна:

$$S(R) = \{(x,y) | \mu_R(x,y) > 0\}.$$

Далее можно рассматривать объединение, пересечение, алгебраическое произведение, сумму, дополнение, дизъюнктивную сумму двух отношений и обычное отношение ближайшее к нечеткому [4].

Композиция двух нечетких отношений. Операция композиции нечетких отношений R_1 в $X \times Y$ и R_2 в $Y \times Z$ позволяет определить нечеткое отношение в $X \times Z$.

Max-min композиция. Пусть $R_1 \subset X \times Y$ и $R_2 \subset Y \times Z$; (max-min)-композиция отношений R_1 и R_2 обозначается $R_1 \circ R_2$ и определяется выражением

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

где $x \in X, y \in Y, z \in Z$.

Пример

Пусть функции принадлежности заданы на конечном универсальном множестве (рис. 4.10).

R_1

	y_1	y_2	y_3	y_4	y_5
x_1	0.1	0.2	0	1	0.7
x_2	0.3	0.5	0	0.2	1
x_3	0.8	0	1	0.4	0.3

R_2

	z_1	z_2	z_3	z_4
y_1	0.9	0	0.3	0.4
y_2	0.2	1	0.8	0
y_3	0.8	0	0.7	1
y_4	0.4	0.2	0.3	0
y_5	0	1	0	0.8

$R_1 \circ R_2$

	z_1	z_2	z_3	z_4
x_1	0.4	0.7	0.3	0.7
x_2	0.3	1	0.5	0.8
x_3	0.8	0.3	0/7	1

Рис. 4.10 – Пример расчета max-min композиции.

На рисунке заданы R_1 и R_2 матрицами и рассчитывается композиция $R_2 \circ R_1$. Рассчитаем $(x,z) = (x_1, z_1)$.

$$\min\left(\mu_{R_1}(x_1,y_1), \mu_{R_2}(y_1,z_1)\right) = \min(0.1;0.9) = 0.1;$$
$$\min\left(\mu_{R_1}(x_1,y_2), \mu_{R_2}(y_2,z_1)\right) = \min(0.2;0.2) = 0.2;$$
$$\min\left(\mu_{R_1}(x_1,y_3), \mu_{R_2}(y_3,z_1)\right) = \min(0;0.8) = 0;$$
$$\min\left(\mu_{R_1}(x_1,y_4), \mu_{R_2}(y_4,z_1)\right) = \min(1;0.4) = 0.4;$$
$$\min\left(\mu_{R_1}(x_1,y_5), \mu_{R_2}(y_5,z_1)\right) = \min(0.7;0) = 0;$$
$$\max_{y_i}\left(\min \mu_{R_1}(x_i,y_i), \mu_{R_2}(y_i,z_i)\right) = \max(0.1;0.2;0;0.4;0) = 0.4;$$

Результат представлен на рис. 4.10.

(Max-)-композиция.* Операцию $\wedge$ в предыдущем примере произвольно можно заменить любой другой, для которой выполняются те же ограничения, что и для $\wedge$: ассоциативность и монотонность неубывания по каждому аргументу.

Тогда можно записать:

$$\mu_{R_1 * R_2}(x, z) = \bigvee_y \left[\mu_{R_1}(x, y) * \mu_{R_2}(y, z)\right].$$

Теперь можно представить различные композиции.

Например:

1) (max-·) — композиция, где · это умножение и формула приобретает вид:

$$\mu_{R_1 \cdot R_2}(x, z) = \bigvee_y \left[\mu_{R_1}(x, y) \cdot \mu_{R_2}(y, z)\right].$$

2) Замена операции $\min(\wedge)$ на среднее арифметическое. Тогда формула приобретает вид:

$$\mu_{R_1 * R_2}(x, z) = \bigvee_y \left[\frac{1}{2}\left(\mu_{R_1}(x, y) + \mu_{R_2}(y, z)\right)\right].$$

Выбор варианта (max-*)-композиции определяется свойствами задачи.

Рассматривая нечеткие бинарные отношения, введем некоторые типы отношений.

1) Транзитивное и рефлексивное нечеткое бинарное отношение называется нечетким *отношением предпорядка*.
2) Антисимметричное нечеткое отношение предпорядка называется нечетким *отношением порядка*.
3) Транзитивное рефлексивное симметричное нечеткое бинарное отношение называется *отношением подобия*.
4) Нечеткое бинарное отношение, обладающее свойствами антирефлективности, симметричности и (min-max)-транзитивности называется *отношением различия*.

4.3.3 Элементы теории приближенных рассуждений

Системы нечеткого логического вывода играют важнейшую роль в многочисленных приложениях нечетких множеств (экспертные системы, системы автоматического формирования БЗ, распознавание образов, проектирование сложных систем, нейронные сети, системы принятия решений, понимание естественного языка и т. д.).

В основе большинства таких систем лежат логические правила вида «Если . . . , то . . . », в которых посылки и выводы являются нечеткими понятиями [2–7].

Приближенные рассуждения на основе modus ponens. Известно правило вывода modus ponens в обычной логике:

Посылка 1: если x есть A, то y есть B

Посылка 2: x есть A

Следствие: y есть B,

где x, y — имена объектов, A, B — обозначения понятий областей рассуждения U и V соответственно.

Пример

Посылка 1: если слива черная, то слива спелая

Посылка 2: эта слива черная

Следствие: эта слива спелая.

Обобщением данного правила с посылками, являющимися нечеткими понятиями, можно записать:

Посылка 1: если x есть A, то y есть B

Посылка 2: x есть A^1

Следствие: y есть B^1.

Пример

Посылка 1: если слива черная, то слива спелая

Посылка 2: эта слива очень черная

Следствие: эта слива очень спелая.

Мы можем получить обычный modus ponens при $A^1 = A$ и $B^1 = B$, а в последнем примере у нас обобщенный modus ponens. В посылке 1 мы видим некоторое соответствие между A и B. В литературе приводится несколько отношений, соответствующих такому соответствию.

Если мы рассмотрим отношения в универсальных множествах U и V; функции принадлежности $\mu_A(u), \mu_B(u)$; операции $\times, \cup, \cap, \sim$ и $\oplus$, т. е декартово произведение, объединение, пересечение, дополнение и ограниченная сумма, то представим следующие нечеткие отношения, которые могут служить формализацией нечеткого условного высказывания «Если x есть A, то y есть B» [116].

$$1)\ R_m = (A \times B) \cup (\sim A \times V) = \max\Big[\min\big(\mu_A(u), \mu_B(v)\big), 1 - \mu_A(u)\Big].$$

$$2)\ R_a = (\sim A \times V) \oplus (U \times B) = \min\big(\mu_A(u), \mu_B(v)\big).$$

$$3)\ R_S = A \times V \overset{S}{\Rightarrow} U \times B = \mu_A(u) \overset{S}{\rightarrow} \mu_B(v),$$

$$\text{где } \mu_A(u) \overset{S}{\rightarrow} \mu_B(v) = \begin{cases} 1 & \text{при } \mu_A(u) \leqslant \mu_B(v), \\ 0 & \text{при } \mu_A(u) > \mu_B(v). \end{cases}$$

$$4)\ R_g = A \times V \overset{g}{\Rightarrow} U \times B = \mu_A(u) \overset{g}{\rightarrow} \mu_B(v),$$

$$\text{где } \mu_A(u) \overset{g}{\rightarrow} \mu_B(v) = \begin{cases} 1 & \text{при } \mu_A(u) \leqslant \mu_B(v), \\ \mu_B(v) & \text{при } \mu_A(u) > \mu_B(v). \end{cases}$$

$$5)\ R_{sg} = \left(A \times V \overset{s}{\Rightarrow} U \times B\right) \cup \left(\sim A \times V \overset{g}{\Rightarrow} U \times \sim B\right) =$$
$$= \min\left[\mu_A(u) \overset{s}{\rightarrow} \mu_B(v), \left(1 - \mu_A(u)\right) \overset{g}{\rightarrow} \left(1 - \mu_B(v)\right)\right].$$
$$6)\ R_{gg} = \left(A \times V \overset{g}{\Rightarrow} U \times B\right) \cap \left(\sim A \times V \overset{g}{\Rightarrow} U \times \sim B\right) =$$
$$= \min\left[\mu_A(u) \overset{g}{\rightarrow} \mu_B(v), \left(1 - \mu_A(u)\right) \overset{g}{\rightarrow} \left(1 - \mu_B(v)\right)\right].$$
$$7)\ R_{gs} = \left(A \times V \overset{g}{\Rightarrow} U \times B\right) \cap \left(\sim A \times V \overset{s}{\Rightarrow} U \times \sim B\right) =$$
$$= \min\left[\mu_A(u) \overset{g}{\rightarrow} \mu_B(v), \left(1 - \mu_A(u)\right) \overset{s}{\rightarrow} \left(1 - \mu_B(v)\right)\right].$$
$$8)\ R_{ss} = \left(A \times V \overset{s}{\Rightarrow} U \times B\right) \cap \left(\sim A \times V \overset{s}{\Rightarrow} U \times \sim B\right) =$$
$$= \min\left[\mu_A(u) \overset{s}{\rightarrow} \mu_B(v), \left(1 - \mu_A(u)\right) \overset{s}{\rightarrow} \left(1 - \mu_B(v)\right)\right].$$

Следствие B^1 в обобщенном modus ponens получается из посылки 1 и посылки 2 как max-min-композиция нечеткого множества A^1 и нечеткого отношения, полученного в одном из правил, т. е. $R_m, R_a, R_s, \ldots, R_{ss}$.

Например: $B^1_m = A^1 \circ R_m = A^1 \circ \left[(A \times B) \cup (\sim A \times V)\right]$.

Таким образом, из одной посылки A^1 мы можем получить различные выводы, и это различие в основном зависит от решаемой задачи и ЛПР.

В [4–7] для сравнения различных методов нечетких рассуждений формулируются интуитивно различные требования к связи между A^1 и B^1. В качестве A^1 берутся высказывания:

A^1 = очень A

A^1 = более или менее A

A^1 = не A

Каким может быть B^1 для A^1? Если существует сильная причинная связь между высказываниями «x есть A» и «y есть B» в высказывании «если x есть A, то y есть B», то для A^1 = очень A, мы должны требовать B^1 = очень B. Если причинная связь не является жесткой, для указанного A^1 можно требовать выполнения и $B^1 = B$. Если, например, утверждение «Если x есть A, то y есть B» неявно подразумевается как утверждение «если x есть A, то y есть B, иначе y не есть B», то для A^1 = не A мы должны требовать выполнения B^1 = не B.

Приближенные рассуждения на основе modus tollens подобны вышеприведенным рассуждениям, а следствие A^1 получается в результате max-min-композиции соответствующих отношений $R_m, R_a, R_s, \ldots, R_{ss}$ и нечеткого множества B^1.

Пример

Посылка1: если x есть A, то y есть B

Посылка 2: y есть не B^1

Следствие: x есть не A^1.

4.3.4 Лингвистическая переменная

Понятие лингвистической переменной. Л. Заде ввел понятие нечеткой переменной как тройку [3]:

$$\langle \alpha, U, G \rangle,$$

где α — наименование (имя) нечеткой переменной;

U — область ее определения (универсальное множество);

G — нечеткое множество в U, описывающее ограничения на возможные значения нечеткой переменной α (ее семантику).

В зависимости от характера множества U нечеткие переменные могут быть разделены на числовые и нечисловые. К числовым отнесем переменные, у которых $U \subset R^1$.

Лингвистическая переменная является переменной более высокого порядка, чем нечеткая переменная. Это определяется тем, что значениями лингвистической переменной являются нечеткие переменные. Например, значениями лингвистической переменной «качество» могут быть «низкое», «среднее» и т. д. Каждое из этих значений является названием нечеткой переменной.

Итак, значениями лингвистической переменной являются не числа, как у числовой переменной, а слово или предложение в естественном или формальном языках. Это свойство лингвистической переменной дает возможность приближенно описывать сложные, количественно трудноописываемые на привычном естественном языке системы и явления.

Для описания структуры лингвистической переменной используются следующие два правила:

- Синтаксическое, которое задается в форме грамматики, порождающей название значений переменной;
- Семантическое, которое определяет алгоритмическую процедуру для вычисления смысла каждого значения. Семантическое правило M (например, экспертный опрос) позволяет превратить каждое новое значение лингвистической переменной, образуемое процедурой V, в нечеткую переменную, т. е. приписать ему семантику путем формирования соответствующего нечеткого множества.

