МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
имени М.В. ЛОМОНОСОВА

Факультет
вычислительной математики
и кибернетики

Ю.С. Корухова

УПРАВЛЕНИЕ ЗНАНИЯМИ

Учебное пособие

МОСКВА

2012

УДК 004.8(075.8)
ББК 32.813я73
К69

Печатается по решению Редакционно-издательского совета факультета вычислительной математики и кибернетики МГУ имени М.В. Ломоносова

Рецензенты:
Н.Н. Попова, доцент;
А.И. Майсурадзе, доцент

Корухова Ю.С.

К69 **Управление знаниями:** Учебное пособие. – М.: Издательский отдел факультета ВМиК МГУ имени М.В. Ломоносова (лицензия ИД N 05899 от 24.09.2001 г.); МАКС Пресс, 2012. – 48 с.
ISBN 978-5-89407-485-6
ISBN 978-5-317-04157-1

В пособии рассматриваются вопросы, связанные с управлением знаниями в организациях, и технологии, используемые на разных этапах: для представления знаний, хранения информации, построения выводов, использования прецедентов, анализа данных.

Пособие предназначено для слушателей спецкурса «Управление знаниями», читаемого автором на факультете ВМК МГУ для студентов 3–5 курсов.

УДК 004.8(075.8)
ББК 32.813я73

Учебное издание

КОРУХОВА Юлия Станиславовна

УПРАВЛЕНИЕ ЗНАНИЯМИ

Учебное пособие

Издательский отдел
Факультета вычислительной математики и кибернетики МГУ имени М.В. Ломоносова
Лицензия ИД N 05899 от 24.09.01 г.

119992, ГСП-2, Москва, Ленинские горы, МГУ имени М.В. Ломоносова, 2-й учебный корпус

Напечатано с готового оригинал-макета
в издательстве ООО "МАКС Пресс". Лицензия ИД N 00510 от 01.12.99 г.

Подписано в печать 24.07.2012 г.
Формат 60x90 1/16. Усл.печ.л. 3,0. Тираж 100 экз. Заказ 280.

119992, ГСП-2, Москва, Ленинские горы, МГУ им. М.В. Ломоносова, 2-й учебный корпус, 527 к.
Тел. 939-3890, 939-3891. Тел./Факс 939-3891

ISBN 978-5-89407-485-6
ISBN 978-5-317-04157-1

Посвящается светлой памяти моего Учителя
Владимира Николаевича Пильщикова

Введение

В современном мире одним из самых важных ресурсов, обеспечивающих национальную экономическую и военную силу, являются знания. Несмотря даже на неформализованность многих видов знаний и возникающих из-за этого сложностей при передаче и автоматизированной обработке, в различных научных областях создаются методы для решения таких задач. Оценивая рыночную стоимость той или иной организации, в расчет берется не только ее финансовая прибыль, но и информационные базы данной компании, знания ее работников, ее структурная организация. Важной составляющей организации является ее интеллектуальный капитал, который складывается из так называемого "человеческого" капитала и структурного.

В первую составляющую входят знания, хранимые работниками компании и теми лицами, с кем она взаимодействует. Каждый сотрудник организации обладает знаниями, компетенциями, отношением к окружающим его вещам. Он имеет определенные отношения с коллегами внутри организации, включая партнеров и клиентов организации, с другими субъектами бизнес-среды. Работая в организации, сотрудники используют свои знания для решения бизнес-задач организации, формируя тем самым ее интеллектуальный капитал. Таким образом, знания сотрудников и результаты их применения для решения задач организации являются важной составляющей интеллектуального капитала организации, формирующего ее рыночную стоимость.

Структурный капитал включает в себя все знания, которые остаются, когда работники организации уходят домой: информационные базы, программное обеспечение, различную литературу, связанную с деятельностью компании.

Управление знаниями есть выполнение действий над интеллектуальным капиталом организации. Также понятие *управление знаниями* может быть определено как действия и технологии, необходимые для выгодного использования ресурса знаний.

На Западе дисциплина управления знаниями (Knowledge Management) появилась около двадцати лет назад. Если ранее для получения превосходства над конкурентами достаточно было обладать некоторыми уникальными знаниями, то сейчас во многом уровень владения информацией у фирм-конкурентов выровнялся, и эффективность определяется часто не наличием тех или иных знаний, а умением ими "управлять". Первыми, кто профессионально стал управлять знаниями в рамках своих предприятий, были консалтинговые и юридические компании, информационные бюро. Специфика их бизнеса такова, что им требуется сохранять в некотором виде свой предыдущий опыт, тиражировать его для передачи в разные подразделения компании, в филиалы. Сегодня невозможно представить по-настоящему эффективную

организацию, не занимающуюся в той или иной мере управлением знаниями. Успешный опыт в этой области демонстрируют совершенно разные компании: Microsoft, Газпром, РАО ЕЭС, British Petroleum, Северсталь, McKinsey & Company и многие другие. Один из авторов книг, анализирующих и описывающих стратегии управления компаниями Арие де Геус, отвечавший в компании Shell за стратегическое планирование, сделал вывод, что способность познавать быстрее своих конкурентов – устойчивое конкурентное преимущество компании[1].

Живой интерес компаний к дисциплине управления знаниями обусловлен рядом причин. Основными можно считать следующие:

- сложность предметных областей возрастает, увеличивается объем данных в информационных системах, причем значительное количество информации хранится в неструктурированном виде;
- динамика развития бизнеса является высокой: быстро создаются новые компании, продукты, услуги, меняется структура рынка, что требует быстрой реакции на происходящие события;
- для больших проектов, как правило, требуются знания далеко не одного человека, и становится важен не только опыт конкретного сотрудника, но и умение им воспользоваться при взаимодействии с другими специалистами.

Курс "Управление знаниями" входит в программы разных университетских специальностей. В рамках настоящего пособия рассмотрение предмета будет проведено с точки зрения специалиста по информационным технологиям, первостепенное внимание будет уделено компьютерным технологиям, которые используются для организации хранения, приобретения, обработки знаний.

1. Знания и их классификация

В процессе управления знаниями мы сталкиваемся с тремя близкими, но все же разными по смыслу терминами: данные, информация и знания.

В рамках курса под *данными* мы будем понимать произвольные символьные последовательности. Такие последовательности являются записью некоторых фактов, наблюдений, отражают наше восприятие каких-либо явлений... Данные можно легко переписать, организовать их хранение, передавать по компьютерным сетям. В качестве примера данных можно рассмотреть последовательность из 14 символов:

```
+7(495)9391000
```

Информация - это подмножество данных, которое включает в себя только те данные, для которых известен контекст их использования, цель их хранения и определено соответствие этой цели. Добавив к приведенному выше примеру такие характеристики, мы получим из данных информацию:

```
+7(495)9391000  - это номер телефона справочной МГУ
```

Информация включает только наборы символов, имеющие смысл для какой-либо области.

Знания — это информация, которую можно использовать при совершении действий, в том числе для получения новой информации и новых знаний. Так информацию о телефоне тоже можно сделать знанием:

```
Если позвонить по номеру справочной +7(495)9391000, можно
получить информацию о МГУ.
```

1 Arie de Geus http://www.ariedegeus.com/

Рассмотрим пример из другой предметной области. Каждый день во время работы кафе печатается значительное количество чеков, на каждом из которых содержится последовательность символов. Содержание чеков можно рассматривать как данные. Если выбрать из них те, которые содержат оплату бутербродов, то сведения о количестве проданных бутербродов в день по всем последнего месяца будет являться ценной информацией для менеджера кафе. Особенно, если данный менеджер обладает специфическими для его области знаниями, как, например: "Если продажи бутербродов падают, нужно улучшить их дизайн и добавить растительные компоненты..." Знания используются для оценки и управления ситуацией. Например, чтобы решить, как нам поступить или как применить имеющуюся информацию.

В литературе [3, 4] приводится классификация знаний по различным признакам.

Знания бывают *декларативными или процедурными*. Декларативные знания — это знания "о чем-то", о некотором факте. Приведем пример: "При прочих равных условиях увеличение цены продукта приведет к снижению уровня продаж". Процедурные знания — это знания об осуществлении некоторых действий. Таковым является, например, знание о том, как ездят на велосипеде.

Знания делятся на *явные и неявные*. К первой категории относятся знания, которые можно выразить в символьном виде (с помощью слов, цифр...). К такому виду можно отнести, например, знание которым обладает директор магазина, который знает всех клиентов в округе магазина. Неявными называются знания, представление которых формализовать не удается: это подозрения, интуиция...

Знания бывают *общими и специализированными.* К общим знаниям относятся те, которыми обладает значительное число людей, в отличие от специализированных знаний, владение которыми является привилегией только специалистов в конкретной области. Рассмотрим в качестве примера область – медицину. Знание о том, что средствами обработки ран являются зеленка и йод, можно назвать общим, тогда как знаниями о лечении конкретного заболевания, например, кори, владеет только врач – специалист.

По степени использования информации о предметной области знания можно классифицировать на *поверхностные и глубокие*. Первые применяются для решения несложных задач и не требуют детального понимания всей предметной области. Так, например, при принятии решения о выдаче кредита на небольшую сумму банковскому работнику может быть достаточно убедиться в наличии стабильной зарплаты и недвижимости, то есть решение будет приниматься на основе всего нескольких правил, установленных банком. Если же речь идет о выдаче более значительной суммы организации, то служащему банка потребуется оценить порядка 14 различных параметров и использовать глубокие знания предметной области (условий кредитования).

Также знания можно разделить на *рассуждения и эвристические знания*. Рассуждения могут проводиться разными способами:

- по аналогии

Например, зная, что батарейка аналогична аккумулятору (так как оба используются для хранения энергии и как источники питания электронных устройств), можно понять такой факт — раз аккумулятор теряет заряд со временем, то же произойдет и с батарейкой.

- формальными методами (дедуктивным или индуктивным)

Дедуктивный метод позволяет строить новые факты на основе известных с помощью заданных правил вывода. Известный литературный герой А. Конан-Дойля

Шерлок Холмс применял данный метод рассуждений для расследования преступлений: на основе известных фактов он строил выводы, которые далее использовал для построения следующих фактов цепочки рассуждений.

Индуктивный метод в качестве исходных данных имеет несколько фактов, на основе которых строится некоторое общее правило, выявляется закономерность. Например, на основе фактов, что человек "любит рыбалку, охоту и занимается альпинизмом" можно сделать вывод, что он любит заниматься спортом на улице. Однако, построенное правило может потребовать уточнения при появлении новых фактов — метод не гарантирует, что правило останется истинным.

- на основе прецедентов

В данном виде рассуждений используется накопленный ранее опыт, и для решения задачи, производится поиск похожей на нее, уже решенной ранее. В данном случае используется опыт не только книг и учебников, но и описание случаев из практики того или иного специалиста.

Эвристические методы позволяют получить решение в тех случаях, когда другие методы рассуждений оказываются неприменимым, например, из-за огромного количества вариантов вывода. В качестве обоснования их применения зачастую можно привести только успешный опыт предыдущего использования конкретного правила построения выводов (эвристики). По сравнению с остальными методами рассуждений они часто позволяют быстрее получить результат, однако эвристики не гарантируют его получения.

В любой организации знания содержатся в нескольких источниках. Первым, конечно, следует назвать сотрудников – "владельцев" знаний. Работая в определенной области, сотрудники обладают о ней специфическими знаниями, но кроме этого важными являются также знания о коллегах по работе. Так, например, знание сильных и слабых сторон работников коллектива разработчиков поможет организатору проекта правильно распределить между ними конкретные задания и добиться оптимального выполнения работы. Другим источником знаний является их письменное, электронное и другие представления, а также технологии и системы, которые позволяют с таким представлением работать. И третье местоположение знаний, которое необходимо упомянуть, содержится в самой структуре организации, в правилах взаимодействия ее подразделений, сотрудников. Устроившись на работу, новый сотрудник помимо знаний о предметной области и местоположении дополнительной информации, как правило узнает, с кем конкретно и как он взаимодействует в процессе работы, узнает о правилах, традициях и культуре организации, которые также являются составными компонентами ее знаний.

В процессе управления знаниями можно выделить несколько этапов, которые циклически повторяются. Начальным является этап сбора знаний, поиск которых происходит в самых различных источниках, включая аудио- и видеофайлы, электронные письма. Далее полученная из всех источников информация должна быть организована, чтобы по ней можно было автоматически находить требуемые факты и на их основе генерировать новые знания. На этом этапе важными техническими характеристиками являются скорость, точность и полнота поиска наряду с дружественностью интерфейса системы управления знаниями. После организации информации, как правило проходит этап ее усовершенствования и уточнения. Используя технологии и инструменты интеллектуального анализа данных (data mining), на основе собранной информации могут быть получены новые знания. Однако накопление, организация и усовершенствование собранной информации не

даст никакого результата, если она не будет доступна сотрудникам. Поэтому завершающим этапом цикла управления знаниями является этап распространения знаний, для того, чтобы они могли эффективно использоваться сотрудниками организации, ее различных подразделений.