Введем понятие лингвистической переменной как

$$\langle A, T(A), U, V, M \rangle,$$

где A — название переменной;

$T(A)$ — терм-множества переменной A, т. е. множество названий лингвистических значений переменной A, причем каждое из таких значений — нечеткая переменная со значениями из универсального множества U;

V — синтаксическое правило (обычно грамматика), порождающее названия значений лингвистической переменной A;

M — семантическое правило, которое ставит в соответствие каждой нечеткой переменной из $T(A)$ нечеткое подмножество универсального множества U.

Пример

Пусть определяется температура окружающего воздуха с помощью понятий-значений: «очень холодно», «холодно», «свежо», «не холодно», «тепло», «жарко». Каждому понятию-значению соответствует определенная температура (рис. 4.11). При этом минимальная и максимальная температуры соответственно равны −40 °C и +40 °C. Тогда формально лингвистическая переменная «температура» представляется набором:

$$\langle \text{температура, очень холодно, холодно, свежо, не холодно, тепло, жарко } [-40, +40], V, M \rangle,$$

где V — процедура перебора элементов терм-множества T;

M — процедура, которая ставит в соответствие нечетким переменным (жарко, тепло и т. д.) температуру окружающего воздуха. Иначе. Процедура M определяет взаимосвязь нечетких переменных с соответствующими функциями принадлежности.

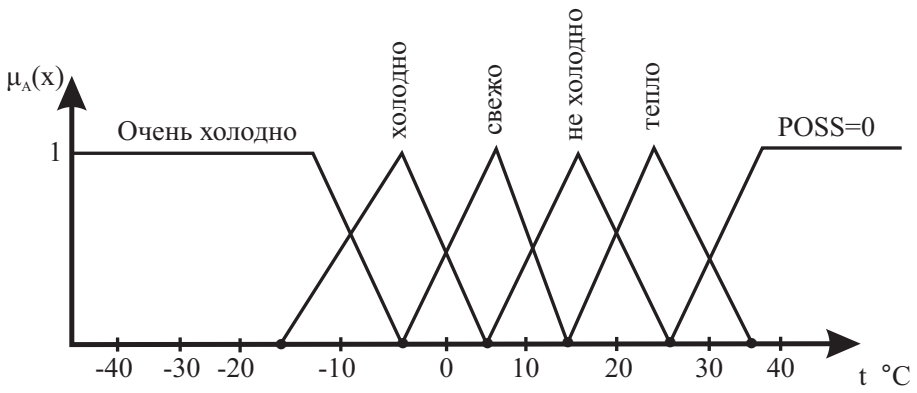
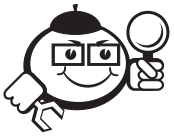


Рис. 4.11 – Лингвистическая переменная «температура».



Пример

Пусть оценка объема вырабатываемой продукции производится с помощью понятий нечетких переменных — «мало», «ниже среднего», «среднее», «выше среднего», «много». При этом максимальный объем выработки продукции равен 20 т/ч. Формализация такой оценки может быть проведена с помощью лингвистической переменной A = «объем продукции», характеризуемой набором:

$$\langle \text{Объем продукции}, T(A)[0.20], V, M \rangle,$$

где $T(A) = \{$мало, ниже среднего, среднее, выше среднего, много$\}$ — терм-множество лингвистической переменной, $U = [0, 20]$ — универсальное множество, характеризующее область определения базовой переменной для каждого из термов и для базовой переменной;

V — процедура перебора элементов множества $T(A)$;

M — процедура экспертного опроса, с помощью которого определяется смысл нечеткой переменной, т. е. множества M.

. . . . . .

Л. Заде различает базовые термины («холодно», «нормально», «жарко») и модификаторы («очень», «не»). Модификаторы могут применяться как к базовым переменным, так и к комбинациям базового термина и модификатора («очень-очень холодно»). Правила применения модификаторов задаются синтаксическим правилом V.

Разница между базовым термином и модификатором заключается в следующем. Для базовых переменных функции принадлежности задаются, а модификаторы действуют как некоторые операторы над этими функциями. Например, в качестве «очень» предлагается следующая модификация функции принадлежности какого-либо термина: $\mu(u) = \mu^2(u)$ или $\mu(u) = \mu^{1/2}(u)$ и т. д.

Теперь рассмотрим ситуацию, когда лингвистическая переменная A = «рост» имеет два терм-множества $T_1(A) = \{$низкий, высокий$\}$ и $T_2(A) = \{$низкий, средний, высокий$\}$ — имеет три терм-множеств. Интуитивно ясно, что функции принадлежности «низкий», «высокий» в обоих случаях будут различаться: новое понятие «средний» модифицирует их, сдвигая к концам универсума. Это говорит о том, что семантика некоторого термина зависит от контекста.

Проблема. Можно ли, учитывая некоторые особенности восприятия человеком объектов реального мира и их описания, сформулировать правило выбора оптимального множества значений признаков, по которым описываются эти объекты?

Возможны два критерия оптимальности:

- Под оптимальным понимаются такие множества значений, используя которые человек испытывает минимальную неопределенность при описании объекта.
- Если объект описывается некоторым количеством экспертов, то под оптимальными понимаются также множества значений, которые обеспечивают минимальную степень рассогласованности описаний.

Представим методику выбора оптимального множества значений качественного признака.

1) Формируются все возможные («разумные») множества значений признаков.
2) Каждое множество значений признака представляется в виде полного ортогонального семантического пространства.
3) Для каждого множества значений вычисляется степень нечеткости полного ортогонального семантического пространства.
4) В качестве оптимального множества значений как по критерию 1, так и по критерию 2 выбирается то множество, степень которого минимальна.

В повседневной жизни часто используются приближенные выводы на основе нечетких исходных высказываний. Например:

Если температура низкая, то время нагрева большое.

Температура слишком низкая.

Время нагрева слишком большое.

Для формализации такого вывода Заде предложил обобщенное правило modus ponens в форме:

Посылка 1: если x есть A, то y есть B, иначе есть C

Посылка 2: x есть A^1

Следствие: y есть D,

где A, A^1, B, C, D — нечеткие переменные, представленные нечеткими множествами, заданными на следующих областях определения U, U, V, V, V; x, y — лингвистические переменные. Это правило имеет существенную особенность по сравнению с традиционным правилом modus ponens

$$\frac{A, A \to B}{B}.$$

В обобщенном правиле множества A и $A1$ не совпадают. Если A и $A1$ более или менее сопоставимы, то можно получить следствие D, в некоторой степени подобное B, иначе D, в некоторой степени подобно C. Ранее было отмечено, что для формализации правила $A \to B$ можно использовать нечеткое отношение, определяемое с помощью декартова произведения $A \times B$. По аналогии с этим, для формализации первой посылки обобщенного правила вывода, Заде предложил два композиционных отношения. Одно из них называется максиминным и имеет вид

$$R_m = (A \times B) \cup (\sim A \times C).$$

Здесь нечеткое отношение $A \times B$ соответствует первой части правила (т. е. «если A, то B»), а нечеткое отношение $\bar{A} \times C$ — второй части правила (т. е. «если не A, то C»). В частном случае, когда альтернативная ветвь правила отсутствует, отношение R_m сводится к отношению $R = A \times B$, которое называют отношением минимум.

Следствие D обобщенного правила вывода можно вывести из посылки 1 и посылки 2 на основании max-min свертки нечеткого множества A^1 и R_m:

$$D = A^1 \circ R_m = A^1 \circ \left[(A \times B) \cup (\sim A \times C)\right] \equiv$$
$$\equiv \left\{ v \middle| \bigvee_u \left\{ \mu_A(u) \wedge \left[\left(\mu_A(u) \wedge \mu_B(v)\right) \vee \left(\left(1 - \mu_A(u)\right) \wedge \mu_C(v)\right) \right] \right\} \right\}.$$

Пример

Рассмотрим пример. Найдем следствие правила о температурах. Пусть множество U соответствует температурам, а множество V — времени нагрева:

$U = \{30, 40, 50, 60\}$, $V = \{30, 60, 120, 240\}$ [мин].

Введем нечеткие переменные A (низкая температура), A^1 (слишком низкая температура) и B (большое время нагрева):

$$A = \{(30|1.0), (40|0.8), (50|0.2), (60|0)\},$$
$$A^1 = \{(30|1.0), (40|0.2), (50|0), (60|0)\},$$
$$B = \{(30|0), (60|0.1), (120|0.8), (240|1.0)\}.$$

Следствие D определяется с помощью формулы

$$D = A^1 \circ [A \times B].$$

Определяем произведение $A \times B$:

$$A \times B = \begin{array}{c} \\ u_1 \\ u_2 \\ u_3 \\ u_4 \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{bmatrix} 0 & 0.1 & 0.8 & 1.0 \\ 0 & 0.1 & 0.8 & 0.8 \\ 0 & 0.1 & 0.2 & 0.2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Отсюда

$$\mu_D(v) = \begin{bmatrix} 1.0 & 0.2 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0.1 & 0.8 & 1.0 \\ 0 & 0.1 & 0.8 & 0.8 \\ 0 & 0.1 & 0.2 & 0.2 \\ 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0.1 & 0.8 & 1.0 \end{bmatrix}$$

Таким образом, $D = \{(30|0), (60|0.1), (120|0.8), (240|1.0)\}$, т. е. время нагрева действительно большое.

. . . . . .

Отметим, что в лингвистической логике существует несколько вариантов построения нечетких выводов. Рассмотренный вариант, основанный на применении максиминного отношения, не всегда совпадает с нашей интуицией.

В нечеткой среде в виде нечетких понятий и отношений могут быть выражены все элементы задачи.

Возникновение нечеткого описания задачи возможно в следующих случаях.

1) Ограничения на ресурсы моделирования (временные, стоимостные) не позволяют получить в принципе существующую четкую информацию и вынуждают системных аналитиков воспользоваться знаниями экспертов, которые выражаются последними в нечеткой словесной форме. В результате обычная задача оказывается погруженной в нечеткую среду.
2) Имеющаяся числовая информация не позволяет найти решение формальными методами при существующих ограничениях на ресурсы, но ЛПР тем не менее находит, пользуясь своим опытом, который он может передать другому ЛПР в виде совокупности нечетких правил.
3) На ранних стадиях проектирования сложных (быть может, ранее не создававшихся) объектов, созданных на том или ином пути проектирования. Ресурсы на проработку всех вариантов проекта отсутствуют, а опыт конструкторов выражается качественно. Ставится задача отсева части вариантов на основе векторного показателя качества с нечеткими оценками значений его компонентов. В данном случае задача проектирования уже в исходном виде является погруженной в нечеткую среду.

Контрольные вопросы и задания к главе 4

1) Запишите условия задачи в случае неопределенности.
2) Сформулируйте виды неопределенности описания задач.
3) Дайте характеристику физической и лингвистической неопределенностей.
4) Охарактеризуйте неоднозначность описания задач.
5) Дайте характеристику терминов, качественно характеризующих количество отсутствующей информации об элементах задачи.
6) Дайте характеристику источников (причин) возможной неоднозначности описания задачи.
7) Расскажите об особенностях знаний в БЗ.
8) Поясните смысл понятий «полнота», «непротиворечивость», «монотонность», «неточность», «неопределенность» знаний.
9) Приведите примеры вышеперечисленных понятий знаний.
10) Представьте формальное представление монотонности логических выводов.
11) Каким термином мы определяем истинность информации?
12) Для каких понятий используется количественная мера (например, функция неопределенности)?
13) Сформулируйте основные понятия о теории вероятности, возможности, свидетельства.
14) Опишите Байесовский метод.
15) Опишите метод коэффициентов уверенности.
16) Приведите примеры расчета коэффициентов уверенности логического заключения при различных логических связях $(\wedge, \vee)$ между фактов в условной части правила.
17) Расскажите об отличии теории свидетельств от Байесовского подхода и метода коэффициентов уверенности.
18) Представьте и опишите геометрическую интерпретацию распределения уверенности с закрепленными массами уверенности жестко и с неизвестными массами уверенности.
19) Запишите функцию и свойства распределения уверенности теории Демпстера-Шейфера.
20) Дайте характеристику согласованному распределению уверенности и возможности.
21) Определите понятие «нечеткое» множество.
22) На чем основывается логика человеческих рассуждений?
23) Расскажите о заслугах Л. Заде в разработке общей теории n-местных логик.