В современных организациях на каждом из этапов возможно использование технологий и систем управления знаниями, рассмотрение которых будет проведено в следующих главах.

Задания к главе 1

1. Вспомните несколько различных ситуаций, в которых Вам сегодня приходилось принимать решения. Решения могут быть самыми различными, например, "на перекрестке я повернул направо", "в следующем году я еду в экспедицию на Южный полюс" и т. д. Поясните, какие данные, информация и знания использовались Вами при принятии каждого из решений.
2. На примере какой-либо организации назовите места нахождения знаний в ней. Проведите классификацию названных Вами знаний.

2. Системы управления знаниями. Искусственный интеллект как технология управления знаниями

По классам решаемых задач системы управления знаниями можно разделить на следующие основные группы:

- системы организации и хранения знаний,
- системы распространения знаний,
- системы использования знаний,
- системы, помогающие создавать новые знания.

При решении таких задач разработчикам систем, как правило, требуются технологии организации баз данных, хранилищ данных, а также технологии позволяющие автоматически проводить с символьным представлением информации действия, традиционно выполняемые человеком. Автоматизацией таких задач занимается область называемая "искусственный интеллект".

В настоящее время компьютеры используются не только как вычислительные устройства, но и производят работу с символьным представлением информации из самых различных предметных областей. Поручая компьютеру решение подобных задач, необходимо понимать, что по существу он все равно остается исполнителем заданных ему алгоритмов, выполняя ограниченное множество команд. В условии таких ограничений требуется "научить" компьютер оперировать символьной информацией (принимать решения, делать выводы) подобно человеку. Поэтому в управлении знаниями используются технологии, методы, разработанные в области искусственного интеллекта.

Датой рождения искусственного интеллекта (ИИ) традиционно считается 1956 год, когда Джон МакКарти организовал в Дартмутском колледже семинар, собравший исследователей – пионеров в области ИИ: Марвина Минского, Алана Ньюэлла, Герберта Саймона. Продемонстрированная на семинаре программа "Логик-теоретик" показала возможность автоматического доказательства логических формул, представленных в виде символьной записи. Таким образом, первые программы

выполняющие автоматизированную обработку знаний, появились уже более полувека назад.

В 1963 году была представлена система А. Ньюэлла и Г. Саймона "Общий решатель задач" (GPS - General Problem Solver), в которой была сделана попытка промоделировать процесс мышления в виде некоторого алгоритма. Решатель задач представлялся именно "общим": при задании информации о любой предметной области, применялась общая схема решения задачи. Поиск решения проводился в заданном пространстве состояний предметной области. Среди состояний выделялось исходное и целевое (достижение которого соответствовало нахождению решения поставленной задачи). Преобразования одного состояния в другое проводилось с помощью операторов. Кроме описания самих операторов, требовалось задать их приоритет относительно друг друга. Практика использования системы показала, что для решения даже довольно простых задач требуется задавать системе огромное количество информации, связанной с описанием предметной области.

В другом направлении исследований — разработке экспертных систем, наоборот, было сделано ограничение на область работы системы. Целью ставилась разработка систем, использующих узкоспециализированные (экспертные) знания, но способных решать в заданной ограниченной области достаточно сложные задачи на уровне человека – эксперта. Такие системы появились в 70-х годах прошлого века и получили название систем, основанных на знаниях или экспертных систем[2]. Их особенность состоит в том, что они осуществляют не поиск решения задачи среди известных, а выводят решение, применяя заданные в системе знания (правила). Таким образом, экспертные системы являются примерами систем управления знаниями. Одним из первых примеров реализации такого подхода была программа DENDRAL [8], разработанная в Стэнфордском университете. Система выводила молекулярную структуру химического вещества на основе информации, полученной от масс-спектрометра. В медицинской области известной была система MYCIN [8], которая диагностировала бактериальные инфекции и давала рекомендации по их лечению. Геологическая система PROSPECTOR [8] действовала как консультант-помощник при поиске залежей руд: имея данные о геологии района, она оценивала вероятность обнаружения в нем определенных видов минеральных отложений. Система была доведена до промышленной эксплуатации. Создавались системы, основанные на знаниях, и для других областей: математики, военного дела, электроники, сельского хозяйства, юриспруденции [20]. При разработке подобных программ существенным является выбор подходящего представления знаний, который позволяет хранить знания, использовать их для поиска ответов на вопросы и вывода новых знаний.

3. Представление знаний. Логический подход

Для автоматизированной обработки информация о предметной области должна быть представлена для компьютера в символьном виде. Представление должно удовлетворять набору требований, чтобы обработка информации могла проводиться автоматизированно: должен быть задан синтаксис, семантика представления и описан механизм получения новой информации и знаний из имеющихся. Первым рассмотрим

2 В некоторой литературе экспертными системами называется только подмножество систем, основанных на знаниях, в котором используются глубокие экспертные знания. В подмножество не включаются системы, содержащие общие знания.

логический подход к представлению знаний. Основная идея подхода состоит в том, что вся информация в предметной области, включая задачи, которые требуется решить, представляется в виде набора формул в некоторой логике. Знания отображаются как совокупность формул, а получение новых знаний происходит в результате выполнения логического вывода. В данной главе в качестве языка представления знаний будет рассмотрена логика предикатов первого порядка.

Синтаксис языка.

Формально, предложения логики предикатов первого порядка строятся из следующих элементов:

- переменные,
- константы,
- знаки пунктуации (`( ) , .`),
- логические связки (`not and or` →),
- особый вид связок для переменных – кванторы (∀ ∃)
- функциональные символы,
- предикатные символы[3].

Константы, функциональные символы, предикатные символы и их смысл определяются предметной областью.

Термом называется переменная, константа, а также вызов функции от одного или нескольких термов. Логическим выражением называется предикат от параметров-термов, либо одна из конструкций вида `not` E, E_1 `and` E_2, E_1 `or` E_2, где E, E_1, E_2 – логические выражения.

Рассмотрим представление знаний на примере конкретной предметной области — железнодорожные перевозки. Константами в данной области будут станции (`Moscow, Zvenigorod, Dublin, Belfast, Cork` и другие). Чтобы отразить факт, что данные объекты являются железнодорожными станциями, введем предикат с одним параметром `station(X)`, который является истинным, если в качестве аргумента ему задан населенный пункт, в котором есть станция, и ложным во всех остальных случаях. Упомянутые выше города станции имеют, что можно описать в виде предложения логики предикатов:

```
station(Moscow) and station(Zvenigorod) and station(Dublin)
and station(Belfast) and station(Cork)
```

В предложении использована связка `and` так как утверждается истинность всех пяти фактов. Поскольку ситуация, когда задан набор аксиом, описывающих некоторую область встречается достаточно часто, вместо `and` для краткости используется запятая (,).

Для железнодорожных перевозок необходимо знать, между какими станциями есть сообщение, что будет выражено с помощью предиката `path` с двумя аргументами. Будем считать `path(X,Y)` истинным, если есть железнодорожная ветка между `X` и `Y`. Добавим к описанию предметной области аксиомы, использующие этот предикат:

```
path(Moscow, Zvenigorod),
path(Dublin, Belfast),
path(Cork, Dublin).
```

Введенный предикат `path` обладает некоторыми очевидными для нас, но не представленными в системе свойствами. Во-первых, по железной дороге можно ехать

3 Язык может быть расширен за счет введения дополнительных логических операций, например, эквивалентности (≡)

в обоих направлениях, то есть наличие пути из X в Y означает и наличие пути из Y в X. Поскольку подобная информация не связана с конкретными объектами, для ее выражения понадобятся переменные и кванторы:

$$\forall X \forall Y \; path(X,Y) \rightarrow path(Y,X) \qquad (3.1)$$

("для любых объектов X и Y верно следующее: если есть путь между X и Y, то есть и путь между Y и X")

Во-вторых, если есть путь из X в Z, и есть путь из Z в Y, значит можно проехать из X в Y. Данное утверждение верно для любых X,Y, поэтому они будут связаны квантором всеобщности. Для каждой пары X, Y достаточно будет найти хотя бы одно Z, в который из них есть путь. Поэтому переменная Z будет связана квантором существования:

$$\forall X \forall Y \; (\exists Z \; path(X,Z) \text{ and } path(Z,Y)) \rightarrow path(X,Y) \qquad (3.2)$$

На примере этих аксиом видно, что в выражениях не присутствуют свободные (не связанные кванторами) переменные. Это не случайно: для представления знаний используются логические предложения — выражения, не содержащие свободных переменных.

Итак, исходная информация о предметной области представлена в виде множества аксиом. Посмотрим, как на основе представленных в логическом виде знаний получать новые знания. Этот процесс называется логическим выводом и одним из методов вывода для логики первого порядка является метод резолюций. Метод резолюций применяется для решения задач следующего вида. Имеется конечное множество аксиом $A_1, A_2, \ldots, A_n$. Требуется показать, что некоторое логическое предложение G логически следует из этого множества, то есть является истинной формула: A_1 and A_2 and... and $A_n \rightarrow G$.

Общая идея метода резолюций. Применим метод доказательства от противного. Предположим, что G – ложно. Значит выражение not G является истинным. Исходное множество аксиом было истинным, значит и выражение

A_1 and A_2 and... and A_n and not G

также должно оставаться истинным (то есть множество аксиом должно быть непротиворечивым). Построив противоречие для этого утверждения, мы покажем несостоятельность исходной гипотезы, значит G – истинно.

Рассмотрим применение метода на примере из области железнодорожных перевозок.

Шаг 1. Приведем аксиомы к конъюнктивной форме, используя логические тождества[4]. Тогда все множество аксиом будет представлено как набор предикатов, отрицаний предикатов и дизъюнкций. Все переменные подразумеваются связанными кванторами всеобщности

```
1. station(Moscow),
2. station(Zvenigorod),
3. station(Dublin),
4. station(Belfast),
5. station(Cork),
6. path(Moscow, Zvenigorod),
7. path(Dublin, Belfast),
8. path(Cork,Dublin),
9. not path(X,Y) or path (Y,X),
10.    not path(X,Z) or not path(Z,Y) or path(X,Y),
```

4 Полный перечень применяемых преобразований и обоснование метода можно найти в книге Ч. Чень, Р. Ли "Математическая логика и автоматическое доказательство теорем"

последние две аксиомы получены из (3.1) и (3.2) с использованием преобразований

A → B **в** not A or B

not (A and B) **в** not A or not B.

Шаг 2. Применим метод резолюций, чтобы показать, что есть железнодорожный путь из Cork в Belfast: path(Cork, Belfast). Для этого добавим к множеству аксиом отрицание доказываемого факта:

11. not path(Cork, Belfast)

и покажем противоречивость множества, содержащего аксиомы 1 – 10 и предложение 11

Шаг 3. Для доказательства строим новые предложения — резольвенты.

Выбираем пару утверждений содержащих предикат P и его отрицание not P (они называются контрарными литерами):

A_1 or ... or **P** or...or A_n

B_1 or ... or **not P** or...or B_n

Их резольвентой назовем предложение

A_1 or ... or A_n or B_1 or ... or B_n,

содержащее дизьюнкцию всех исходных литералов, кроме P и not P

Резольвента добавляется к исходному множеству предложений (если она ранее не присутствовала в нем). Согласно правилам записи, выражения и контрарные литеры могут содержать внутри себя переменные. В этом случае применяется унификация: выполняется поиск таких значений переменных, при которых в предложениях появятся одинаковые подвыражения P, причем одно из них с отрицанием. Значения переменных подбираются таким образом, чтобы все вхождения одной и той же переменной были заменены одним и тем же термом, разные переменные могут быть заменены как на разные термы, так и на одинаковые. Этот процесс называется унификацией.

Вернемся к примеру, построим резольвенту для аксиомы 10 и предложения 11.

10. not path(X,Z) or not path(Z,Y) or path(X,Y).

11. not path(Cork, Belfast)

их подвыражения path(X,Y) и path(Cork, Belfast) станут одинаковыми, если переменная Y примет значение Belfast, а переменная X – значение Cork. Тогда в выражениях появятся контрарные литеры и можно построить резольвенту:

12. not path(Cork,Z) or not path(Z,Belfast)

Далее, построим резольвенту для предложения 12 и аксиомы 7, приняв значение переменной Z равным Dublin, получим

13. not path(Cork,Dublin)

Построим резольвенту для предложения 13 и аксиомы 8. Унификации на данном шаге не требуется, и, как видно, логические утверждения противоречат друг другу. Их резольвентой будет пустой дизъюнкт, который и говорит о противоречивости множества.

Доказательство на этом завершается, в результате его было показано, что path(Cork,Belfast) является истинным утверждением, логически следует из заданного набора аксиом.

Как видно, метод резолюций является именно методом, а не алгоритмом: на каждом шаге мы сталкиваемся с проблемой выбора подходящих пар выражений. Поэтому практические реализации метода сталкиваются с проблемой перебора

вариантов применения правил. Однако для особого вида дизъюнктов, содержащих один литерал без логического отрицания, такая процедура реализована в языке логического программирования Пролог [9].

Программа на языке Пролог состоит из набора предложений, являющихся фактами и правилами. Факты в языке Пролог описываются логическими предикатами с конкретными значениями. Например:

```
path(dublin,belfast).
```
[5]

Правила в Прологе записываются в форме правил логического вывода с логическими заключениями (они пишутся слева) и списком логических условий (указываются справа, через запятую). Например, правило

```
path(X,Y):-path(X,Z),path(Z,Y).
```

соответствует логическому предложению

```
(path(X,Z) and path(Z,Y)) → path(X,Y).
```

Для начала работы программы на Прологе необходимо задать ей вопрос (проверить истинность некоторого факта или набора фактов), на который интерпретатор Пролога формулирует ответ. Для нахождения ответа используется метод резолюций, на котором основан механизм вывода языка.

Например, задав программе вопрос

```
?-path(dublin,belfast).
```

Будет получен ответ `true`, так как данный факт известен программе. Если же в качестве параметра предиката указать переменную, то при поиске ответа будет задействован механизм унификации, и будут предъявлены те значения переменных, при которых выражение является истинным. Например, на вопрос

```
?-path(X,belfast).
```

будет получен ответ, что данный факт можно доказать при определенном значении `X`:

```
X=dublin
true.
```

При решении задачи интерпретатор Пролога использует механизм поиска с возвратом (backtracking).

Логический подход к представлению знаний имеет свои достоинства. Во-первых, в качестве «фундамента» здесь используется классический аппарат математической логики, методы которой хорошо изучены и формально обоснованы. Во-вторых, существуют достаточно эффективные процедуры вывода (в том числе реализованные в языке логического программирования Пролог). Также знания, хранящиеся в системе, легко дополнять и организовывать проверку непротиворечивости. Однако, очевидны и недостатки такого представления: оно является достаточно удобным для автоматизированной обработки, но не всегда очевидным и естественным для человека, и перевод на логический язык представляет собой сложную задачу. Несмотря на модульность базы знаний, она не является структурированной, то есть связи между модулями если и известны разработчику базы знаний, но не отражены в явном виде при записи знаний. Мощный механизм логического вывода также имеет свои особенности: вывод является монотонным, то есть один раз доказанное утверждение уже нельзя "отменить", оно будет оставаться истинным, что не всегда удобно. Например, из факта "у меня есть свободное время" может следовать "я могу поехать на итальянский курорт", однако появление дополнительной информации "у меня нет средств" отменяет сделанный ранее вывод. Также логический подход имеет

5 Примеры приведены для swi-prolog, который доступен на сайте www.swi-prolog.org. Согласно его синтаксису, имена переменных должны начинаться с заглавной буквы или символа _, константы – со строчной буквы.

свои особенности в работе с негативной информацией, так как в логике принимается гипотеза о замкнутости мира: все, что не истинно, является ложным. В реальности же существуют факты ни истинность, ни ложность которых не доказаны. Еще одну проблему вызывают противоречия, которые не представимы в логической системе, однако они присутствуют в знаниях людей. Например, очевидное знание "все птицы умеют летать" сложно представить одновременно с фактами, что "пингвин — птица" и "пингвины не летают". Поэтому кроме логического подхода были разработаны и альтернативные способы представления знаний.

Задания к главе 3

3.1. Представьте высказывание «Все кошки говорят на одном языке» с помощью логики предикатов первого порядка. Используйте предикат `speaks(X,L)`, который принимает истинные значения тогда и только тогда, когда X говорит на языке L.

3.2. Какие аксиомы и правила необходимы, чтобы логически вывести факт `grandson(Tom,Anna)`, если даны факты `mother(Tom,Patricia)` и `mother(Patricia, Anna)`?

3.3. Определив необходимые константы, функциональные и предикатные символы и задав их интерпретацию, представить на языке логики предикатов следующую информацию:

a) Некоторые школьники сдавали экзамен по геометрии весной 2012 года.

b) Не каждый студент, поступивший на первый курс университета получает диплом.

c) Только один студент участвовал в экзамене по китайскому языку весной 2011 года.

d) Выпускник Лингвистического университета знает как минимум два языка.

e) Нет дыма без огня.

f) Слышит звон, да не знает, где он.

3.4. В городе N живут 5 подозрительных личностей: Джек, Том, Сэм, Энди и Лиз. Прошлой ночью в этом городе был ограблен самый большой банк. Полиции удалось собрать некоторую информацию о грабителе (грабителях). Помогите полиции определить имя (имена) преступника, представив известную информацию в виде фактов и правил языка Пролог и задав соответствующий вопрос программе.
Грабитель был мужчина, который приехал на грузовике, одетый в темную одежду.
Факты, также известные полиции: Том в манере одеваться копирует Энди. Сэм и Джек — заядлые курильщики. Джек ездит на Volvo, а Энди на грузовике. Лиз гоняет по городу на велосипеде. Сэм перемещается на том же транспорте, что и Энди. Лиз любит красные костюмы и куртки. Энди как правило одевается в зеленое. Все курящие жители города N, носят темную одежду.

4. Продукционные системы

Часто в жизни мы сталкиваемся с ситуациями, когда знания специалистов представлены в виде правил

`ЕСЛИ <что-то произошло>, ТО <выполнить действия>.`

Подобные тексты часто встречаются, например, в инструкциях к различным приборам и устройствам. Поскольку именно таким способом удобно представлять

многие специализированные знания, рассмотрим данный подход более подробно.

4.1 Продукции. Прямой и обратный вывод

Продукцией называется правило вида `Если <условия>, то <действия>`, смысл которого заключается в следующем: при выполнении всех заданных в правиле условий, нужно выполнить записанные в нем действия. С подобными правилами мы часто сталкиваемся не только в специализированных областях, но и в повседневной жизни. Например, правила

`"Если Вы сели в троллейбус, то нужно заплатить за проезд",`
`"Если вода кипит, то часть воды превращается в пар"`

являются продукциями.

Можно рассматривать продукционные правила по аналогии с логическими формулами следствия: `предпосылка → заключение`. Левая часть такой формулы является декларативной, содержит некоторые утверждения, правая — процедурная: в ней записаны действия, в том числе инициирующие выполнение других правил продукции.

Задача, которую решают с помощью продукций, — это поиск цепочки правил, преобразующих исходный набор фактов в целевой. Преобразования обычно выполняются с помощью прямого или обратного вывода.

При *прямом выводе* принцип рассуждений может быть сформулирован как "выводим из известных фактов все возможные, пока не получим того, что требуется". Вывод называется прямым, потому что для получения новых фактов мы двигаемся по направлению отношения логического следствия (→) в записи продукционных правил. Стартовой точкой для рассуждений является набор фактов, описывающих задачу (см. рисунок 4.1). Если среди записанных фактов есть такие, которые содержатся в левой части какого-либо продукционного правила, то данное правило может быть применено. Применяется одно из подходящих правил, и в результате получается измененный набор фактов (см. результат применения правила на рисунке 4.1). Далее процесс применения правил продолжается аналогичным образом, пока среди набора фактов не будет получено то, что требовалось вывести, либо пока ни одно из правил не окажется неприменимым. Результат решения задачи зависит от последовательности применения правил: останов в ситуации "ни одно правило неприменимо" не означает, что решение найти невозможно.

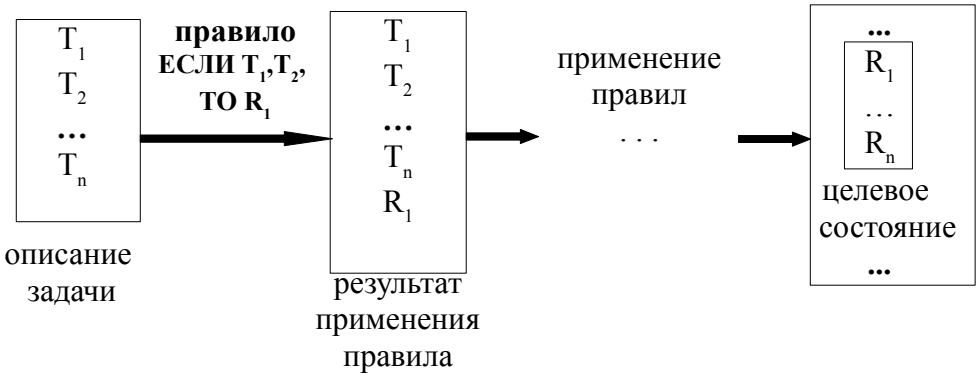


Рис.4.1 Стратегия прямого вывода

При применении правил мы сталкиваемся с известной в искусственном интеллекте *проблемой рамок и границ (frame problem)*: требуется разграничить те факты, которые остаются после применения правила истинными, от тех, которых

стали ложными и должны быть удалены из дальнейшего рассмотрения. Например, если объект A находился в точке X, то есть был истинным факт at(A,X), а в результате применения правила он переместился в точку Y, то к известным фактам нужно добавить at(A,Y) и удалить at(A,X), который стал ложным.

Другой стратегией применения правил является стратегия *обратного вывода.* В данном случае мы двигаемся от цели, и смотрим, что нужно доказать, чтобы цель была выполнена. Для этого нужно найти правила, содержащие цель в правой части. Тогда их левая часть (условие) добавляется к списку целей, которые нужно вывести. Часть (или даже все) условия могут сразу оказаться доказанными, если они были заданы как исходные данные. Каждое из недоказанных условий становится новой целью и процесс вывода продолжается.

Рассмотрим применение стратегий прямого и обратного вывода на примере. Модельная область знаний — классификация животных. Известные факты: "животное желтого цвета", "имеет длинную шею". Требуется вывести факт, что это жираф.
Заданные правила (для краткости примера ограничимся набором из 4 правил):

1. Если животное желтого цвета, то добавить к известным факт "любит солнце".
2. Если животное желтого цвета, то добавить факт "животное живет в саванне".
3. Если животное белое и живет на северном полюсе, то добавить факт "белый медведь".
4. Если животное живет в саванне и имеет длинную шею, то добавить факт "жираф".

Решение 1.
Применим стратегию прямого вывода. К исходному множеству фактов

"животное желтого цвета"
"имеет длинную шею"

применимо правило 1, либо правило 2, поскольку все условия их левых частей выполнены. Поскольку подходящими являются оба правила, выбор между ними должен быть сделан на основе дополнительной информации. В рассматриваемом примере мы будем использовать правило с меньшим номером. Применив правило 1, получим новое множество фактов:

"животное желтого цвета"
"имеет длинную шею"
"любит солнце"

Поскольку в данном множестве не содержится цели ("жираф"), процесс применения правил продолжается. Применив правило 2, получим новое множество:

"животное желтого цвета"
"имеет длинную шею"
"любит солнце"
"животное живет в саванне"

Теперь применимым становится правило 3, которое позволит добавить в множество фактов "жираф". Таким образом, цель достигнута.
Решение 2.
Применим для решения задачи стратегию обратного вывода. Добавим в множество целей факт, который требуется вывести по условию задачи:

"жираф"

Данный факт встречается в правой части только одного правила — четвертого. Чтобы его вывести, необходимо доказать факты "животное живет в саванне" и "имеет длинную шею", указанные в левой части правила. Они становятся очередными

целями, которые требуется вывести. Множество целей имеет вид:

"животное живет в саванне"

"имеет длинную шею"

Второй факт есть в исходных данных задачи, значит вторая цель выведена. Первый факт присутствует в правой части второго правила и может быть заменен на факт из левой части данного правила: "животное желтого цвета", который и становится новой целью. Однако этот факт содержится в исходных данных задачи, то есть его вывод завершен. Процесс вывода на этом останавливается, так как все цели были успешно доказаны. Значит, на основе имеющихся исходных данных и правил вывода доказан факт "жираф".

Иногда на практике применяется смешанная стратегия вывода, при которой строятся как новые факты, выводимые из исходных (движение в прямом направлении), так и промежуточные цели, которые требуется доказать (движение от исходной цели, обратный вывод). Когда все цели будут содержаться в множестве известных фактов процесс вывода считается успешно завершенным.