24) Опишите S и π — функции принадлежности (математически и графически).
25) Определите понятие «множество нечетких подмножеств» и его свойство и сравните с множеством четких подмножеств.
26) Сформулируйте простейшие операции над нечеткими подмножествами.
27) Рассмотрите расчетные формулы расстояний для нечетких подмножеств.
28) Опишите понятие «нечеткое отношение» и приведите пример расчета проекций нечетких отношений.
29) Сформулируйте композицию двух нечетких отношений и расчете max-min композиции.
30) Приведите примеры приближенных рассуждений на основе modus ponens и обобщенного modus ponens.
31) Почему мы можем получить из одной посылки A? различные выводы в обобщенном modus ponens?
32) Определите понятие «лингвистическая переменная» и приведите пример.
33) Приведите пример модификатора термина или комбинации терминов. В каких случаях возникает нечеткое описание задачи?

Литература к главе 4

[1] Достоверный и правдоподобный вывод в интеллектуальных системах / В. Н. Вагин [и др.] ; под ред. В. Н. Вагина и Д. А. Поспелова. — М. : Физматлит, 2004. — 704 с.

[2] Заде Д. А. Понятие лингвистической переменной и его применение к принятию приближенных решений / Д. А. Заде. — М. : Мир, 1976. — 168 с.

[3] Бондарев В. Н. Искусственный интеллект: учеб. пособие для вузов / В. Н. Бондарев, Ф. Г. Аде. — Севастополь: Изд-во севНТУ, 2002. — 615 с.

[4] Кофман А. Введение в теорию нечетких множеств : пер. с франц. / А. Кофман. — М., Радио и связь, 1982. — 432 с.

[5] Левин Р. Практическое введение в технологию искусственного интеллекта и экспертных систем с иллюстрацией на Бейсике : пер. с англ. / Р. Левин, Д. Дранг, Б. Эдельсон. — М. : Финансы и статистика, 1991. — 239 с.

[6] Нариньяни А. С. Недоопределенность в системе представления и обработки знаний // Изв. АН СССР. Техн. Кибернетика. — 1986. — №5. — С. 3–28.

[7] Нечеткие множества в моделях управления и искусственного интеллекта / под ред. Д. А. Поспелова. — М. : Наука, 1986. — 369 с.

[8] Нечеткие множества в моделях управления и искусственного интеллекта / А. Н. Аверкин [и др.] ; под ред. Д. А. Поспелова. — М. : Наука, 1986. — 312 с.

[9] Обработка нечеткой информации в системах принятия решений / А. Н. Борисов [и др.]. — М.: Радио и связь, 1989. — 304 с.

[10] Представление и исследование знаний : пер. с япон. / Х. Уэно [и др.] ; под ред. Х. Уэно, М. Исудзука. — М. : Мир, 1989. — 220 с.

[11] Рыжов А. П. Элементы теории нечетких множеств и измерения нечеткости [электронный ресурс] / А. П. Рыжов. — М. : Диалог — МГУ, 1998. — Режим доступа: ryjov@mech.math.msu.su.

[12] Штовба С. Д. Введение в теорию нечетких множеств и нечеткую логику [электронный ресурс] / Д. С. Штовба. — Режим доступа: http://www.matlab.ru/fuzzylogic/book1/index.asp.

[13] Яхъяева Г. Э. Нечеткие множества и нейронные сети : учеб. пособие / Г. Э. Яхъяева. — М. : Интернет-Университет Информационных технологий; БИНОМ. Лабораторные знания, 2006. — 316 с.

[14] Shafer G. A. Mathematical Theory of Tvidence / G. A. Shafer. — Princeton: Princeton Univ. Press, 1976. — 297 p.

Глава 5

ПРОДУКЦИОННЫЕ СИСТЕМЫ

Термин «продукция» принадлежит американскому логику Э. Посту. В понимании Поста в качестве продукции выступала только та ее часть, которую теперь называют ядром. Запись «ЕСЛИ А, ТО В» трактовалась как оператор замены цепочки А цепочкой В в некотором входном слове, т. е. продукции, были тем, что позже стали называть подстановками и использовать при описании различных уточнений понятия алгоритма. Идея продукции используется в языках логического программирования, нашедших применение в СИИ. Среди языков подобного типа наиболее известен Пролог. Продукционный подход стал стилем нового этапа программирования в 90-е годы.

Продукции наряду с фреймами являются наиболее популярными средствами представления знаний в ИС. Продукции, с одной стороны, близки к логическим моделям, что позволяет организовать на них эффективные процедуры вывода, а с другой стороны, более наглядно отражают знания, характерные для логических исчислений, что дает возможность изменять интерпретацию элементов продукции.

Продукционные модели в настоящее время широко используются в системах представления знаний и были впервые применены в СИИ в 1972 г.

5.1 Представление продукционных систем

Продукционные модели могут быть реализованы как процедурно, так и декларативно. Их простота и строгая форма послужили основой ряда интересных свойств, что сделало их удобным средством представления знаний. Рядом исследователей отмечалось, что использование продукционных моделей имеет уже само по себе особую психологическую важность, хотя они могут быть с успехом использованы и вне рамок психологического моделирования.

В том практическом смысле, который представляет для нас интерес, продукция есть правило-продукция, представляющая собой пару: ситуация→действие, посылки→заключение, причина→следствие и т. п. Подобного рода правила

встречаются в различных областях знаний и видах деятельности; в повседневной жизни мы постоянно окружены различного рода правилами поведения, уличного движения, грамматическими правилами.

Если говорить о программировании, то продукция выступает как тройка (имя продукции, условие применимости, оператор). В некоторых случаях продукция близка по смыслу импликации «если-то», так что можно принять для продукции обозначение в виде импликации.

$$\alpha \to \beta,$$

а если требуется раскрыть более подробно условие применимости, то можно использовать запись следующего вида:

$P_1 \wedge P_2 \wedge \ldots \wedge P_n \to B$, где $P_i (i = 1, 2, \ldots, n)$ — условия применимости, образующие конъюнктивную форму; B — заключение, которое может трактоваться и как действие (что существенно отличает такие продукции от импликаций).

Приведем пример.

Правило-продукция из области эксплуатации оборудования: «Если температура газа $T > 30\,^{\circ}\mathrm{C}$ и давление $49 \cdot 10^4$ Па, то нарушен режим, поэтому необходимо включить вентилятор и выключить жидкостный насос».

Модульное представление продукционной системы. Если говорить о продукционной системе безотносительно к пользователю, то она выступает как программная система, которая может быть представлена состоящей из трех модулей, или блоков:

- глобальная база данных (просто база, или Б-Модуль);
- множество правил-продукций (П-Модуль);
- система управления или интерпретатора (У-Модуль).

Связь модулей показана на рис. 5.1 [4–6].

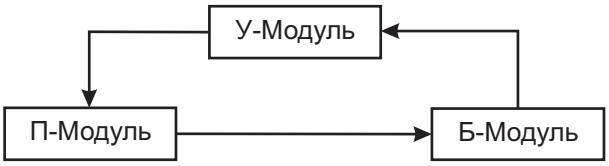


Рис. 5.1 – Продукционная система как динамическая система.

Глобальную базу данных не следует путать с базой данных в общепринятом смысле слова. Это скорее некая рабочая зона базы данных, которую можно рассматривать как динамическую систему, изменяющую свое состояние $x(t)$ под воздействием правил-продукций. Множество правил-продукций (П-Модуль) представляет собой базу знаний системы. Из П-Модуля система управления выбирает по определенной стратегии нужные продукции для воздействия на глобальную базу данных и перевода ее из состояния $x(t)$ в состояние $x(t + 1)$. Продукционная система функционирует, пока не дойдет до терминального состояния или не остановится из-за отсутствия в данной ситуации продукции. Терминальное состояние может означать, что задача решена. В то же время, если терминальное состояние определено, как состояние, которое наступает не более чем заранее обусловленное число тактов, то факт достижения этого состояния за указанное число тактов означает, что задача продукционной системы не решена. Стимулом действия системы

управления является различие между текущим $x(t)$ и терминальным состоянием. Таким образом, формально функционирование продукционной системы можно записать следующим образом:

$$x(t+1) = f\big(x(t), u_i(x)\big),$$

где $u \in U$; U — множество правил-продукций.

В классических продукционных системах БД представляет собой переменную часть системы, в то время как правила и интерпретатор чаще всего не меняются. Будучи реализованы процедурально, классические продукционные модели обладают весьма привлекательным свойством модульности. Поэтому правила могут быть добавлены или удалены без возникновения неожиданных побочных эффектов. Причина этого заключается также в том, что в классических системах вызов процедур осуществляется только в зависимости от состояния данных; процедуры, как правило, не активируются другими процедурами. Поэтому продукционные системы могут быть с большим успехом использованы для областей знаний, о которых располагаем только некоторым набором независимых правил (эвристик), а не четкой теорией, вполне завершенной и последовательной, и где поэтому нет алгоритмов, прямо приводящих к цели.

Продукционные системы, в которых сначала анализируется антецендентная часть (условная), имеют так называемую условно-выводную архитектуру (МЕТА — ДЕНДРАЛ).

Альтернативным типом архитектуры, которая достаточно часто используется в экспертных системах, являются целе-выводимые продукционные системы.

Рассмотрим упрощенный пример продукционной системы с целе-выводимой архитектурой. Буквами здесь обозначены элементы БД, и они считаются истинными, если содержатся в ней.

БД: A, F.

Правило 1: $A \wedge B \wedge C \to D$.

Правило 2: $D \wedge F \to G$.

Правило 3: $B \to C$.

Правило 4: $F \to B$.

Правило 5: $G \to H$.

Предположим, что цель состоит в том, чтобы вывести истинность H. В первую очередь проверяется, находится ли H в БД? Так как в данном случае это не так, то система пытается вывести истинность H, используя правила, имеющие H в правой части. Таким является правило 5. Теперь система пытается вывести истинность G, так как истинность последнего влечет за собой H. Снова проверяется БД: в БД нет G, следовательно, организуется поиск правила, содержащего G в правой части. Такое правило 2, поэтому целью теперь становится вывести истинность D и F. Для этого достаточно показать, что A истинно (так как A находится в БД), B — истинно (согласно правилу 4, так как B находится в БД), C — истинно (согласно правилу 3). Так как истинность D и F доказана, то из правила 2 следует истинность G, а из истинности G — следует истинность H (правило 5). Таким образом, цель достигнута.

В общем виде под продукцией понимается выражение следующего вида:

$$(i); Q; P; A \to B; N.$$

Здесь i — имя продукции, с помощью которого данная продукция выделяется из всего множества продукций. В качестве имени может выступать лексема, отражающая суть данной продукции (например, «покупка книги»), или порядковый номер продукции.

Элемент Q характеризует сферу применения продукции. Основным элементом продукции является ядро $A \to B$. Интерпретация ядра продукции может быть различной и зависит от того, что стоит справа и слева от знака секвенции. Обычное прочтение: ЕСЛИ A, ТО B, но может быть и более сложное, например, ЕСЛИ A, ТО B_1, ИНАЧЕ B_2. Секвенция может использоваться в обычном логическом смысле как знак логического следования B из истинного A (если A не является истинным выражением, то о B ничего сказать нельзя). Возможны и другие интерпретации ядра продукции, например A описывает некоторое условие, необходимое для того, чтобы можно было совершить любое действие B.

Классификация ядер продукции. Ядра продукции можно классифицировать по различным основаниям. [7–10] Прежде всего, все ядра делятся на два больших типа: детерминированные и недетерминированные. В детерминированных ядрах при актуализации ядра и при выполнимости A правая часть ядра выполняется обязательно; в недетерминированных ядрах B может выполняться и не выполняться. Таким образом, секвенция $\to$ в детерминированных ядрах реализуется с необходимостью, а в недетерминированных — с возможностью. Интерпретация ядра в этом случае может, например, выглядеть так: ЕСЛИ A, ТО ВОЗМОЖНО B.