На практике процесс вывода не всегда завершается, так как могут присутствовать продукционные правила, применимые бесконечно. Приведем пример: пусть в предметной области есть два объекта — куст и слон. С течением времени куст растет. Слон ест куст, при этом куст уменьшается, а слон растет. Обозначив в скобках размер соответствующего объекта, запишем продукционные правила:

Если "слон(x)" и "куст(y)", то добавить факты "слон(x+1)" и "куст(y-1)"

Если "куст(y-1)", то добавить факт "куст(y)"

Применение данных правил может производиться бесконечно.

Правила, заданные в форме продукций, используются для получения выводов в продукционных системах.

4.2 Продукционные системы

Продукционная система — это вычислительный формализм, в котором используются правила–продукции. Основными компонентами продукционной системы являются (рисунок 4.2):

- база данных (рабочая память)

В рабочей памяти хранится набор фактов, описывающих решаемую задачу, и тех, которые были получены в результате вывода.

- база знаний (база правил)

В базе знаний хранится набор продукций, выражающих знания, относящиеся к предметной области.

- управляющая стратегия (механизм вывода)

Управляющая стратегия определяет функционирование системы. Она является универсальной и не зависит от базы данных и базы правил, с которыми работает.

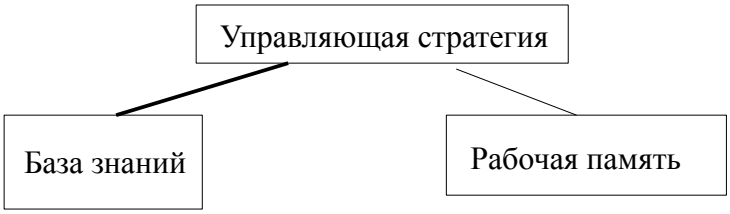


Рис. 4.2 Архитектура продукционной системы

Работа продукционной системы заключается в циклическом выполнении следующих шагов:

Шаг 1. Нахождение применимых правил. Напомним, что к применимым относятся правила, условиям которых удовлетворяют хранящиеся в рабочей памяти факты.

Шаг 2. Разрешение конфликтов. На первом шаге может быть найдено несколько подходящих правил, из них требуется выбрать одно. Выбор может быть сделан на основе различных факторов, в том числе:

- правилам могут быть заранее приписаны статические оценки (приоритеты), тогда выбирается правило с наибольшей оценкой;
- выбирается более специализированное правило (с бóльшим количеством условий). Например, из правил:
  1. Если "птица", то добавить факт "умеет летать".
  2. Если "птица" и "пингвин", то добавить факт "не умеет летать"

  следует выбрать второе, поскольку оно более специализированное;
- реализуется стратегия выбора правила, в котором задействован последний появившийся в рабочей памяти факт;
- стратегия применения новых правил (не выбирается правило, которое только что было применено).

Шаг 3. Выполняются действия, записанные в правой части выбранного правила. При этом изменяется состояние рабочей памяти. Далее, либо будет обнаружено, что целевой факт присутствует в рабочей памяти (вывод построен), либо переходим к шагу 1.

При разработке продукционной системы существенной частью является выявление правил из информации о предметной области. Одной из распространенных моделей, которые помогают сформулировать правила, являются деревья решений. В нелистовых вершинах такого дерева находятся факторы, влияющих на принятие решения, и из каждой вершины выходит столько ветвей, сколько этот фактор имеет значений. Каждому листу дерева приписан вывод, который следует сделать при наличии факторов со значениями, упомянутыми на пути от корня дерева к данному листу. Пример дерева решений для оценки угрозы наводнения приведен на рисунке 4.3.

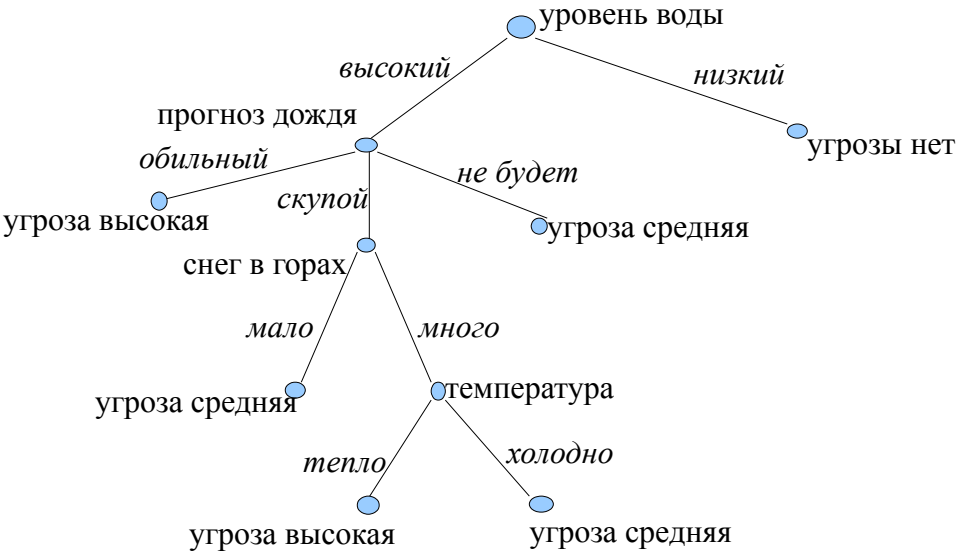


Рис 4.3 Пример дерева решений для оценки угрозы наводнения

Рассмотрим процесс построения этого дерева. Для начала при помощи специалиста в данной области нужно выделить факторы, от которых зависит угроза наводнения. Будем рассматривать четыре фактора: уровень воды в реке, прогноз дождя, наличие снега в горах, температуру воздуха. Для построения дерева могут быть использованы только факторы, имеющие конечный набор значений (если же это не так, то необходимо выделить конечное число подмножеств всех значений, которые существенно влияют на рассматриваемую задачу). Для упомянутых выше факторов значения приведены в таблице 4.1.

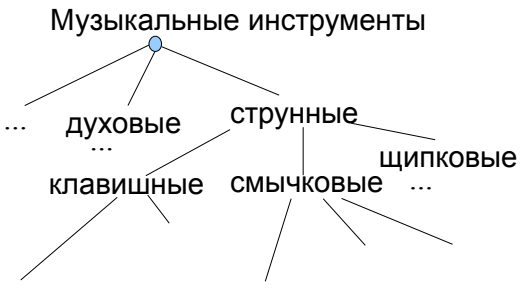
фактор, влияющий на угрозу наводнения	**рассматриваемые значения**
уровень воды в реке	высокий / низкий
прогноз дождя	обильные дожди / скупой / не будет
снег в горах	много / мало
температура воздуха	тепло / холодно

Таблица 4.1 Факторы, влияющие на угрозу наводнения

Какие конкретно значения уровня воды относятся к категориям "высокий" и "низкий" определяется после консультации со специалистом в данной предметной области. Выбираем один из факторов – уровень воды и помещаем его в корень дерева (рисунок 4.3). Данный фактор принимает одно из двух значений, поэтому у корня появляются две дочерние вершины. Приписываем этим вершинам либо следующий фактор, либо один из трех выводов: угрозы наводнения нет (никаких действий предпринимать не нужно), угроза высокая (нужно проводить специализированные работы), угроза средняя (требуется повышенное наблюдение, нужно готовить специализированную технику).

После построения дерева продукционные правила формируются следующим образом: каждое правило соответствует одному пути в дереве от корня к листовой вершине, в левую часть правила помещаются все факторы, проверяемые в вершинах этого пути с их значениями, а в правую часть — вывод, приписанный листу дерева, например: "`Если уровень воды низкий, то угрозы наводнения нет`".

В основе построения продукционных правил может также находиться и другой вид дерева, например, дерево классификации. На рисунке 4.4 приведен в качестве примера фрагмент дерева классификации музыкальных инструментов и продукции, сформулированные с помощью данного дерева.



- *Если муз. инструмент "струнный", "смычковый" и "маленького размера", то добавить факт "скрипка".*
- *Если муз. инструмент "струнный", "смычковый" и "большого размера", то добавить факт "контрабас".*
- *Если муз. инструмент "струнный", "клавишный" и "большой", то добавить факт "рояль".*

рояль скрипка ...
пианино контрабас

- *Если муз. инструмент "струнный", "клавишный" и "небольшой", то добавить факт "пианино".*

Рис. 4.4 Фрагмент дерева классификации музыкальных инструментов и выписанные по нему правила

Таким образом, при наличии дерева решений или дерева классификации, построенного с помощью эксперта в конкретной предметной области, задача формулировки правил становится существенно проще. При наличии запрограммированной управляющей стратегии, задача построения системы управления знаниями сводится к заданию продукционных правил и исходных данных. Такие возможности предоставляет, например, система CLIPS [23]

4.3 Среда CLIPS

Среда CLIPS (C Language Integrated Production System — интегрированная с языком Си среда построения продукционных систем) реализует цикл работы продукционной системы: в нее встроена управляющая стратегия. Система разрабатывалась в NASA (США) с 1984 года, в настоящее время является доступной по адресу http://clipsrules.sourceforge.net и поддерживается своим разработчиком Г. Райли.

Для построения работающей продукционной системы в среде CLIPS необходимо задать правила и факты, после чего можно будет автоматически получать из них выводы.

Сразу после запуска CLIPS – приложения на выполнение на экране появляется командная строка, с помощью которой можно заносить информацию в систему и выполнять служебные команды[6].

Вернемся к примеру из области классификации животных, рассмотренному на стр. 15 . Для работы данного примера требуется задать факты "животное желтого цвета", "имеет длинную шею"[7], что выполняется двумя командами assert, записывающими факты в рабочую память:

```
CLIPS > (assert (yellow animal))
```

Ответ системы (факт успешно записан):

```
==> f-1     (yellow animal)
<Fact-1>
CLIPS> (assert (has long neck))
```

Ответ системы (факт успешно записан):

```
==> f-2     (has long neck)
<Fact-2>
```

В языке CLIPS правила имеют следующий формат:

```
(defrule <имя правила>
    <необязательный комментарий>
    <необязательное объявление>
    <предпосылка_1>
    <предпосылка_2>
```

6 Более подробную информацию об использовании системы CLIPS можно найти, например, в книгах:
П. Джексон "Введение в экспертные системы,
А.П. Частиков, Т.А. Гаврилова, Д.Л. Белов "Разработка экспертных систем. Среда CLIPS"
и на сайте http://clipsrules.sourceforge.net

7 Для единообразной работы в различных операционных средах тексты примеров для CLIPS переведены на английский язык

```
      . . .
     <предпосылка_n>
=>
     <действие_1>
      . . .
     <действие_n>
)
```

С помощью предиката **defrule** определим правила классификации:

(defrule rule_1 “The first rule” (salience 10) (yellow animal) => (assert (loves sun)))	(defrule rule_2 (yellow animal) => (assert (savanna)))	(defrule rule_3 (white animal) (North Pole) => (assert (polar bear)))	(defrule rule_4 (savanna) (has long neck) => (assert (giraffe)))

Параметр **salience** показывает степень важности правила: при разрешении конфликтов выбираются правила с большей степенью важности. Параметр может принимать целочисленные значения из диапазона [-10000, 10000]. Если этот параметр опущен при определении правила, то он устанавливается равным 0.

Данные факты также могут быть сохранены в текстовом файле с расширением .CLP, содержимое которого нужно загрузить перед началом работы интерпретатора.

Запуск интерпретатора выполняется командной

```
CLIPS> (run)
```

Вывод на основе заданных фактов и правил позволяет получить в базе знаний факт **giraffe**, который и требовался по условию задачи.

После выполнения программы, как правило, тебуется очистка рабочей памяти системы с помощью команды

```
CLIPS> (reset)
```

Представление знаний в виде продукций имеет свои преимущества:

- правила не зависят друг от друга, что позволяет достаточно легко изменять содержимое базы знаний и организовывать ее в виде модульной структуры;
- простая структура правил;
- имеется возможность представления противоречивой информации.

Однако при работе с подобным представлением информация о каждом объекте оказывается распределенной по базе знаний. В целом, продукционное представление является понятным не только инженеру знаний, но и эксперту, поэтому оно является распространенным представлением для экспертных систем.

Задания к главе 4

4.1. Представить знания, связанные с решением некоторого класса задач в виде набора продукций. Привести пример решения конкретной задачи с использованием сформулированных продукционных правил (для решения использовать стратегию прямого или обратного вывода).

4.2. Реализовать в среде CLIPS модельную продукционную систему на основе правил, сформулированных при решении предыдущей задачи.

5. Системы, основанные на знаниях

В современном мире знания становятся особо ценным ресурсом и многие организации хотят их сохранить в таком виде, чтобы знания можно было передавать, автоматизированно обрабатывать. Рассмотрев некоторые из возможных вариантов представления знаний для их хранения и автоматизированной обработки, обратимся к системам, которые такие знания используют, — *системам, основанным на знаниях*.