Возможность определяется некоторыми оценками реализации ядра. Если задана вероятность выполнения B при актуализации A, то продукция будет такой: ЕСЛИ A, ТО C ВЕРОЯТНОСТЬЮ P РЕАЛИЗОВАТЬ B. Оценка реализации ядра может быть лингвистической, связанной с понятием терм-множество лингвистической переменной, например: ЕСЛИ A, ТО C БОЛЬШОЙ ДОЛЕЙ ВЕРОЯТНОСТИ B. Существуют и другие оценки реализации ядра.

Детерминированные продукции могут быть однозначными и альтернативными. Во втором случае в правой части ядра указываются альтернативные возможности выбора, которые оцениваются специальными весами выбора. В качестве таких весов могут использоваться вероятностные оценки, лингвистические оценки, экспертные оценки и т. п. (например, ЕСЛИ A, ТО ЧАЩЕ ВСЕГО НАДО ДЕЛАТЬ B_1, РЕЖЕ B_2).

Особым типом являются прогнозирующие продукции, в которых описываются последствия, ожидаемые при актуализации A, например: ЕСЛИ A, ТО С ВЕРОЯТНОСТЬЮ P МОЖНО ОЖИДАТЬ B.

Дальнейшую классификацию ядер продукций можно провести, опираясь на типовую схему ИС (рис. 5.2). Если x и y обозначают любой из блоков рисунка ($О$, $Д$, $З$, $Л$), то ядро $A_x \Rightarrow B_y$ означает, что информация об A берется из блока x, а результат срабатывания продукции B посылается в блок y. Комбинации x и y, осмысленные с точки зрения ИС, отмечены в табл. 5.1 знаком «+».

Рассмотрим часто встречающийся тип продукции $A_3 \Rightarrow B_3$. В этом случае A_3 и B_3 представляют собой определенные фрагменты информации, хранящейся в базе знаний. При сетевом представлении это могут быть фрагменты семантической сети, при логических моделях — формулы того или иного исчисления. Тогда смысл продукции $A_3 \Rightarrow B_3$ состоит в дополнении или замене одного фрагмента БЗ другим.

Таблица 5.1 – Возможные комбинации связи в ядре продукции.

A \ B	о	д	з	л
о		+		+
д	+	+	+	+
з	+		+	+
л	+		+	+

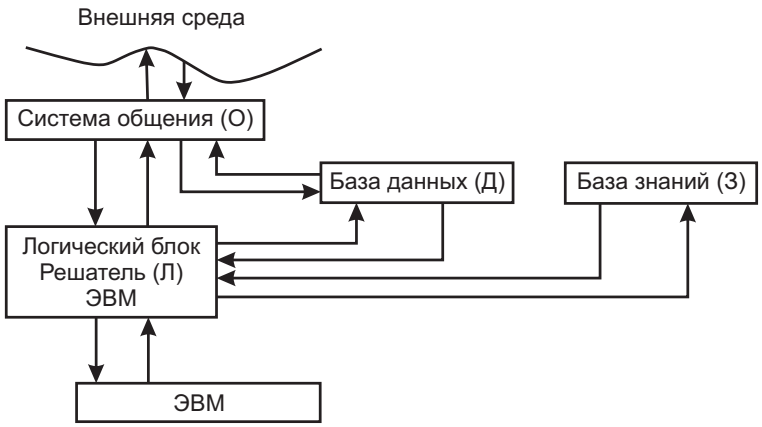


Рис. 5.2 – Типовая схема ИС.

Для актуализации необходимо, чтобы в БЗ существовал фрагмент, совпадающий с *A*. При поиске в БЗ *A* играет роль образца, а процедура такого поиска называется поиском по образцу.

Элемент *P* есть условие применимости ядра продукции. Обычно *P* представляет собой логическое выражение (как правило, предикат). Когда *P* принимает значение «истина», ядро продукции активизируется. Если *P* ложно, то ядро продукции не может быть использовано. Например, если в продукции «НАЛИЧИЕ ДЕНЕГ; ЕСЛИ ХОЧЕШЬ КУПИТЬ ВЕЩЬ *X*, ТО ЗАПЛАТИ В КАССУ ЕЕ СТОИМОСТЬ И ОТДАЙ ЧЕК ПРОДАВЦУ» условие применения ядра продукции ложно, т. е. если денег нет, то применить ядро продукции невозможно.

Элемент *N* описывает постусловия продукции. Они актуализируются только в том случае, если ядро реализовалось. Постусловия продукции описывают действия и процедуры, которые необходимо выполнять после реализации *B*. Например, после покупки некоторой вещи в магазине необходимо в описи товаров, имеющихся в этом магазине, уменьшить товар этого типа на единицу.

Правило продукции, как уже было сказано, имеет имя, посылку и заключение.

Посылка правила (левая часть) может представлять собой любое логическое выражение. Это может быть единственное условие или несколько условий. Посылка правила может включать: переменные, рабочие переменные, переменные поля, переменные ячейки, переменные утилиты, переменные среды, функции (числовые, строчные, логические), операторы (числовые, отношения, булевы).

После определения посылки правила указываются необходимые действия СИИ, если посылка верна.

Требуемые действия могут представлять собой последовательность одной или

нескольких команд системы. Это могут быть команды присвоения значения переменной, команды ввода или вывода.

Кроме этих традиционных команд, правила могут включать следующие команды системы:

- обращение к другим наборам правил;
- обработка электронных ведомостей;
- управление реляционной БД;
- управление формами;
- статистический анализ;
- общей обработки текста;
- генерации отчетов;
- выполнение команд ОС и внешних программ.

Кроме имени, посылки и заключения, каждое правило может иметь и другие характеристики. Эти варианты включают приоритет, стоимость, комментарии, последовательность готовности, список переменных потребностей, список переменных изменений, причину и стратегию тестирования для оценки посылки правила. При определении правила все варианты или некоторые из них могут игнорироваться.

Подчеркнем еще раз. В системах продукций знания представляются с помощью набора правил «если A то B». Здесь A и B могут пониматься как ситуация-действие, причина-следствие, условие-заключение и т. д. Часть правила продукции записывается с использованием знака логического следования:

$$A \to B$$

Однако не следует отождествлять правило-продукцию и отношение логического следования. Дело в том, что интерпретация продукции зависит от того, что стоит слева и справа от знака логического следования. Часто под A понимается некоторая информационная структура (например, фрейм), а под B — некоторое действие, заключающееся в ее трансформации (преобразовании). Логическая интерпретация рассматриваемого выражения накладывает жесткие ограничения на смысл символов A и B, которые должны рассматриваться как ППФ. Поэтому понятие продукции шире логического следования.

Антецеденты и консеквенты правил формируются из атрибутов и значений, например:

Атрибут:	Значение:
Двигатель	Не заводится
Стартер двигателя	Не работает
Животное	Имеет перья
Животное	Птица

Любое правило состоит из одной (или нескольких пар) *атрибут-значение.* В РП продукционной системы хранятся пары атрибут-значение, истинность которых установлена в процессе решения конкретной задачи к некоторому текущему

моменту времени. Содержимое РП изменяется в процессе решения задачи. Это происходит по мере срабатывания правил.

В процессе логического вывода объем фактов в РП, как правило, увеличивается (уменьшиться он может в том случае, если действие какого-либо правила состоит в удалении фактов из РП). В процессе логического вывода каждое правило из БЗ может обрабатываться только один раз.

При описании реальных знаний конкретной предметной области может оказаться недостаточным представление фактов с помощью — *атрибут-значение*. Более широкие возможности имеет способ описания с помощью триплетов *объект-атрибут-значение*. В этом случае отдельная сущность предметной области рассматривается как объект, а данные, хранящиеся в РП, показывают значения, которые принимают атрибуты этого объекта.

Примеры триплетов:

- Собака — кличка — Граф;
- Собака — порода — ризеншнауцер;
- Собака — окрас — черный.

Одним из преимуществ такого представления знаний является уточнение контекста, в котором применяются правила. Например, правила относящиеся к объекту «Собака», должны быть применимы для собак с любыми кличками, всех пород и окрасов. С введением триплетов правила из БЗ могут срабатывать более одного раза в процессе одного логического вывода, поскольку одно правило может применяться к различным экземплярам объекта (но не более одного раза к каждому экземпляру).

В качестве примера рассмотрим правило, взятое из базы знаний экспертной системы MYCIN, предназначенной для диагностики инфекционных заболеваний.

Если

1) место выделения культуры — кровь, и
2) реакция микроорганизма грамотрицательная, и
3) форма микроорганизма — палочка, и
4) пациент относится к группе риска, то
5) с уверенностью (0.6) название микроорганизма — psendomonia aeruginos.

Условие правила состоит из четырех фактов, соединенных союзом «И». Если все четыре факта имеют место, то верным будет следствие правила. С каждым правилом связывается некоторое число, принимающее значение в диапазоне от 0 до +1, выражающее степень достоверности следствия и называемое коэффициентом уверенности. Это означает, что система работает с неточными и ненадежными фактами.

В базе знаний системы MYCIN факты представляются с помощью триплета: объект — атрибут — значение. Пример фактов приведен в таблице 5.2.

Если в процессе активации правила первые четыре факта окажутся истинными, то в базу знаний будет помещен новый факт, представляемый триплетом объект — атрибут — значение. Одним из преимуществ хранения фактов в виде триплетов является повышение эффективности процедур поиска в базе знаний, так как появляется возможность упорядочения фактов по объектам и атрибутам.

Таблица 5.2 – Представление продукции в системе MYCIN.

№ факта	Объект	Атрибут	Значение
1	Культура	Место	Кровь
2	Микроорганизм	Реакция	Грамотрицательная
3	Микроорганизм	Форма	Палочка
4	Пациент	Принадлежит к группе риска	Истина
5	Микроорганизм	Название	psendomonia aeruginos

Рассмотренная форма записи правила продукции является упрошенной и представляет лишь ядро продукции.

Мы уже говорили, что в общем случае продукционная система включает следующие компоненты:

- базу продукционных правил;
- базу данных (рабочую память);
- интерпретатор.

5.2 Интерпретатор продукционной системы

Предпосылка правила часто рассматривается как образец.

Образец — это некоторая информационная структура, определяющая обобщенную информацию (условие, состояние и т. д.) окружающей действительности, при которой активизируется правило.

Рабочая память (глобальная база данных) отражает конкретные ситуации (состояния, условия), возникающие во внешней среде. Информационная структура, представляющая конкретную ситуацию внешней среды в рабочей памяти, называется образом (иногда также образцом в РП).

Вывод по образцу. Интерпретатор реализует логический вывод. Процесс вывода является циклическим и называется поиском по образцу. Рассмотрим его в упрощенной форме. Текущее состояние моделируемой предметной области отражается в рабочей памяти в виде совокупности образов, каждый из которых представляется посредствам фактов. Рабочая память инициализируется фактами, описывающими задачу. Затем выбираются те правила, для которых образцы, представляемые предпосылками правил, сопоставимы с образами в рабочей памяти. Данные правила образуют конфликтное множество. Все правила, входящие в конфликтное множество, могут быть активизированы. В соответствии с выбранным механизмом разрешения конфликта активизируется одно из правил. Выполнение действия, содержащегося в заключении правила, приводит к изменению состояния рабочей памяти. В дальнейшем цикл управления выводом повторяется. Указанный процесс

завершается, когда не окажется правил, предпосылки которых сопоставимы с образами рабочей памяти (рис. 5.3).

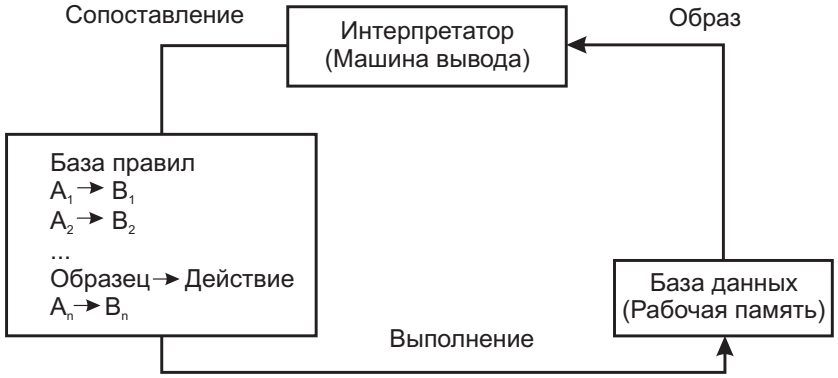


Рис. 5.3 – Продукционная система.