Как правило, работа подобных систем ориентирована на конкретную предметную область, знания о которой в системе записаны, и на их основе система дает ответы на вопросы, получает новые знания. Для функционирования системы в рассматриваемой области используются как явные знания (патенты, руководства пользователя, содержимое баз данных), так и неявные (опыт сотрудников организации). В современной ситуации изолированное знание приносит меньше стратегической пользы, чем использование того же знания совместно с другими, поэтому возникает необходимость передачи, сохранения знаний, например, чтобы ими могли пользоваться сотрудники другого подразделения организации. Осознание полезности систем, которые могут копировать дорогостоящие или редко встречающиеся человеческие знания, привело к расцвету этой технологии в 80-90 годы прошлого века. В основе таких систем лежат знания специалиста-эксперта, представленные в специальном виде, подходящем для построения выводов, рассуждений, а задачей системы является принятие решений, совпадающих с решениями экспертов в заданной предметной области.

Экспертные системы не ориентированы на решение каких-то универсальных задач, они работают в конкретной предметной области, заданной их создателями — инженерами знаний и программистами. Фундаментом экспертной системы любого типа является база знаний, которая составляется на основе знаний специалистов. Правильно выбранный эксперт и удачное формальное представление его знаний позволяет наделить экспертную систему уникальными способностями и ценными знаниями, хотя она и остается при этом компьютерной программой. Врач, к примеру, хорошо диагностирует болезни и эффективно назначает лечение, обычно не потому, что он обладает некими врожденными способностями, а потому что имеет качественное медицинское образование и большой опыт в лечении своих пациентов. Ценность всей экспертной системы как законченного программного продукта во многом определяется именно качеством созданной базы знаний. Именно поэтому подобные системы еще называют системами, основанными на знаниях.

Одним их первых опытов успешного коммерческого использования систем, основанных на знаниях, была система XCON, разработанная совместно исследователями из Университета Карнеги – Меллон и группой интеллектуальных систем компании DEC в 1980 году. Система подбирала подходящую конфигурацию компьютерных систем на основе требований пользователя и заранее заданных ограничений. Заказчик подавал свой запрос подсистеме XSEL, формирующей из него внутреннее представление. С таким представлением работала уже непосредственно XCON, которая проверяла корректность заданной пользователем конфигурации. Если она была некорректной, то есть собрать компьютер по ней не удавалось, то система отправляла отчет в отдел обслуживания клиентов, специалисты которого объясняли заказчику суть возникших проблем. Корректная конфигурация передавалась в производственный отдел вместе с инструкциями по сборке. После появления

устройства, оно также передавалось в отдел обслуживания. Система автоматизировала часть производственного процесса, чем помогала выполнять заказы более эффективно, принося выгоду компании.

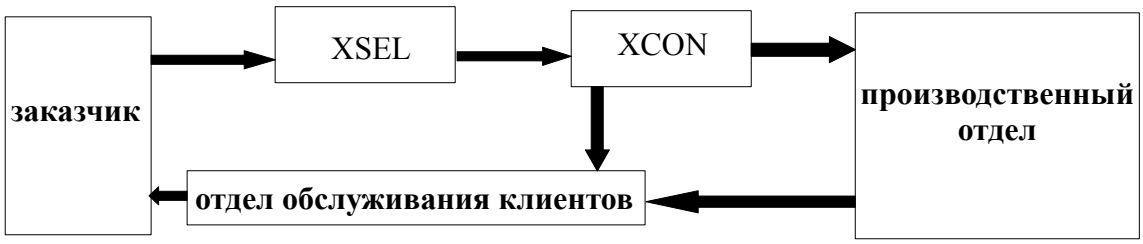


Рис. 5.1 Схема работы экспертной системы XCON

Система XCON по сути решала задачу структурного дизайна. Однако экспертные системы способны принести пользу и при решении других типов задач. Приведем классификацию задач, решаемых экспертными системами:

- задачи мониторинга
  Системы, занимающиеся решением задач этого класса, принимают входные данные от некоторого количества датчиков в течение продолжительного периода времени, наблюдая за определенными параметрами. Как только какой-то из них принимает недопустимое значение, либо недопустимой становится комбинация значений, система сигнализирует об этом и запускает выполнение определенных действий. Задачи, решаемые подобными системами, сложны для человека, потому что от постоянной монотонной работы он устает, и его внимательность снижается, либо же человек не успевает следить за большим количеством датчиков. Задачи мониторинга возникают при работе со сложными приборами и устройствами, и компьютерные системы способны здесь выступить в качестве помощника.
- диагностика
  Данная задача состоит в выборе одной или нескольких рекомендаций на основе исходных данных. Диагностика может делаться для разных областей: медицинская система ставит диагноз, выступая помощником врача; получив знания из автомобильной области, система становится "компьютерным автомехаником"...
- структурный дизайн
  В постановке задачи структурного дизайна задается множество статических (не зависящих от времени) ограничений и правила составления структуры из элементов. Такой структурой может быть, например, компьютер, элементами — его составные части (материнская плата, модули оперативной памяти, внешние запоминающие устройства). На основе заложенных в систему правил, можно сделать вывод о возможности (или, наоборот, невозможности) сборки компьютера заданной конфигурации.
- планирование, составление расписаний
  Системы, решающие задачи планирования, также получают в качестве входных данных набор ограничений, но их ограничения являются динамическими, то есть они должны быть выполнены в определенный промежуток времени. От времени зависит и множество доступных ресурсов. Так, если мы рассматриваем задачу составления расписания для института, то доступные ресурсы (свободные аудитории и свободное время преподавателей) зависят от

времени.

С точки зрения поддержки пользователя системы, основанные на знаниях, могут находиться на разных позициях (рисунок 5.2).

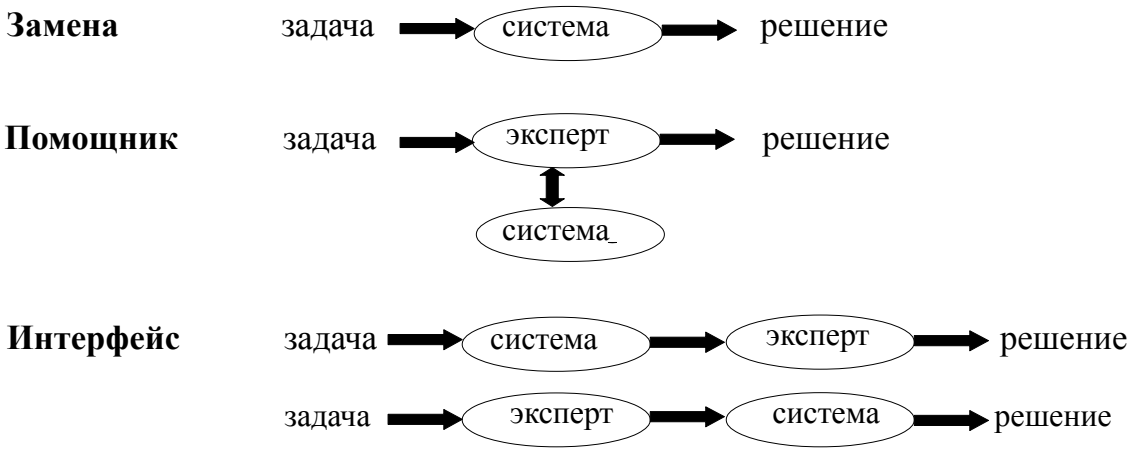


Рис 5.2 Классификация систем, основанных на знаниях, с точки зрения поддержки пользователя.

Во-первых, вариантом работы является замена специалиста-человека, на котором остаются лишь контролирующая функция. При таком подходе система, основанная на знаниях по сути является мозгом робота или другого механического устройства. Работа при минимальном участии человека особенно актуальна в "недружественных" для человека средах, например, в радиоактивной среде или в открытом космосе. Вторым, и наиболее распространенным вариантом, использования систем, основанных на знаниях, является их применение как помощника-консультанта: участие человека необходимо при постановки задачи системе, при принятии решения, система выступает его консультантом, зачастую подсказывая правила и альтернативы, которые человек может не учесть. Однако окончательное решение задачи принимает именно человек-специалист, и он же за него отвечает. Третьей позицией, которую может занимать система, является интерфейс, который может как помочь формализовать список требований (подобную задачу решала подсистема XSEL, рис. 5.1), так и выдать в формализованном виде результат решения задачи.

Использование систем, основанных на знаниях, дает организациям целый ряд преимуществ и в итоге приводит к финансовой выгоде. Наличие таких систем позволяет справляться с возрастающей сложностью задач. В промышленности и бизнесе сложность задач растет в связи с появлением большого количества информации, которую требуется учитывать при принятии решения. Так, например, за системой XCON было признано преимущество при ее сравнении с экспертами в области проектирования компьютеров. Другим преимуществом является возможность поручить работу системе при решении задач в неблагоприятных для человека условиях: при работе на вредном производстве. Также было оценено, что в среднем организация может использовать экспертную систему дольше, чем опыт одного конкретного специалиста. Увеличивается и "переносимость" знаний в различные подразделения организации. Для примера приведем организацию, занимающуюся ремонтом бытовой техники. Диагностировать неисправность порой может действительно только опытный мастер, количество таких специалистов ограничено, а проблемы с техникой возникают повсеместно. Но они могут быть решены и специалистом с небольшим опытом работы, имеющим в качестве помощника

экспертную систему. Совместно с системой он во многом способен действовать на уровне опытного специалиста. Кроме упомянутых преимуществ стоит отметить, что системы организуют удобный доступ к информационным ресурсам компании, играют роль "интеллектуального интерфейса", что помогает как в работе с клиентами, так и при обучении новых сотрудников. Также системы позволяют унифицировать требования в разных подразделениях одной и той же организации, например, при принятии решения о выдаче кредита в разных отделениях одного банка. На основе решения, предлагаемого системой в одинаковых ситуациях, сотрудники разных подразделений с большей вероятностью будут принимать похожие решения.

Системы, основанные на знаниях, способны принести пользу самым различным организациям. Однако, упоминая их положительные стороны, стоит обозначить и трудности, связанные с их разработкой и использованием. "Сила" экспертных систем заключается в знаниях, которые они используют. Но знания должны быть получены от специалистов и представлены в формализованном виде. И процесс получения знаний, и процесс их перевода в формальное представление являются нетривиальными задачами. Рассмотрим простую задачу: описать алгоритм перехода дороги. Последовательность действий можно сформулировать, например, так:

1. Остановиться у дороги.
2. Проверить, есть ли поблизости светофор.
3. Если светофора нет, то посмотреть сначала налево (убедиться, что нет приближающихся машин), затем направо (с той же целью).
4. Если есть приближающиеся машины, то следует подождать, а если их нет, то переходить дорогу.

Дополнительно нужно рассмотреть случай перехода дороги со светофором.
В данном описании часто встречающейся ситуации мы принимаем многие промежуточные решения автоматически, "по интуиции" и, пытаясь их формализовать, пропускаем или забываем описать такие шаги. Также и у эксперта присутствует очевидная для него информация, которую он забывает рассказать. Так, в примере о переходе дороги неявно предполагается, что мы находимся в стране с левосторонним движением транспорта, не рассмотрен случай наличия подземного или надземного перехода.

Другой проблемой является определение ответственности за неверно принятые системой решения, особенно если в результате такого решения был нанесен ущерб. Причиной ошибки могли являться как неверно записанные экспертные знания (которые неоднозначно сформулировал эксперт или же они были неверно проинтерпретированы при занесении в базу знаний), так и проблемы при реализации управляющей стратегии, неверный ввод исходных данных и т.д. Эта проблема, в частности, не позволяет полностью заменить присутствие специалиста системой.

В создании экспертной системы непосредственно принимают участие специалист-эксперт, программист-разработчик системы (или коллектив разработчиков), а "посередине" между ними находится инженер знаний, который из неформализованных знаний эксперта делает формальное представление и ставит конкретную задачу разработчику. С одной стороны, инженер знаний должен иметь представление о предметной области, для которой создается система, чтобы оценить уровень экспертизы используемых специалистов. С другой стороны, инженер знаний должен представлять проблемы, возникающие при проектировании и реализации.

Процесс перевода в формальное представление мыслей, способов рассуждений и

знаний экспертов называется *приобретением (извлечением) знаний*. Также он может включать работу с литературой, чертежами и рисунками, техническими описаниями и другими письменными и электронными источниками информации. Процесс приобретения знаний состоит из трех этапов:

1. Получение информации от эксперта с использованием подходящего метода.
2. Интерпретация полученной информации.
3. Использование интерпретации для построения правил, моделирующих процесс принятия решения экспертом.