Таким образом, процесс вывода, основанный на поиске по образцу, состоит из четырех шагов:

- выбор образа;
- сопоставление образа с образцом и формирование конфликтного набора правил;
- разрешение конфликтов;
- выполнение правила.

Продукционные системы были предложены Постом в 1943 г. и рассматривались как модель организации вычислительного процесса. При этом правило-продукция требовалось как оператор замены одной цепочки литер в некотором слове на другую цепочку. Систему продукций можно рассматривать как формальную систему с алфавитом $A = \{a_1\, a_2\, a_3\, \ldots\, a_n\}$, конечным множеством аксиом и конечным множеством правил вывода, имеющих вид:

$$P_i\colon \alpha_i s \to s\beta_i,\quad i = 1, \ldots, k,$$

где α_i, β_i, s — слова алфавита A. Применение данного правила к некоторой цепочке литер, состоящей из слов α_i и s, означает вычеркивание α_i и приписывание β_i. Указанная замена соответствует подстановке, являющейся основным оператором нормальных алгоритмов Маркова.

Интересные результаты были получены А. Ньюэллом и Г. Саймоном при разработке системы GPS. Они установили, что правила продукции соответствуют элементам знаний, накапливаемым в долговременной памяти человека. Такие элементы памяти активизируются, если возникает соответствующая ситуация (по образцу). Новые элементы памяти могут накапливаться в памяти без необходимости перезаписывания уже существующих элементов. Рабочая память продукционных систем аналогична кратковременной памяти человека, которая удерживает в центре внимания текущую ситуацию. Содержимое рабочей памяти, как правило, не сохраняется после решения задачи. Благодаря указанной аналогии продукционные модели представления знаний получили широкое распространение в ЭС, моделирующих решение задач человеком-экспертом в той или иной области.

Управление выводом. В ходе решения проблемы машина вывода (интерпретатор) выполняет две задачи — собственно логический вывод и управление выводом. Логический вывод в продукционных системах не отличается особой сложностью и реализуется на основе процедуры поиска по образцу, описанной выше. Управление выводом в продукционных системах предлагает решение двух вопросов:

1) с чего следует начинать процесс вывода;
2) как поступить, если на некотором шаге вывода возможен выбор различных вариантов его продолжения.

Ответ на первый вопрос приводит к прямой и обратной цепочкам рассуждений, а на второй вопрос — к механизмам разрешения конфликтов в продукционных системах.

Применение процедуры поиска по образцу в продукционных системах соответствует поиску решения задач в пространстве состояний или в пространстве редукций задач. Поэтому для управления выводом в продукционных системах могут применяться стратегии слепого и упорядоченного поиска.

В настоящем разделе ограничимся рассмотрением стратегий прямого и обратного выводов (соответственно поиска, управляемого данными и целью), а также основных методов разрешения конфликтов [3].

Прямой и обратный выводы. Прямой вывод начинается с задания исходных данных решаемой задачи, которые фиксируются в виде фактов в рабочей памяти системы. Правила, применяемые к исходным данным, обеспечивают генерацию новых фактов, добавляемых в рабочую память. Процесс продолжается, пока не будет получено целевое состояние рабочей памяти.

В таблице 5.3 представлен пример поиска, управляемого данными, на множестве продукций, записанных в виде формул пропозиционной логики. Здесь на каждом шаге работы машины вывода применяется простая стратегия разрешения конфликтов: активизируется последнее из успешно сопоставленных правил. Порядок сопоставления правил соответствует их номерам.

Множество правил-продукций:

1) $G \wedge H \rightarrow C$
2) $I \wedge K \rightarrow D$
3) $L \wedge M \rightarrow E$
4) $N \rightarrow F$
5) $O \rightarrow F$
6) $C \rightarrow A$
7) $D \rightarrow A$
8) $E \rightarrow B$
9) $F \rightarrow B$
10) $A \rightarrow goal$
11) $B \rightarrow goal$

Исходные данные:
$START = \{L, M, N\}$

Таблица 5.3 – Прямой вывод по образцу.

№ шага	Рабочая память	Конфликтное множество	Активизируемое правило
0	L,M,N	—	—
1	L,M,N	3.4	4
2	L,M,N,F	3.9	9
3	L,M,N,F,B	3.11	11
4	L,M,N,F,B, goal	3	Остановка

На рис. 5.4, *а* изображен граф решения. Вершины графа соответствуют высказываниям, а ребра — соответствующим правилам вывода. Процесс вывода начинается с размещения множества исходных фактов *L*, *M*, *N* в рабочей памяти и заканчивается, когда в рабочую память будет помещена целевая вершина Goal. Направление вывода обозначено рядом с графиком и соответствует движению от стартовой к целевой вершине, т. е. от исходных данных (высказывания *L*, *M*, *N*).

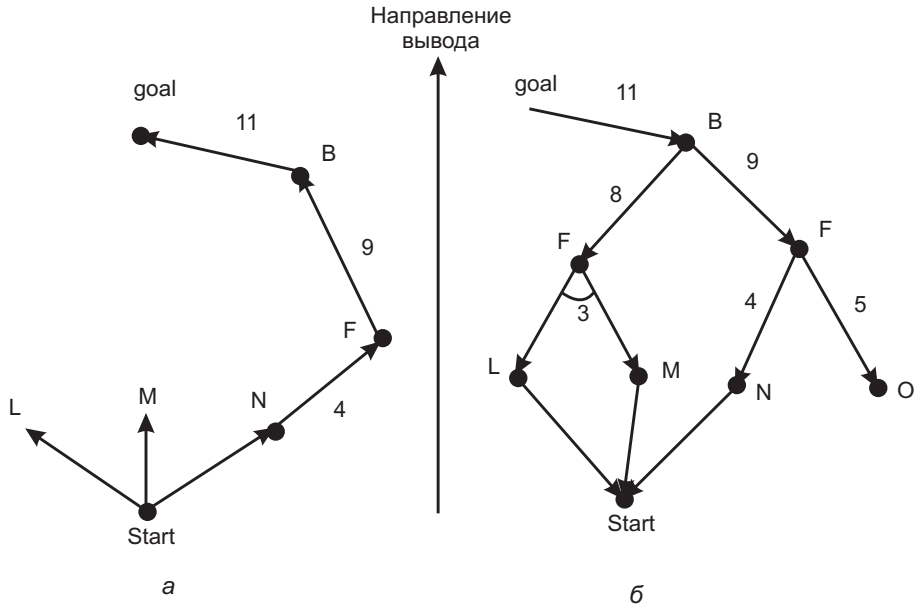


Рис. 5.4 – Граф решения задачи.

Поэтому такой вывод называется выводом, управляемым данными.

Отметим, что выбранная стратегия активизации последнего правила, добавленного в конфликтное множество, соответствует поиску в глубину. Именно эта стратегия наиболее часто применяется в экспертных системах продукционного типа, т. к. она соответствует процессу рассуждений эксперта при решении той или иной задачи. Поведение системы для пользователя выглядит довольно логичным: он видит, что система стремится свести каждую цель к подцели.

Если активизировать на каждом шаге работы машины вывода первое правило конфликтного множества, то граф решения будет строиться методом поиска в ширину (рис. 5.5). В этом случае все альтернативные цепочки рассуждений, имеющиеся на графе решений, продолжаются параллельно, и ни одна не обгоняет другую.

Безусловно, возможны и смешанные стратегии вывода, когда объединяются оба вида поиска — поиск в ширину и поиск в глубину, а также различные эвристические стратегии.

Кроме прямого вывода, в продукционных системах широко применяется и обратный вывод, т. е. вывод, управляемый целевыми условиями. Такой вывод начинается с целевого утверждения, которое фиксируется в рабочей памяти. Затем отыскивается правило-продукция, заключение которого сопоставимо с целью. Условия данного правила помещаются в рабочую память и становятся новой подцелью. Процесс повторяется до тех пор, пока в рабочей памяти не будут найдены факты, подтверждающие целевое утверждение. Проиллюстрируем обратный вывод на множестве продукций предыдущего примера.

Процесс вывода начинается с того, что в рабочую память помещается целевое утверждение goal, истинность которого необходимо подтвердить или опровергнуть, а также множество исходных фактов $\{L, M, N\}$, которые считаются истинными утверждениями.

Удобно рассматривать переменные правил как многозначные объекты, характеризуемые тремя возможными значениями: «не определено», «ложь», «истина». В этом случае начальное значение переменной goal — «не определено», а факты L, M, N имеют значение «истина».

Обратная цепочка рассуждений для рассматриваемого примера изображена на табл. 5.4. Здесь факты, имеющие значение «истина», выделены курсивом. Вывод начинается с утверждения goal, которое рассматривается как текущая подцель. Все продукционные правила, заключение которых сопоставимо с текущей подцелью, добавляются в конфликтное множество правил. На каждом шаге активизируется первое правило конфликтного множества. Посылки данного правила добавляются в рабочую память и на следующем шаге выступают в качестве новой подцели системы. Такая стратегия разрешения конфликтов соответствует поиску в ширину (табл. 5.4).

Множество правил-продукций:

1) $G \wedge H \to C$
2) $I \wedge K \to D$
3) $L \wedge M \to E$
4) $N \to F$
5) $O \to F$
6) $C \to A$
7) $D \to A$
8) $E \to B$
9) $F \to B$
10) $A \to goal$
11) $B \to goal$

Исходные данные:

$START = \{L, M, N\}$

Таблица 5.4 – Обратный вывод по образцу.

№ шага	Рабочая память	Конфликтное множество	Активизируемое правило
0	Goal,L,M,N	10,11	10
1	Goal,A,L,M,N	11,6,7	11
2	Goal,A,B,L,M,N	6,7,8,9	6
3	Goal,A,B,C,L,M,N	7,8,9,1	7
4	Goal, A,B,C,D,L,M,N	8,9,1,2	8
5	Goal, A,B,C,D,E,L,M,N	9,1,2,3	9
6	Goal, A,B,C,D,E,F,L,M,N	1,2,3,4,5	1
7	Goal, A,B,C,D,E,F,G,H,L,M,N	2,3,4,5	2
8	Goal, A,B,C,D,E,F,G,H,I,KL,M,N	3,4,5	3
9	Goal, A,B,C,D,E,F,G,H,I,KL,M,N	4,5	остановка

Указанный процесс повторяется до тех пор, пока все условия (посылки) некоторого правила не станут истинными, т. е. совпадут с фактами, имеющимися в рабочей памяти. В этом случае заключение правила тоже получает значение «истина». Данное истинное значение распространяется вверх по цели активизированных правил. Если целевая вершина получает значение «истина», то процесс поиска на этом заканчивается. Если этого не происходит, то вновь выбирается первое правило конфликтного множества и т. д.

На рис. 5.5 при активизации правила 3 устанавливается, что его посылки *L* и *M* являются фактами. Следовательно, фактом является и заключение *E*. Далее, если верно *E*, то верно и *B* (правило 8). И, наконец, если верно *B*, то справедливо и целевое утверждение *goal* (правило 11).

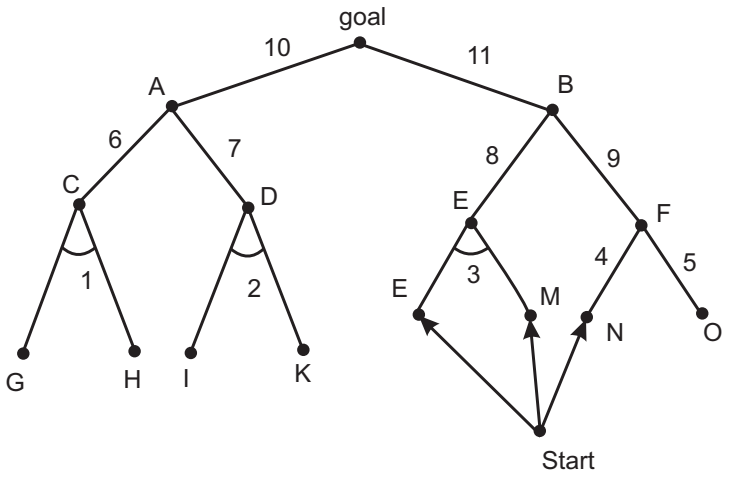


Рис. 5.5 – Обратный вывод в продукционных системах.

Рассмотренные примеры показывают, что вывод в продукционных системах в общем случае соответствует поиску решений в И-ИЛИ графах. В частном случае, когда правила-продукции не содержат в условной задаче конъюнкций, вывод в продукционных системах будет соответствовать поиску решений в пространстве состояний. Часто в продукционных системах применяют комбинированные мето-

ды поиска решений. Например, применяют двунаправленный поиск. В этом случае сначала поиск ведут в прямом направлении до тех пор, пока число состояний не станет большим. Затем выполняется обратный поиск уже достигнутых состояний (подцелей). При такой организации поиска имеется одна опасность. Если применяются методы эвристического поиска, то, возможно, что подграфы, обследуемые в прямом и обратном направлениях, окажутся разными и не «встретятся». Это приведет к увеличению времени поиска по сравнению с однонаправленным поиском. Этой опасности нет, если применяются методы «слепого перебора». Тогда комбинированный метод значительно сокращает время поиска.

Отметим, что ситуация, когда в рабочую память заранее вносятся все известные факты, встречается редко. Обычно в рабочей памяти изначально содержится только часть фактов, необходимых для вывода. Остальные сведения продукционная система выясняет сама, задавая вопросы пользователю, опрашивая измерительные или иные подсистемы, сопряженные с ней.

Например, следуя цепочке обратных рассуждений, изображенной на рис. 5.5, система направляет поиск от цели к фактам. Когда будет активизировано правило 1, то система пытается установить, являются ли высказывания G и H фактами, т. е. приписано ли им значение «истина». Так как значения высказываний G и H неизвестны, то система может задать пользователю вопрос относительно истинности высказываний G и H. Полученный ответ вносится в рабочую память. Если высказывания G и H получат значения «истина», то это изменит ход рассуждений системы и произойдет отказ от анализа других цепочек правил.

Сравнивая прямой и обратный выводы, следует отметить, что прямой вывод имеет более широкую область применения. Объясняется это тем, что обратный вывод будет эффективен в том случае, если решаемая проблема характеризуется немногими, четко определенными целевыми состояниями. Когда количество целевых состояний велико или их определение является частью самой проблемы, то обратный вывод малопригоден. Подобные ситуации, где план является одновременно и целевым состоянием, и искомым решением, имеют место в медицинских ЭС при назначении методики решения и т. д.

Поиск решений. При создании ИИС используют различные модели представления знаний: логические, продукционные, сетевые. Наибольшее распространение получили ЭС, основанные на продукционных правилах. Рассмотрим особенности управления поиском решений в таких системах, а так же выясним, как формулируется объяснение принятых решений.

БЗ ЭС продукционного типа состоит из множества правил вида

$$\text{Если } C, \text{ то } X,$$

где C — последовательность решений (образцов), определяющих условия активизации правила; X — последовательность действий, которые могут модифицировать состояние рабочей памяти.

Вывод, основанный на поиске по образцу: выбор образа; сопоставление образа с образцом и формирование конфликтного набора правил; разрешение конфликтов; выполнение правила.

Продукционное правило напоминает условный оператор процедурных языков программирования. Однако несмотря на внешнюю схожесть с условным

оператором, продукционные правила характеризуются следующими отличительными признаками:

- действия X выполняются, когда выражения (образцы) из C сопоставимы с соответствующими элементами (фактами) рабочей памяти;
- возможно одновременное сопоставление выражения из C с несколькими элементами рабочей памяти, что порождает необходимость запоминания экземпляров потенциально конфликтующих правил;
- действия X могут заключаться как в добавлении новых элементов в рабочую память (БД), так и в исключении существующих элементов;
- правила часто содержат дополнительную управляющую информацию, например: название модуля БЗ, к которому относится правило; приоритет правила; фаза выполнения, в течение которой разрешается выполнять правило.

Алгоритм поиска решения в ЭС продукционного типа использует понятие списка конфликтных правил S. К конфликтным правилам относят все экземпляры правил, условная часть которых сопоставима с соответствующими элементами рабочей памяти. Обобщенный алгоритм прямого вывода на правилах-продукциях можно записать так.

- Begin
- Выбрать экземпляр правила из S.
- Изменить состояние рабочей памяти в соответствии с заключением правила.
- Модифицировать список S, добавив или удалив экземпляры правил в соответствии с изменением состояния рабочей памяти.
- End

Здесь задание начальных условий цикла сопряжено со значительными объемами вычислений, так как формирование списка конфликтных правил S требует сопоставления всех фактов РП со всеми выражениями (образцами) из предпосылок всех правил. Отметим, что один и тот же образец правила может быть сопоставлен с несколькими фактами. Поэтому список S состоит не из правил, а из экземпляров правил, которые представляют собой частные случаи правил с конкретными подстановками. В ходе выполнения алгоритма нет необходимости повторять сопоставление всех фактов РП и всех образцов правил. На каждой итерации анализируются только изменения состояния РП, связанные с выполнением последнего правила.

5.3 Эффективность поиска решений в продукционных системах

Рассмотрим сравнение с образцом образов (объектов) рабочей памяти. Существует много разных видов соответствия образцу, но все они требуют, чтобы программа могла распознать, что одно выражение E, скажем (Петр любит Lisp), является примером другого, F, скажем (Петр любит X). Мы можем увидеть в этой форме два различных уровня — синтаксический (относящийся к форме выражения) и семантический (относящийся к его смыслу). Синтаксически E — это то же, что F,

если мы осуществим подстановку в *F* Lisp вместо *X*. Семантически любовь Петра к Lisp является примером любви к чему-либо; таким образом, *E* «соответствует» *F*, так как *X* означает все, что любит Петр. В этом примере *E* — это «объект», а *F* — «образец», так как *F* содержит переменную. Это соответствие не работало бы, если бы в качестве объекта мы взяли *F*, а в качестве образца *E*.

Рассмотрим следующее правило продукций:

IF Петр любит *X* & *X* — язык программирования

THEN Петр использует *X*.

Если дано, что Петр любит Lisp и что Lisp — язык программирования, мы вправе сделать вывод, что Петр использует Lisp. Однако чтобы сделать этот вывод, программа должна обнаружить, что условия (Петр любит *X*) и (*X* — язык программирования) удовлетворены для некоторого значения *X*. Таким образом, в поисках кандидата на соответствие условиям программа должна обратиться к набору, в котором хранятся выражения, а затем тщательным образом убедиться, что найденные кандидаты действительно соответствуют выражениям, не забывая о необходимости подстановки для преобразования одного выражения в другое.

Трудности, связанные с поиском соответствия образцу, заключаются в достаточно больших затратах на вычисления. Размер обрабатываемого списка при поиске вхождения одного списка в другой является функцией длины и вложенности этих двух списков, а также числа переменных в образце, поскольку каждый раз, когда вы ставите переменной в соответствие константу, вы должны быть уверены, что новая связь согласована с предыдущей. В системах продукций вы имеете дело с двумя наборами списков, например набором элементов рабочей памяти и набором левых частей правил продукций. При этом вы пытаетесь определить те левосторонние правила, которые удовлетворяют содержимому рабочей памяти. Это называется соответствием многих образцов многим объектам.

Здесь заложено несколько возможных источников неэффективности.

1) Интерпретатор может просто просматривать рабочую память, проверяя все списковые структуры по очереди для каждого правила.
2) Интерпретатор может только проверять некоторые подмножества рабочей памяти для каждого правила, отыскивая соответствия только элементам, помеченным как наиболее вероятные кандидаты на соответствие этому правилу, но при этом все же для N правил необходимо N проходов рабочей памяти.
3) При последовательных циклах распознавания некоторые объекты могут ставиться в соответствие некоторым образцам многократно, каждый раз вызывая обработку одного и того же списка.

Многие из созданных ранее систем продукций тратили свыше 90 % своего времени на поиск соответствия образцам, организованный таким способом. Даже когда для хэширования элементов рабочей памяти использовались различные виды индексирования, то все равно оставались итерации по рабочей памяти внутри и между циклами.

В качестве основного механизма поиска соответствия образцу для OPS-интерпретаторов стал использоваться алгоритм соответствия RETE (предложенный Форге). В основе его лежат два фундаментальных соображения:

1) Левые части продукций часто имеют одинаковые условия, и примитивный подход будет стремиться удовлетворить эти условия на рабочую память N раз для N вхождений. Это пример внутрицикловой итерации (частичного сопоставления).
2) Рабочая память только модифицируется, причем совсем немного в каждый отдельный момент времени, в то время, как примитивный алгоритм стремится найти соответствие всем образцам по всем элементам рабочей памяти для каждого цикла. Это пример межцикловой итерации (сопоставление между правилами).

Описываемый алгоритм сокращает число внутрицикловых итераций на продукциях путем использования сортирующей сети с древовидной структурой. Образцы в левой части продукций компилируются в эту сеть, и алгоритм соответствия вычисляет набор для заданного цикла путем обработки этой сети. Межцикловая итерация на рабочей памяти сокращается с помощью обработки токенов, указывающих, какие образцы соответствуют каким элементам рабочей памяти, и затем производится просто модификация этого набора при изменении рабочей памяти.

Приведем упрощенный пример работы алгоритма RETE.

Для того чтобы определить экземпляры правил, которые можно будет активизировать в дальнейшем и, следовательно, включить в список S, машина вывода систем, основанных на правилах, выполняет на этапе инициализации частичное сопоставление.

Частичное сопоставление осуществляется, когда часть образцов предпосылки правила сопоставима с фактами РП. Если частичные сопоставления были запомнены на этапе инициализации, то на очередной итерации сопоставляются только оставшиеся образцы из предпосылки экземпляра правила. Частичное сопоставление позволяет повысить скорость вывода.

Пример

Пусть имеется правило $(P_1 \wedge P_2 \wedge \ldots \wedge P_8) \rightarrow Q$

На этапе инициализации для этого правила были успешно сопоставлены образцы $P_1, \ldots, P_6$, и соответствующие подстановки запомнены машиной вывода. После выполнения очередного правила из списка S машина вывода определяет, имеются ли в РП изменения, оказывающие влияние на результаты частичного сопоставления для рассматриваемого правила (т. е. на $P_1, \ldots, P_6$). Если изменения не касаются, например, образцов $P_1, P_2, \ldots, P_5$, но делают невозможным сопоставление для образца P_6, то машина вывода сохраняет подстановку для образцов $P_1, P_2, \ldots, P_5$. Если же изменения не касаются образцов $P_1, P_2, \ldots, P_6$ и дополнительно успешно выполняется сопоставление для образцов P_7 и P_8, то рассматриваемое правило помещается в список активизируемых правил.

Дальнейшее повышение эффективности операций сопоставления основано на распространении результатов частичного сопоставления между правилами. Это

позволяет исключить многократное сопоставление, по сути, одних и тех же образцов, встречающихся в различных правилах. Пусть имеются два правила:

$$P_0 \wedge P_1 \wedge P_2 \wedge P_3 \to Q_1$$
$$P_0 \wedge P_1 \wedge P_4 \wedge P_2 \to Q_2$$

Тогда результаты сопоставления образцов P_0 и P_1 для первого правила могут быть распространены на второе правило. Но подстановки, сделанные в ходе сопоставления образца P_2 первого правила, не могут использоваться во втором правиле из-за того, что раньше должно выполниться сопоставление образца P_4, которое может сделать недопустимым запомненные подстановки для P_2.

В соответствии с алгоритмом RETE, в результате компиляции правил формируется сеть, состоящая из двух частей. Первая часть сети представляет сеть образцов, а вторая часть — сеть объединения. Рассмотрим следующие правила, в которых образцы представлены триплетами «объект-атрибут-значение»:

$$\big(P_0(\text{длина ? } x)(\text{высота ? } x)\big)\big(P_1(\text{длина ? } x)(\text{ширина ? } y)\big) \to Q_1$$
$$\big(P_0(\text{длина ? } y)(\text{высота ? } y)\big)\big(P_1(\text{длина ? } x)(\text{ширина ? } y)\big) \to Q_2$$

В ходе сопоставления в переменные, начинающиеся символом «?», выполняются подстановки. Вхождениями одной и той же переменной в различные образцы должны назначаться одинаковые значения. Если переменная входит несколько раз в один и тот же образец, то она получает значения в соответствии с сетью образцов. Если переменная входит в различные образцы, то значения назначаются с помощью сети объединения (рис.5.6).