Одним из распространенных методов извлечения знаний является беседа с экспертом, которому предлагается ответить на вопросы: "Что Вы делаете на начальном этапе решения задачи?", "Какую информацию рассматриваете далее?" и "Какие ограничения Вы учитываете при построении решения?" На основании беседы с экспертами инженер знаний формирует набор сценариев (типичных ситуаций). Если требуется, после их анализа, беседа с экспертом будет продолжена для прояснения отдельных вопросов. Итогом становятся знания, представленные в некотором формализованном виде: диаграммы, деревья решений и т.д.

Инженер знаний должен быть способен оценить, насколько знания эксперта подходят для конкретной области. К примеру, если речь идет о создании системы мониторинга состояния спортсменов, занимающихся на тренажерах в зале, то более ценным окажется опыт инструктора по фитнесу, чем, например, врача-офтальмолога с б**о**льшим стажем работы. Также инженер знаний должен оценить уровень эксперта. На практике эксперты различного уровня могут оказаться полезными. Солидные эксперты с большим опытом работы способны достаточно широко взглянуть на проблему, описать общие шаги ее решения. Однако общение с ними может оказаться непростым для инженера знаний, так как они используют специализированные термины и, как правило, заостряют внимание на ключевых шагах в решении задачи, пропуская детали. Эксперты среднего уровня обычно хоть и обладают меньшим опытом, но с ними проще найти общий язык инженеру знаний. Специалисты-новички зачастую имеют более фрагментарные знания, однако, поскольку многие задачи им приходится самим решать "с нуля", они могут наиболее подробно описать шаги решения, что является важным для системы, основанной на знаниях.

Одним из основных методов получения знаний является беседа с экспертом, которая может быть проведена как в свободной, так и в структурированной форме. В последнем случае кроме вопросов эксперту даются и варианты ответов (выбор ответа из нескольких вариантов, отображение ответа на шкале). Структурировать беседу часто приходится в случае, если работа проводится с несколькими экспертами. Важным моментом при подготовке интервью является корректность и однозначность поставленных вопросов. Вопросы должны быть логически корректными и одинаково понимаемыми разными людьми.

При работе с несколькими экспертами появляются как дополнительные преимущества (такие, как более подробное описание предметной области), так и дополнительные сложности. Инженеры знаний нередко сталкиваются с ситуацией, когда на один и тот же вопрос эксперты дают несовпадающие ответы. Это связано с их индивидуальным восприятием еще не формализованной задачи. Поэтому для работы с несколькими экспертами были предложены специальные методы извлечения знаний.

Мозговой штурм — метод, основанный на спонтанном высказывании идей, которые формулируются и высказываются участниками в краткой и четкой форме.

После того, как все идеи были высказаны, поводится их обсуждение.

Метод номинальной групповой техники предлагает участникам выбранной группы экспертов изложить свои идеи по поводу решения той или иной задачи из предметной области в письменном виде. На следующем этапе каждый участник докладывает суть своего проекта, а представленные варианты рассматриваются членами группы без обсуждения и критики. Затем каждый член группы опять независимо от остальных, в письменном виде проставляет оценки рассмотренных идей. Решение, получившее наивысшую оценку, признается лучшим.

Метод Дельфи получил свое название от Дельфийского оракула, способного предсказывать будущее. Метод предполагает, что эксперты взаимодействуют только анонимно. Такой метод полезен, когда сбор группы экспертов невозможен: например, если в состав участников решения проблемы включены специалисты из различных филиалов и подразделений организации, географически отдаленные друг от друга и от центрального офиса организации. Поиск решения задачи происходит в несколько шагов:

1. Членам группы предлагается ответить на детально сформулированный перечень вопросов по рассматриваемой проблеме.
2. Каждый член группы отвечает на вопросы независимо и анонимно.
3. Результаты ответов собираются в центре, и на их основании составляется общий документ, содержащий все предлагаемые варианты решений.
4. Каждый член группы получает копию этого материала.
5. Ознакомление с предложениями других участников может изменить мнение по поводу возможных вариантов решения проблемы.

Шаги 4 и 5 повторяются столько раз, сколько необходимо для достижения согласованного решения. Как и при использовании номинальной групповой техники, здесь обеспечивается независимость мнения отдельных экспертов.

Когда, наконец, набор знаний экспертов получен и представлен в некотором виде (таблицы, деревья решений, наборы правил и т.д.) возникает вопрос о проектировании системы. Традиционной для экспертных систем является архитектура, изображенная на рисунке 5.3.

В состав экспертной системы входят: решатель, рабочая память (база данных), база знаний, подсистема приобретения знаний, подсистема объяснений, интерфейсный (диалоговый) компонент.

Решатель предназначен для того, чтобы, используя исходные данные из рабочей памяти и знания из базы знаний, формировать такую последовательность правил, которая, позволяет решить необходимую задачу. Решатель управляет процессом выбора и применения правила.

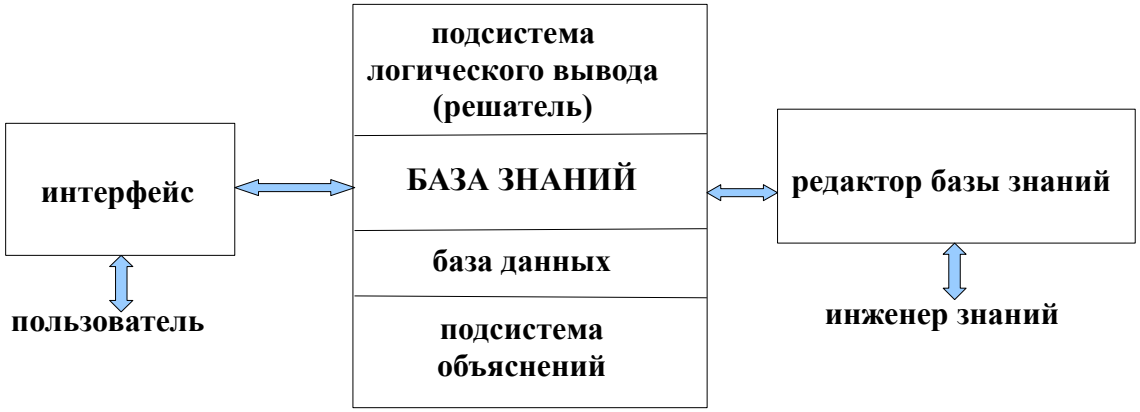


Рис.5.3 Архитектура экспертной системы.

В базе данных (рабочей памяти) хранятся исходные и промежуточные данные решаемой задачи. База знаний необходима для хранения долгосрочных данных, описывающих рассматриваемую область, и правил, описывающих необходимые действия над данными этой области. Содержимое этой базы таково, что на его основе можно получать новые знания и информацию.

Компоненты приобретения знаний (редактор базы знаний) автоматизируют процесс наполнения экспертной системы знаниями.

Подсистема объяснений интерпретирует в доступную для пользователя форму методы решения задачи, отвечает на вопрос, почему был выполнен тот или иной шаг в процессе вывода. Это облегчает эксперту тестирование системы и повышает доверие пользователя к полученному результату.

Интерфейсный компонент необходим для организации дружественного общения с пользователем, как в ходе решения задач, так и в процессе приобретения знаний и объяснения результатов работы. С его помощью в систему вносятся вопросы и поправки, а также визуализируются ответы.

Работа экспертной системы может осуществляться в двух режимах: режиме приобретения знаний и режиме консультации. Режим консультации еще называют режимом решения задачи или режимом использования экспертной системы. В режиме приобретения знаний инженер знаний добавляет новые знания, полученные от эксперта. Они позволяют экспертной системе в режиме использования уже самостоятельно решать задачи из проблемной области. В режиме консультации общение с экспертной системой осуществляет конечный пользователь, которого интересует результат и, возможно, способ его получения. Пользователь может не быть специалистом в данной проблемной области, в зависимости от назначения системы (в этом случае он обращается только за конечным результатом), или быть специалистом (в этом случае он может сам получить результат, но хочет ускорить процесс получения результата, либо возложить на систему рутинную работу). В режиме консультации данные о задаче, после обработки их диалоговым компонентом, поступают в рабочую память. Решатель, руководствуясь входящими данными из рабочей памяти, общими данными о проблемной области и правилами из базы знаний, формулирует решение задачи. Если ответ системы не понятен пользователю, то он может обращаться за объяснениями: "Почему система задает вопрос?", "Как был получен ответ?"

В заключение стоит отметить, что несмотря на сложности, связанные с разработкой экспертных систем, и имеющиеся ограничения по их использованию, системы уже доказали всю свою ценность и значимость во многих важных приложениях, выступая в роли помощника – специалиста.

Задание к главе 5

5.1. Попробуйте выступить в роли инженера знаний для решения следующей задачи: сформулируйте правила, критерии, стратегии поведения, которые помогают первокурснику успешно сдать первые две сессии. Для решения задачи выберите одного или нескольких экспертов, оцените их уровень знания предметной области, получите и переведите их знания о данной области в формальный вид.
Результат работы представьте в виде дерева решений.

6. Рассуждения на основе прецедентов

Рассмотренные системы, основанные на знаниях, построены на базе правил, полученных от экспертов. Однако в этом кроется и их слабая сторона: далеко не всегда эксперт способен сформулировать, почему именно он сделал тот или иной вывод. Для эксперта такая ситуация не представляет проблем, если вывод был правильным. А вот в случае работы системы, требуется объяснение, как именно она построила решение. Проблемой является и разное представление одних и тех же фактов экспертами: та температура воздуха, которую один эксперт назовет "холодной", может быть названа "прохладной" по мнению другого эксперта. А для системы данные два значения могут оказаться существенно различными. Неверная интерпретация слов эксперта инженером знаний и неверный перевод информации в систему также приводят к ошибкам. С ростом же количества внесенных в систему знаний, контролировать правильность ее выводов человеку становится сложнее и сложнее. Работа встроенного механизма вывода замедляется с ростом пространства поиска решения. Эти факторы существенно влияют на перспективность использования экспертных систем.

Другая принципиальная сложность, связанная с разработкой экспертных систем, возникает при работе с новыми, не до конца изученными областями. Для таких областей эксперты еще просто не успели появиться, однако обмен опытом и знаниями о работе все равно оказывается необходимым. Так при появлении в 2009 г. нового вируса гриппа H1N1 опыт его лечения передавался поначалу в виде описаний успешных случаев применения тех или иных препаратов, которые назначались по аналогии с другими вирусами гриппа, а общей схемы (правил) лечения на начальном этапе сформулировано не было.

Для подобных областей был предложен другой подход к созданию интеллектуальных систем: если экспертов нет или их знания получить сложно, нужно записать в систему источники, из которых сами эксперты получают знания, то есть набор конкретных ситуаций и решений, которые в них принимались. Такие ситуации, описывающие прошлый опыт, называются прецедентами, а метод получения знаний с их помощью - *рассуждением на основе прецедентов* (case-based reasoning, CBR). В основе подхода лежит идея, что, принимая решение в конкретной ситуации, люди вспоминают похожую на нее, с которой приходилось сталкиваться в прошлом, и действуют по аналогии с прошлым опытом. Рассмотрим пример:

> *"Приехав на машине в небольшой и уютный городок, я собрался припарковаться под очаровательным цветущим деревом. Но потом вспомнил, как позавчера уже оставлял машину в Москве (около работы) под таким деревом, и опавшие с него цветы сделали всю ее липкой, словно мокрый леденец. Пришлось мне тут отъехать от дерева на пару метров."*

В данном случае решается конкретная задача: выбор места для стоянки машины. Информации о конкретном городе нет, однако есть опыт выбора парковки в другом городе. Этот опыт "адаптируется" к конкретной ситуации, и на основе него принимается решение.

Системы, реализующие рассуждения на основе прецедентов содержат в себе следующие компоненты:

- библиотеку прецедентов - описание некоторых ситуаций и принимаемых в них решений;
- средства поиска "подходящего" прецедента в библиотеке (с учетом того, что

решаемая задача хоть и похожа, но обычно не совпадает с библиотечным прецедентом);

- средства, позволяющие адаптировать имеющееся библиотечное решение, к конкретной задаче;
- средства пополнения библиотеки прецедентов.

Цикл работы систем, применяющих для построения выводов рассуждения а основе прецедентов, приведен на рисунке 6.1. При получении задачи система осуществляет поиск похожих на нее прецедентов. Далее, из них выбирается один, признанный самым подходящим. Соответствующее решение из библиотеки прецедентов адаптируется для решения поставленной задачи. Иногда возникает ситуация, что в библиотеке найден прецедент, содержащий такую же задачу, поэтому адаптация не потребуется. После этого предложенное системой решение должно быть оценено пользователем как верное или нет, и решенная задача сохраняется в системе как новый прецедент. Из неверного решения система извлекает опыт: например, фиксирует неподходящие правила адаптации, чтобы не применять их в дальнейшем.