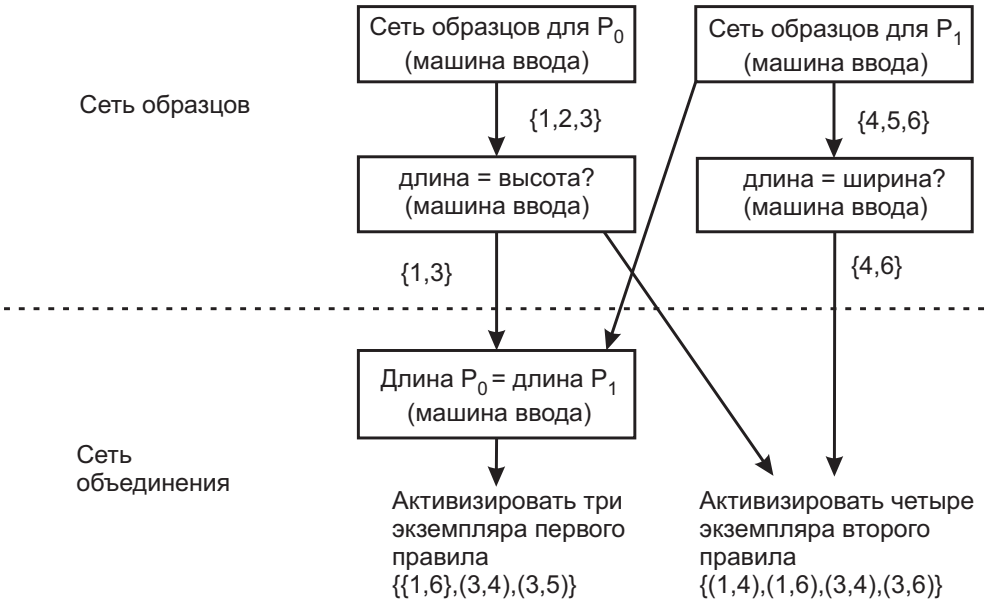


Рис. 5.6 – Сеть, формируемая алгоритмом RETE.

Предположим, что в РП имеются следующие факты, относящиеся к объектам P_0 и P_1

1) $\big(P_0(\text{длина } 12)(\text{высота } 12)\big)$;

2) $\big(P_0(\text{длина } 15)(\text{высота } 10)\big)$;

3) $\big(P_0(\text{длина } 15)(\text{высота } 15)\big)$;

4) $\big(P_1(\text{длина } 15)(\text{ширина } 15)\big)$;

5) $\big(P_1(\text{длина } 15)(\text{ширина } 14)\big)$;

6) $\big(P_1(\text{длина } 12)(\text{ширина } 12)\big)$.

Тогда в результате сопоставления образцов и фактов РП будут активизированы три экземпляра первого правила с подстановками фактов $\{(1,6),(3,4),(3,5)\}$, и четыре экземпляра второго правила с подстановками фактов $\{(1,4),(1,6),(3,4),(3,6)\}$. Из примера видно, что с помощью сети объединения подстановки распространяются между правилами, что исключает лишние сопоставления.

На каждом шаге машина вывода выбирает экземпляр правила из множества конфликтных правил S.

5.4 Механизм разрешения конфликтов

При выполнении условия применимости ядер продукции для группы продукций возникает дилемма выбора той или иной продукции, которая в данной ситуации будет активизирована. Решение этой задачи возлагается на систему управления системой продукций. Если, например, ИС реализована на ЭВМ с параллельной архитектурой, то из фронта готовых продукций может выбираться не одна продукция, а столько, сколько параллельных ветвей может одновременно в данной ситуации выполнить ЭВМ. Но независимо от количества актуализированных продукций задача альтернативного выбора остается.

У задачи управления возможны два пути решения: централизованный и децентрализованный. При централизованном выполнении продукций решение об актуализации принимается специальной системой управления, а при децентрализованном определяется складывающейся в этот момент ситуацией. Если порядок выполнения продукций произволен, то решение определяется текущей ситуацией или системой управления; если порядок важен, то в продукциях должна содержаться информация о требованиях к этому продукту. Если в постусловиях продукций указывается имя продукции, которая должна выполняться после данной, система продукций превращается в обычную программу для ЭВМ, т. е. реализует некоторый алгоритм.

Основные стратегии. Разрешения конфликтов — важная проблема, связанная с управлением порядка применения правил, образующих конфликтное множество.

Порядок активизации правил конфликтного множества определяется выбранной стратегией разрешения конфликтов. Обычно конфликтное множество правил представляется в виде упорядоченного списка. При этом конфликтные правила дописываются в конец этого списка. Простые стратегии разрешения конфликта основаны на том, что выбирается либо первое, либо последнее правило, входящее в список. Выбор первого правила соответствует поиску в ширину, а выбор последнего правила (т. е. только что добавленного) — поиску в глубину. Во многих продукционных системах чаще всего применяют второй способ.

Другими принципами, используемыми для разрешения конфликтов, являются:

- принцип «стопки книг»;
- принцип наиболее длинного условия.

Опишем несколько стратегий управления выполнением продукций [1, 3–5].

1) Принцип «стопки книг». Основан на идее, что наиболее часто используемая продукция является наиболее полезной. При актуализации некоторого фронта готовых продукций для исполнения выбирается та продукция, у которой частота использования максимальна. Подобный принцип управления особенно хорош, когда частота исполнения подсчитывается с учетом некоторой ситуации, в которой ранее использовалась продукция, и это исполнение имело положительную оценку. Управление по принципу «стопки книг» целесообразно применять, если продукции относительно независимы друг от друга, например когда каждая из них есть правило вида «ситуация $(A) \Rightarrow$ действие B».
2) Принцип наиболее длинного условия. Заключается в выборе из фронта готовых продукций той, у которой стало истинным наиболее «длинное» условие выполнимости ядра. Этот принцип опирается на соображение «здравого смысла», что частные правила, относящиеся к узкому классу ситуаций, важнее общих правил, относящихся к широкому классу ситуаций, так как первые учитывают больше информации о ситуации, чем вторые. Трудность использования данного принципа состоит в том, что надо заранее упорядочить условия по вхождению друг в друга по отношению «частное-общее». Управление по принципу наиболее длинного условия в продукциях, образующих фронт готовых продукций, целесообразно применять в тех случаях, когда знания и сами продукции хорошо структурированы по привязкам к типовым ситуациям, на которых задано отношение типа «частное-общее».
3) Принцип метапродукций. Основан на идее ввода в систему продукций специальных метапродукций, задачей которых является организация управления в системе продукций при возможности неоднозначного выбора из фронта готовых продукций. Например, если инфекция есть pelvic-abcess и имеются продукции, входящие в состав фронта, в которых в условии A упоминается grampus-rods, то продукции, у которых в A имеется enterobacteriaceae, следует активизировать раньше, чем продукции, содержащие в A grampus-rods.
4) Принцип «классной доски». Основан на идее спусковых функций. При реализации принципа «классной доски» в ИС выделяется специальное рабочее поле памяти — аналог классной доски. На этой «доске» параллельно выполняющиеся процессы находят информацию, инициирующую их запуск, на нее же они и выносят информацию о своей работе, которая может оказаться полезной для своих процессов.

Как правило, на «классной доске» выделены специальные поля для формирования условий применимости ядер продукций, различные для разных сфер применения продукций, специальные поля для записи результатов срабатывания продукций

и для записи постусловий, если они адресованы другим продукциям. С принципом «классной доски» может комбинироваться принцип управления с помощью метапродукций, т. к. он требует проверки некоторых условий, которые фиксируются в рабочем поле памяти, а также другие принципы управления.

5) Принцип приоритетного выбора. Связан с введением статических или динамических приоритетов на продукции. Статические приоритеты могут формироваться априори на основании сведений о важности продукционных правил в данной проблемной области. Динамические приоритеты вырабатываются в процессе функционирования системы продукций и могут отражать, например, такой параметр, как время нахождения продукции во фронте готовых продукций.

Существует два типа выполнения систем продукций: прямой и обратный. В первом случае поиск идет от левых частей продукций, т. е. проверяются условия A и актуализируются те продукции, для которых A имеет место; во втором — от изначально заданных B, по которым определяются необходимые для B значения A, которые, в свою очередь, отождествляются с правыми частями ядер продукций в системе.

6) Управление по имени. Основан на задании имен продукций, входящих в некоторую систему, некоторой формальной грамматики или другой процедуры, обеспечивающей сужение фронта готовых продукций и выбор из него очередной продукции для выполнения.

Например, пусть система продукций представлена четырьмя простейшими продукциями, состоящими лишь из ядерных частей: (*а*) $A \Rightarrow B$; (*б*) $B \,\&\, D \Rightarrow A$; (*в*) $A \vee B \Rightarrow D$; (*г*) $D \Rightarrow C$. В таком виде система продукций явно недетерминирована. Если выполняется A, то фронт готовых продукций включает продукции с именами (*а*) и (*в*), а если выполняются B и D, то три последние продукции. Для устранения подобной недетерминированности может быть введена некоторая грамматика для имен продукций: (*а*)⇒(*б*); (*в*)⇒(*б*); (*а*)⇒(*г*). Тогда если в некоторый момент была выполнена продукция с именем (*б*), то новые продукции выполняться не будут. Если же в некоторый момент выполнилась продукция с именем (*а*), то после нее могут выполняться продукции с именем (*б*) и (*г*). Но левые части ядер в этих продукциях таковы, что неоднозначность возникает лишь в случае, когда B и D одновременно выполнены. С помощью грамматики для имен продукций можно задать однозначный алгоритмический процесс.

7) α-β-алгоритм (Эвристический с оценкой $f(v) = g(v) + h(v)$). С помощью этого алгоритма исходная задача сводится к уменьшению пространства состояний путем удаления в нем ветвей, неперспективных для поиска успешного решения, т. е. просматриваются только те вершины, в которые можно попасть в результате следующего шага, после чего неперспективные направления исключаются. Например, в БЗ продукционной системы, заполненной знаниями о животном мире, не следует искать животных, не относящихся к млекопитающим, в направлении, берущем начало от вершины, определяющей млекопитающих. Данная стратегия является определенным компромиссом между поиском в ширину и поиском в глубину. Для ее успешной реализации следует располагать дополнительными

эвристическими знаниями, которые используются при выборе перспективных направлений.

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

5.5 Продукционные системы в приложениях

Иерархическую абстрактную структуру удобно использовать при описании проблемной области продукционной моделью представления знаний, которая применяется для отражения динамики изменения проблемной области, когда для ее задания требуются несколько десятков тысяч продукционных правил. Иерархическая абстрактная структура позволяет разбить продукционные правила на блоки в соответствии с принадлежностью к элементам структуры, принятие решений в которых они задают, и использовать механизм наследования продукционных правил. Механизм наследования продукционных правил позволяет создать БЗ, сделать ее более компактной.

Пусть задана иерархическая структура, которая описывает некоторую абстрактную проблемную область «Аэропорт». Иерархическая структура состоит из трех уровней детализации [1]:

1 уровень — Аэропорт;

2 уровень — классы самолетов (ТУ, ИА, АН . . .);

3 уровень — представители классов (конкретные объекты).

Управление работой аэропорта задается множеством продукционных правил, которые разбиваются на блоки в соответствии с уровнем в иерархической структуре:

- в блоки 1 уровня входят продукционные правила, которые описывают принципы управления аэропортом в целом;
- в блоки 2 уровня входят продукционные правила, которые описывают управление классами;
- число блоков 3 уровня определяется количеством самолетов.

В каждый блок третьего уровня входят продукционные правила, которые задают специфические законы управления конкретным самолетом. Остальные продукционные правила, которые задают общие законы управления классами, могут быть получены благодаря механизму наследования.

Таким образом, иерархическая структура позволяет создать иерархию продукционных правил и использовать принцип наследования продукционных правил, подобно механизму наследования свойств в ISA-иерархии.

Один из способов представления используется в системах *извлечения знаний* решающего правила-продукции. Здесь в качестве посылки выступает описание объекта через его свойства, а заключением будет вывод о принадлежности объекта к определенному классу [2].