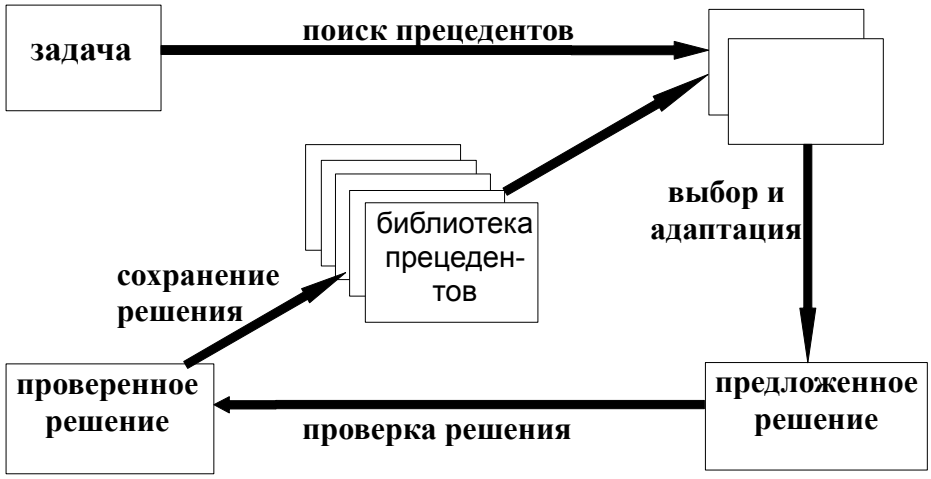


Рис. 6.1. Цикл работы системы, реализующей рассуждения на основе прецедентов

При практической реализации метода рассуждений на основе прецедентов возникают три основные проблемы: выбор способа организации библиотеки прецедентов, определение "похожести" прецедентов и формулировка правил адаптации. Для библиотеки, как правило, требуется индексирование ее содержимого, и реализация быстрого алгоритма поиска. Скорость поиска в библиотеке прецедентов является существенным параметром, определяющим полезность системы. Определение похожести зависит как от предметной области, так и от класса решаемых в ней задач. Так, в области туризма, "похожими" могут как считаться путевки в места с похожим климатом, так и имеющие приблизительно равную стоимость (вне зависимости от географического положения курорта). Правила адаптации также определяются конкретной предметной областью, часть из них может быть сформирована в процессе работы системы, на основе опыта решения системой некоторого набора задач.

Приведем примеры нескольких систем, использующих рассуждения на основе прецедентов. В 1980-е годы была создана система HYPO [2], которая играла роль помощника при обучении студентов-юристов методике ведения судебных дел. И

расследования, и рассуждения в юриспруденции направляются аргументацией, для которой необходимо знать как законы, так и судебную практику, которая и была сформулирована в виде библиотеки прецедентов. Выбор подходящего прецедента должен был учитывать как информацию о текущем случае, так и возможные "ходы" оппонентов, чтобы предусмотреть на них подходящий ответ. Эксперты в кулинарной области-французы разработали несколько систем, предлагающих кулинарные рецепты на основе заданных предпочтений пользователя: созданная в 80-х годах прошлого века система CHEF [7] и система 21 века Taaable (www.taaable.fr) в своей работе используют множество известных рецептов, представляющих собой библиотеку прецедентов. Системы выполняют выбор и адаптацию рецепта из базы прецедентов в соответствии с запросами пользователя.

В настоящее время существуют также инструментальные средства для разработки систем, использующих рассуждения на основе прецедентов. Примерами таких инструментов являются CBR Express и CasePoint, разработанные компанией Inference.

Рассуждения на основе прецедентов используются в системах автоматизации информационно-справочных служб и "горячих линий", системах доступа к информации. При общении с системой сначала вводится простой запрос, например: "Мой компьютер не работает". Из запроса выделяются ключевые слова, по ним происходит поиск в базе прецедентов, и формируется перечень потенциальных решений. Пользователю могут быть также заданы уточняющие вопросы. Предлагаемые варианты решения проблемы могут включать в себя не только тексты, но и видео- или фотоматериалы. Технология вывода по прецедентам представляет собой основу для практически безграничных приложений, которые наращиваются за счет постоянного сбора информации (причем обеспечивается совмещение структурированных и неструктурированных данных, включая мультимедиа). По мнению компаний, активно использующих эту технологию, таких как Nippon Steel, Lockheed и некоторых других, создается самообучающаяся коллективная память, удобная для накопления и передачи профессионального опыта.

Задание к главе 6

6.1. Составьте библиотеку прецедентов и предложите способы формального определения похожести прецедентов и адаптации для некоторой предметной области.

> Пример: запишите, какие блюда в течение недели вы ели на завтрак, обед и ужин. Дополните описания специальными атрибутами, поясняющими, почему было выбрано то или иное блюдо (например, "по случаю праздника", "потому что не было времени готовить" и т. д.) Из полученной информации постройте библиотеку прецедентов, предложив подходящий способ ее организации. Используйте библиотеку прецедентов для ответа на вопрос вида: "Предложите для заданных обстоятельств конкретную еду".Продемонстрируйте построение нескольких решений, пояснив, как определена похожесть прецедентов и какая адаптация выполнена.

В качестве примера может быть рассмотрена и другая предметная область.

7. Фреймовое представление знаний. Онтологии

В упомянутых ранее способах представления знания о каждом конкретном

объекте оказывались распределенными по всей базе знаний. В 70-х годах прошлого века Марвином Минским был предложен подход, ориентированный на представление объектов, их свойств и взаимосвязей с помощью фреймов.

Фрейм[8] — это структура данных для представления стереотипной ситуации, некоторого объекта, события, понятия. Структура имеет имя и набор слотов. Каждый слот соответствует одному элементу или свойству определяемого понятия, он имеет название и значение. Структуру фрейма можно представить следующим образом:

```
Имя фрейма: <имя>
(<имя 1-го слота> <значение 1-го слота>)
(<имя 2-го слота> <значение 2-го слота>)
          ...
(<имя n-го слота> <значение n-го слота>)
```

Марвин Минский полагал, что процессы человеческого мышления базируются на хранящихся в его памяти материализованных, многочисленных запомненных структурах данных - фреймах, с помощью которых человек осознает зрительные образы (фреймы визуальных образов), понимает слова (семантические фреймы), рассуждения, действия (фреймы-сценарии), повествования и т.д. Процесс понимания при этом сопровождается активизацией в памяти соответствующего фрейма и его сопоставлением с наблюдаемой ситуацией или объектом. В случае неудачного сопоставления из памяти, на основе связей фреймов между собой, "выбирается" другой фрейм, который может оказаться более подходящим для рассматриваемой ситуации. Так, например, увидев автомобиль, человек обычно узнает, что объект, находящийся перед ним является автомобилем, даже если такую модель машины он раньше никогда не видел. Процесс последовательной замены одного фрейма другим особенно наглядно проявляется в таких областях человеческого мышления, как понимание естественного языка, рассуждение, вывод по аналогии. Это следует из наших интуитивных представлений о процессе мышления, который начинается с наводящих на мысль, но несовершенных идей, последовательно заменяемых все лучшими и лучшими идеями и в итоге приводящими к результату.

Различают два вида фреймов: фреймы-образцы (прототипы), которые хранятся в базах знаний и фреймы-экземпляры, которые создаются на основе поступающих данных для описания ситуации, конкретной задачи. Структура таких видов фреймов является одинаковой, но у каждого из них есть специфические слоты, которые позволяют отразить иерархию понятий и использовать наследование свойств. На рисунке 7.1 приведены примеры фрейма-образца и фрейма-экземпляра, которые могут быть использованы в системе, основанной на знаниях, для библиотечной области.

Имя фрейма - это идентификатор, присваиваемый фрейму. Фрейм должен иметь имя, единственное в данной фреймовой системе. Каждый фрейм состоит из произвольного числа слотов, причем несколько из них обычно определяются самой системой для выполнения специфических функций, а остальные определяются пользователем.

Книга	
is_a	<печатное издание>

имя фрейма

Книга_25	
instance_of	<Книга>

8 В переводе с английского frame — скелет, остов, рамка.

автор		слоты	автор	"А. Дюма"
название			название	"Асканио"

Рис. 7.1 Фрейм-прототип Книга и фрейм-экземпляр Книга_25

Специальные слоты служат для редактирования базы знаний и управления выводом во фреймовой системе. В число специальных слотов входят

- *is_a*, *a_kind_of,* показывающий фрейм-родитель для данного фрейма-прототипа,
- *instance_of,* сообщающий, экземпляром какого фрейма-прототипа является данный фрейм-экземпляр,
- *has_part*, упоминающий составные части объекта,
- слот указателей дочерних фреймов, который является списком указателей этих фреймов,
- слот для ввода имени пользователя, даты определения, даты изменения, текста комментария.

С помощью специальных слотов отражаются иерархические зависимости между объектами. В любой предметной области присутствуют объекты и понятия, связанные иерархическими отношениями. Вспомним любое из определений, которые встречаются в учебниках: "косинус — это функция, которая...", "глагол — это часть речи...". Упоминаемые в них понятия (косинус, глагол) как правило определяются с помощью уже введенных ранее понятий (функция, часть речи). Это позволяет сделать определения более краткими, и унаследовать свойства базового понятия. Так, из того, что объект книга_25 является книгой (это свойство выражено как instance_of на рисунке 7.1) будет следовать, что у него есть все характеристики, которыми обладает книга. Книга же является печатным изданием, а значит обладает его характеристиками (у него есть издательство, номер ISBN или ISSN[9] и т.д.)

Значением слотов может быть практически любой объект: число, строка, ссылки на другие слоты данного фрейма или на другие фреймы. Со значением слота может быть связана так называемая присоединенная процедура или процедура-демон. Такие процедуры запускаются автоматически при возникновении некоторого события. Процедура с условием "if-needed" запускается, если в момент обращения к слоту его значение не было установлено. Такая процедура может, например, взять отсутствующее значение из родительского фрейма. Процедура "if-added" запускается при попытке изменения значения слота. Например, если один из фреймов хранит количество зимних путешествий, то при изменении у одного из экземпляров путешествий значения слота с "лето" на "зима", количество должно измениться, и это изменение способна выполнить присоединенная процедура. Еще одним примером условия запуска процедуры является "if-removed", которая вызывается при попытке удаления значения слота. Например, если во фреймовой системе есть фрейм-экземпляр, описывающий класс школьников, то при удалении последней фамилии из списка значений слота "ученики", фрейм-экземпляр класс также следует удалить (считаем, что класс не может быть пустым). Удаление объекта и будет выполнять присоединенная процедура с условием "if-removed", связанная со слотом "ученики".

9 ISBN (англ. International Standard Book Number) — уникальный номер книжного издания, необходимый для автоматизации работы с изданием.
ISSN (англ. International Standard Serial Number) — Международный стандартный серийный номер — уникальный номер, позволяющий идентифицировать любое периодическое издание независимо от того, где оно издано.

Вывод информации во фреймовых системах производится путем сопоставления, с помощью наследования и с помощью присоединенных процедур. Вопрос, заданный фреймовой системе формулируется в виде фрейма, некоторые слоты которого не заполнены, и требуется получить их значения в процессе вывода. Например, пусть в системе хранится информация о птицах и требуется дать ответ на вопрос: "Какого цвета птицы, которые не умеют летать?" Запишем вопрос в виде фрейма:

```
(?
(is_a           Птица    )
(цвет           ?        )
(умение летать   "нет"     )
          ...
)
```

Для ответа на заданный вопрос система выполняет сопоставление с известными фреймами, которые имеют значения слоты, совпадающие по названию и значению с вопросом. Предположим, среди фреймов был найден следующий:

```
(Королевский пингвин
(is_a            пингвиновые)
(цвет            "черно-белый с рыжими пятнами")
(умение летать    "нет" )
)
```

Один из его слотов совпадает по значению с вопросом, однако требовалось найти объект, являющейся птицей. Данная информация может быть получена с использованием свойства наследования. Для этого мы будем двигаться по иерархии наследования, используя отношения *a_kind_of* и *is_a*. Согласно первому слоту, королевские пингвины являются представителями семейства пингвиновых, поэтому фрейм пингвиновые

```
(пингвиновые
(a_kind_of    Птица)
   ...
)
```

также будет участвовать в рассмотрении ответа на вопрос. Поскольку в нем присутствует слот с информацией, что пингвиновые являются птицами, данное свойство наследуется и фреймами-потомками. Сопоставление с вопросом проходит успешно, и ответом является значение "черно-белый с рыжими пятнами". Таким образом при выводе был использован механизм сопоставления и информация, полученная по иерархии наследования. Также часть свойств может быть выведена и проверена с помощью присоединенных процедур.