Например: если $pH < 6$, то жидкость — кислота.

Правила формируются в виде $\langle D \rangle \rightarrow \langle C \rangle$, где C — это класс, а D — условие, составленное как конъюнкция простых условий вида <атрибут>-<значение>. Условия могут быть равенствами для символьных атрибутов либо неравенствами для числовых атрибутов. Условие D определяет выборку из базы данных тех объектов, для которых оно верно.

Выборки из множеств положительных и отрицательных примеров обозначим $\delta_D(P)$ и $\delta_D(N)$ соответственно. Правило считается корректным, если оно покрывает все положительные примеры и не покрывает ни одного отрицательного, т. е. $\delta_D(P) = P$ и $\delta_D(N) = \varnothing$. На практике можно найти крайне мало корректных правил, чаще правило выполняется с той или иной степенью достоверности. Поэтому с каждым правилом R мы свяжем следующую вероятностную оценку качества:

$$Q(R) = \frac{|\delta_D(P)|}{|\delta_D(P)| + |\delta_D(N)|},$$

где $|\delta|$ обозначает мощность множества.

Очевидно, для корректного правила $Q(R) = 1$. Целью формального метода является извлечение правил с наибольшим показателем качества $Q(R)$.

Любое решающее дерево может быть преобразовано в набор продукционных правил: каждому пути от корня дерева до конечной вершины соответствует одно продукционное правило. Его посылкой является конъюнкция условий «атрибут-значение», соответствующих пройденным вершинам и ребрам дерева, а заключением имя класса, соответствующее концевой вершине.

В случае приоритетного выбора с каждой продукцией связывается статистический или динамический приоритет P_r, определяющий порядок ее активизации.

Иногда в постусловиях продукции может указываться имя следующей продукции, которую необходимо выполнить. Это превращает систему продукций в обычный алгоритм.

5.6 Объяснение выводов

Рассмотрим принципы формирования объяснения выводов в ЭС продукционного типа. Формирование объяснений выводов в интеллектуальной системе важно по многим причинам. Во-первых, конечные пользователи не смогут доверять рекомендациям системы, если они не будут уверены в том, что рассуждения выполнены корректно. Во-вторых, инженеры по знаниям должны получать полную информацию о работе всех подсистем. В-третьих, эксперты в предметной области нуждаются в проверке хода рассуждений системы, чтобы убедиться в том, что экспертные знания применяются правильно [5, 8, 9].

Для изучения объяснений выводов пользователи системы могут задавать вопросы двух типов:

1) Почему система сочла необходимым задать пользователю определенный вопрос?
2) Как система пришла к соответствующему заключению?

Выясним механизм формирования ответов на эти вопросы на примере обратного вывода. Пусть имеются следующие правила, относящиеся к диагностике автомобиля:

Правило 1:
Если топливо поступает = «да» и
состояние двигателя = «работает с перебоями»,
то
причина = «свечи».

Правило 2:
Если состояние двигателя = «не запускается» и
состояние освещения = «не работает»,
то
Причина = «аккумулятор».

Правило 3:
Если состояние двигателя = «не запускается» и
состояние освещения = «работает»,
то
причина = «стартер».

Правило 4:
Если наличие топлива в баке = «да» и
наличие топлива в карбюраторе = «да»,
то
топливо поступает = «да».

Процесс вывода начинается с ввода в рабочую память цели:
причина = *X*,

где *X* — первая переменная, которая может быть сопоставлена с любой строкой. Значение переменной *X*, полученное в ходе вывода, будет указывать причину неисправности двигателя. Если предположить, что правила активизируются в порядке их записи, то на первом шаге, в результате сопоставления с заключением первого правила, *X* получит значение «свечи». Затем система в соответствии с предпосылками правила 1 попытается установить, поступает ли топливо и как работает двигатель. Для решения первой подзадачи будет активизировано правило 4. Так как факты «наличие топлива в баке» и «наличие топлива в карбюраторе» в рабочей памяти не содержатся, и нет правил, с помощью которых система могла бы установить их значения, то пользователю будут заданы вопросы относительно указанных фактов. Кроме этого, после выполнения правила 4 пользователю будет задан вопрос, касающийся состояния двигателя.

Обычно системе известны допустимые значения фактов. Тогда пользователю предлагается выбрать один из допустимых ответов. В рассматриваемом случае возможен следующий диалог пользователя с системой:

Наличие топлива в баке (да/нет)?
да
Наличие топлива в карбюраторе (да/нет)?
да
Состояние двигателя (работает с перебоями, не запускается)?
работает с перебоями

После чего система сообщает причину неисправности двигателя: причина = = «свечи». Если пользователь желает выяснить, почему система задает тот или иной вопрос, то в качестве ответа вводится слово «почему». Например:

Состояние двигателя (работает с перебоями, не запускается)?
почему

В результате будет получен ответ:

1) топливо поступает = «да»,
следовательно, если
2) состояние двигателя = «работает с перебоями»,
то причина = «свечи».

В этом случае система демонстрирует пользователю текущее правило, с которым был связан заданный вопрос.

Чтобы ответить на вопрос «как», в системе сохранится стек всех решений, принятых в течение сеанса работы. Просматривая стек, можно выяснить, достижение каких подцелей привело к данному решению. Например, при вводе вопроса:

Как получено решение «топливо поступает = «да»

будет получен ответ:

Это следует из правила 4;
Если наличие топлива в баке = «да» и
наличие топлива в карбюраторе = «да»,
то
топливо поступает = «да»;
Факт наличия топлива в баке = «да» введен пользователем;
Факт наличия топлива в карбюраторе = «да» введен пользователем.

Таким образом, формирование ответа на вопрос «как» соответствует просмотру дерева целей в направлении от целевого утверждения, фигурирующего в вопросе, до листьев дерева, соответствующих введенным фактам. Из примера видно, что формирование объяснения для случая обратного вывода реализуется с помощью стекового механизма. Сложнее обстоит дело при прямом выводе, т. е. выводе, управляемом данными.

Прямая цепочка рассуждений обеспечивает пользователя менее полезной информацией. Обусловлено это тем, что на промежуточных этапах вывода трудно судить, куда ведет цепочка рассуждений. Так, в ответ на вопрос «почему» пользователю демонстрируется текущее правило. Дальнейшие объяснения не могут быть получены, пока не выполнится следующее правило. Кроме того, сложно формировать полные ответы на вопрос «как», даже после достижения цели. Информация, которая предоставляется, ограничивается списком выполненных правил.

5.7 Достоинства и недостатки продукционных систем

Основные достоинства, благодаря которым продукционные правила получили широкое распространение, заключаются в следующем:

1) Продукционные правила легки для восприятия человека.
2) Отдельные продукционные правила могут быть независимо добавлены в БЗ, исключены или изменены, при этом не требуется перепрограммирование

всей системы. Как следствие этого, представление больших объемов знаний не вызывает затруднений.

3) С помощью продукционных правил выражаются как декларативные, так и процедурные знания.

К недостаткам систем продукций можно отнести следующее:

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

При разработке небольших систем (десятки правил) проявляются в основном положительные стороны систем продукций, однако при увеличении объема знаний более заметными становятся слабые стороны.

Дальнейшее усовершенствование архитектуры, основанной на использовании правил, заключается в применении «крупно-зернистых» правил продукции, которые представляют «источники знаний», работающие под управлением гибкого планировщика. Каждое из правил состоит из образца вызова, элемента, «немедленного кода» и тела. «Немедленный код» исполняется всегда, когда происходит совпадение с образцом выхода, после чего правило переходит под управление планировщика; тело исполняется по команде планировщика.

Контрольные вопросы и задания к главе 5

1) Охарактеризуйте понятие «продукция» с точки зрения их практического смысла.
2) Представьте модульное представление продукционной модели.
3) Приведите упрощенный пример продукционной системы с различной архитектурой.
4) Рассмотрите общий вид продукции.
5) Дайте характеристику ядер продукции.
6) Составьте возможные комбинации связи в ядре продукции.
7) Приведите типовую схему интеллектуальной системы.
8) Перечислите команды продукционной системы, которые могут включаться в правила.
9) Приведите характеристики правил.
10) Почему понятие продукции шире логического следования?
11) Назовите и объясните, из каких элементов формируются антецеденты и консеквенты правил продукционной системы.
12) Расскажите, как представлялись продукции в системе MYCIN.

13) Что такое образ и образец в продукционной системе?
14) Опишите вывод по образцу в продукционной системе.
15) Назовите четыре шага процесса вывода, основанного на поиске по образцу.
16) Рассмотрите пример сортировки строки в пространстве состояний.
17) Назовите и охарактеризуйте две задачи машины вывода (интерпретатора) в продукционной системе.
18) Какие стратегии могут применяться для управления выводом в продукционных системах?
19) Рассмотрите прямой и обратный выводы по образцу в продукционной системе.
20) Приведите примеры прямого и обратного вывода по образцу в продукционной системе.
21) Поясните решение задач в продукционных системах графически в виде графов И, И-ИЛИ.
22) Проведите сравнение прямого и обратного выводов в продукционных системах.
23) Расскажите, как вы понимаете соответствие многих образцов многим объектам в продукционной системе.
24) Назовите возможные источники неэффективности (большие затраты на вычисления) в продукционных системах.
25) Какими особенностями обладает алгоритм соответствия RETE?
26) Опишите алгоритм частичного сопоставления в продукционных системах.
27) Проанализируйте сеть образов и сеть объединения, формируемые алгоритмом RETE.
28) Назовите и охарактеризуйте основные стратегии разрешения конфликтов в образующихся конфликтных множествах при выводе по образцу в продукционных системах.
29) Сформулируйте преимущество представления продукционных систем иерархической структурой.
30) Рассмотрите принцип формирования объяснения выводов в экспертной системе продукционного типа.
31) Выясните механизм формирования ответов на вопросы пользователя на примере обратного вывода в экспертной системе продукционного типа.
32) Приведите достоинства и недостатки продукционных систем.

Литература к главе 5

[1] Андрейчиков А. В. Интеллектуальные информационные системы: учебник / А. В. Андрейчиков, О. Н. Андрейчикова. — М. : Финансы и статистика, 2006. — 424 с.

[2] Достоверный и правдоподобный вывод в интеллектуальных системах Вагин В. Н. [и др.] ; под ред. В. Н. Вагина и Д. А. Поспелова. — М. : Физматлит, 2004. — 704 с.

[3] Бондарев В. Н. Искусственный интеллект: учеб. пособие для вузов / В. Н. Бондарев, Ф. Г. Аде. — Севастополь: Изд-во СевНТУ, 2002. — 615 с.

[4] Искусственный интеллект : в 3 кн. / под ред. Д. А. Попова. — М. : Радио и связь, 1990. — Кн. 1 : Системы общения и экспертные системы. — 461 с.

[5] Искусственный интеллект : в 3 кн. / под ред. Д. А. Поспелова. — М. : Радио и связь, 1990.

[6] Павлов С. Н Интеллектуальные информационные системы : учеб. пособие / С. Н. Павлов. — Томск: Томский межвузовский центр дистанционного образования, 2004. — 328 с.

[7] Уотерман Д. Руководство по экспертным системам / Д. Уотерман. — М. : Мир, 1989. — 390 с.

[8] Представление и использование знаний : пер. с япон. / Х. Уэно [и др.] ; под ред. Х. Уэно, М. Исудзука. — М. : Мир, 1989. — 220 с.

[9] Экспертные системы. Принципы работы и примеры / под ред. Р. Форсайта. — М. : Радио и связь, 1987. — 324 с.

[10] Элти Дж. Экспертные системы: Концепции и примеры: пер. с англ. / Дж. Элти, М. Кумбс. — М. : Финансы и статистика, 1987. — 192 с.

Учебное издание

Павлов Станислав Николаевич

СИСТЕМЫ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА
ЧАСТЬ 1

Учебное пособие

Корректор Осипова Е. А.
Компьютерная верстка Лигай Т. А.

Подписано в печать 26.09.11. Формат 60x84/8.
Усл. печ. л. 20,46. Тираж 200 экз. Заказ

Издано в ООО «Эль Контент»
634029, г. Томск, ул. Кузнецова д. 11 оф. 17
Отпечатано в Томском государственном университете
систем управления и радиоэлектроники.
634050, г. Томск, пр. Ленина, 40
Тел. (3822) 533018.