Однако не всегда свойства родительского фрейма наследуются потомками. В примере о пингвине упомянут фрейм "Птица", экземпляры которого обычно умеют летать:

```
(Птица
(is_a           Животные)
(умение летать   "да" )
   ...
)
```

Если для фрейма-потомка требуется указать значение слота, отличное от родительского, его можно явно задать, как оно было задано для фрейма "Королевский пингвин". Некоторые фреймовые системы позволяют также указать для определенных слотов фрейма, что их значения не наследуются.

Фреймовое представление знаний нашло свое дальнейшее развитие и в объектно-ориентированном программировании, и в представлении знаний в виде онтологий.

Термин *онтология* имеет различные определения, мы рассматриваем его с точки зрения информатики, так как предполагается использовать онтологии в компьютерных системах управления знаниями. При таком подходе онтология – это формальное явное описание понятий (классов) в рассматриваемой предметной области, свойств (слотов) каждого понятия, и ограничений, наложенных на слоты. Онтология вместе с набором индивидуальных экземпляров классов представляет собой объектно-ориентированную модель некоторой области и образует базу знаний.

В зависимости от содержимого, онтологии делятся на общие, ориентированные на задачу и предметные онтологии.

Общие онтологии описывают наиболее общие понятия (пространство, время, материя, объект, событие, действие и т.д.), которые независимы от конкретной проблемы или области. Примером такой онтологии является CYC (www.opencyc.org), разрабатываемая с 1994 года и являющаяся базой энциклопедических знаний обо всех областях.

Онтология, ориентированная на задачу - это онтология, используемая конкретной прикладной программой. Она отражает специфику приложения, но может также содержать некоторые общие термины (например, в графическом редакторе будут и специфические термины - палитра, тип заливки, наложение слоев и т.д., и общие - сохранить и загрузить файл). Задачи, которым может быть посвящена онтология, могут быть самыми разнообразными: составления расписания, определение целей, диагностика, продажа, разработка программного обеспечения, классификация. При этом онтология задачи использует специализацию терминов, представленных в общих онтологиях.

Предметная онтология (или онтология предметов) описывает реальные предметы, участвующие в какой-либо деятельности (производстве). Например, это может быть онтология всех частей и компонентов автомобилей определенной марки (Audi), сведения об их характеристиках, поставщиках и т.п.

В рамках курса мы остановимся на онтологиях, которые приносят пользу при решении практических задач управления знаниями. Это преимущественно предметные онтологии и онтологии, ориентированные на задачи. Примером может служить онтология проекта TOVE[10] (Toronto Virtual Enterprise). В проекте была поставлена цель создания модели данных, которая

- обеспечивает общую терминологию для предметной области, ее приложения могут совместно использоваться и пониматься каждым участником общения;
- дает точное и по возможности непротиворечивое определение значения каждого термина на основе логики первого порядка;
- обеспечивает задание семантики с помощью множества аксиом, которые автоматически позволяют получать ответ на множество вопросов о предметной области.

Виртуальное предприятие TOVE должно обеспечить построение интегрированной модели некоторой предметной области, состоящей из онтологий: операций, состояний и времени, организации, ресурсов, продуктов, сервиса, производства.

Одним из инструментов создания онтологий является система Protégé[11]. Система позволяет описывать классы, экземпляры классов, редактировать их слоты,

10 Web-страница проекта Tove: http://www.eil.utoronto.ca/enterprise-modelling/tove/index.html
11 Web-страница системы Protégé: http://protege.stanford.edu/

устанавливать связи между объектами. В практике ее применения создание онтологий включает следующие этапы:

1. Определение классов в онтологии.
2. Организация иерархии классов.
3. Определение слотов и их допустимых значений (задание ограничений).
4. Заполнение значений слотов для экземпляров классов.

Процесс разработки онтологии является итеративным, пример разработки подробно описан в [17]. При правильном моделировании, онтология может быть представлена предложениями, где вероятней всего в качестве существительных будут объекты, а отношений – глаголы. Так как онтология представляет предметную область в реальном окружающем мире, понятия в онтологии также должны отражать эту реальность. После создания варианта онтологии проводится его оценка экспертами. С учетом их замечаний процесс разработки продолжается.

В управлении знаниями информация и знания, представленные в виде онтологии могут быть использованы в трех ключевых процессах: в распространении знаний, интеграции информации и знаний из различных источников и автоматизированном получении новых знаний (логическом выводе) [15].

Задания к главе 7

7.1 Приведите пример фрейма–образца и фрейма–экземпляра для некоторой предметной области.
7.2 Для выбранной предметной области составьте набор фреймов с помощью системы Protégé. Продемонстрируйте пример вывода информации.
В качестве примера может быть выбрана онтология для системы, которая помогает организовать работу учебного отдела факультета. Система должна отвечать на следующие вопросы: "Кто ответственный за каждый предмет?", "Какова программа каждого экзамена и кто является экзаменатором?", "Какое количество аудиторий требуется для проведения экзамена?"

8. Хранилища данных

Чем больше в организации накапливается информации, тем сложнее становится хранить ее на бумажных носителях или запоминать. К несчастью, бумажные документы имеют ограниченный доступ, и их трудно изменять. А если из организации уходит высококвалифицированный специалист, потеря его ценных знаний и опыта зачастую оказывается невосполнимой. Поэтому многие организации переходят к использованию хранилищ данных.

Информационные системы в бизнес-сообществе можно поделить на две группы: учетные системы и аналитические. Для первых важно поддерживать корректное выполнение заранее известного набора транзакций: например, обрабатывать покупки в интернет-магазине. При этом нужно учитывать, что процесс заказа товара в магазине мог быть прерван на середине (так как у пользователя пропал доступ к сети), либо же два пользователя могли попытаться купить один единственный компьютер, находящийся на складе, — во всех подобных случаях база данных магазина должна содержать информацию о реально имеющихся товарах. Вторая группа, аналитические системы, работает с данными, относящимися ко всей деятельности организации. Для интернет-магазина подобными данными являются записи обо всех покупках,

сделанных клиентами за все время работы магазина. Имея инструменты для анализа подобных данных, можно сделать вывод, какой товар предложить тому или иному покупателю, стоит ли компании вводить в работу дополнительные серверы на период новогодних праздников и т. д. Множество вопросов, ответы на которые могут находиться в подобных данных, заранее не ограничивается. Данные для таких систем содержатся в хранилищах данных.

Хранилища данных отличаются от традиционных баз данных тем, что они проектируются для поддержки процессов принятия решений, а не просто для эффективного сбора и обработки данных. Как правило хранилище содержит многолетние версии обычной базы данных, физически размещаемые в той же самой базе. Данные в хранилище не обновляются на основании отдельных запросов пользователей. Вместо этого вся база данных периодически обновляется целиком.

Хранилище данных (Data Warehouse) — это репозиторий, содержащий данные организации в исторической перспективе с целью их использования для функционирования организации: для контроля текущей деятельности (подготовки отчетов) и поддержки принятия решений. В одном хранилище могут находиться данные из разных предметных областей, касающиеся разных отделов организации. Данные поступают в хранилище из различных источников, а обновление их происходит согласно заранее установленному регламенту. Области, где используют хранилища данных, различны: транспортная, медицинская (фармацевтическая), финансовая и другие. Данные, содержащиеся в хранилищах, помогают аналитикам ответить на вопросы, какие продукты и сервисы следует развивать, как грамотно распределять поставки товаров и транспорт по регионам...

Хранилища данных, как правило, создаются согласно следующим принципам:

- информация группируется по предметным областям, а не в зависимости от приложений, которые с ней работают;
- собираются данные, относящиеся ко всей деятельности организации, а не только к одной области решаемых ею задач;
- данные меняются согласно расписанию, после начальной загрузки изменение происходит в определенное время, дополнительного редактирования или дополнения данных не делается, данные не удаляются;
- данные в хранилище привязаны к некоторому промежутку или моменту времени.

Архитектура хранилища данных состоит из нескольких уровней (рисунок 8.1). *Уровень базы данных* рассматривает исходные данные, в качестве которых выступают базы данных и СУБД, архивы, разрозненные файлы известных форматов, а также любые иные источники структурированных данных.

Уровень доступа к данным включает инструменты, позволяющие проводить выборку и загрузку данных, изменять формат данных (ETL - Extract, Transform and Load). Основная задача ETL – извлечь данные из разных систем, привести их к согласованному виду, очистить (исправить ошибки и несоответствия) и загрузить в хранилище. Программно-аппаратный комплекс, на котором реализована система ETL, должен обладать значительной пропускной способностью и высокой вычислительной производительностью.

Роль следующего уровня – надежное, защищенное от несанкционированного доступа, *хранение данных.* Зоны временного хранения нужны для реализации специфических бизнес – процессов, например, когда перед загрузкой данных контролер должен их проверить и дать разрешение на их загрузку в хранилище. Метаданные описывают,

как хранятся данные, чтобы освободить пользователя от стандартизованного представления.

Следующий уровень позволяет доставить данные в различные витрины в соответствии с правами доступа, графиком доставки и требованиями к составу данных.

Уровень предоставления состоит из витрин данных, каждая из которых рассчитана на решение конкретных задач. Витрины группируются по различным признакам: территориальным, тематическим, организационным, и т. д. Этот уровень является базой для систем построения отчетов и анализа данных.

<table>
<tr><td>документы</td><td rowspan="4">ETL</td><td>метаданные</td><td rowspan="4">Выборка, реструктуризация, доставка</td><td>Тематическая витрина данных</td><td rowspan="4">==></td><td rowspan="4">Инструменты анализа, построения отчетов, планирования</td></tr>
<tr><td>файлы</td><td></td><td></td></tr>
<tr><td rowspan="2">архивы</td><td>Центральное хранилище</td><td>Региональная витрина данных</td></tr>
<tr><td rowspan="2">транзакционные системы</td><td rowspan="2">Зона временного хранения</td><td rowspan="2">Витрина данных подразделения</td></tr>
</table>

Рис. 8.1 Архитектура хранилища данных

Рассмотрим процесс построения хранилища данных на предприятии. По оценкам западных специалистов, при создании удачных проектов хранилищ данных лишь половина финансовых средств тратилась на аппаратно-программные средства, а другая уходила на консультации. Создавая хранилище данных, имеет смысл рассмотреть те области, где внедрение хранилищ уже показало положительный результат: планирование продаж, прогнозирование и управление, проектирование и разработка новых видов продукции, интеграция цепочки поставок. Целый ряд фирм занимается созданием программного обеспечения для поддержки хранилищ данных. Основная цель создания хранилища данных в том, чтобы сделать все значимые для управления бизнесом данные доступными в стандартизованной форме, пригодными для последующего анализа и получения из одного источника необходимых отчетов. Чтобы достигнуть этого, необходимо извлечь данные из существующих внутренних и внешних источников — создать первоначальную выборку. По итогам ее рассмотрения делается вывод, какие задачи анализа и поддержки принятия решений могут быть решены в пространстве полученных данных.

Как правило, даже небольшие компании используют несколько информационных систем для автоматизации различных сфер своей деятельности. Кроме того, некоторые компании используют отдельные системы в филиалах и региональных офисах. Данные, получаемые от различных структурных элементов компании, оказываются представленными в разном виде, могут содержать противоречия, и показатели, используемые для анализа и принятия решений не могут быть из них получены непосредственно. На стадии сбора и интеграции данных происходит их объединение, приведение к общему представлению и другие преобразования.

Далее выбирается способ представления данных и методы их проверки на полноту и непротиворечивость. После выполнения указанных этапов строится модель

хранилища данных.

9. Генерация гипотез: интеллектуальный анализ данных

Человеческий мозг не приспособлен для восприятия больших коллекций разнородной информации, тем более для улавливания взаимосвязи элементов в таких коллекциях. Традиционно основной дисциплиной, позволяющей работать с такими объемами информации была статистика, однако она часто работает с усредненными характеристиками выборки, которые могут быть малоинформативными (например, средняя высота дома на улице, состоящей из небоскребов и лачуг). Статистические методы дают хорошие результаты, когда заранее сформулирована некоторая гипотеза, а на имеющихся данных требуется ее проверить. Нашей же целью является генерация гипотез на основе больших объемов данных, то есть обнаружение в них заранее неизвестных взаимосвязей и закономерностей.

Требования к интеллектуальному анализу данных были поставлены временем: в связи с усовершенствованием технологий записи и хранения данных деятельность любого предприятия сопровождается регистрацией данных на разных этапах его работы. Возникает вопрос, что делать с подобными записями? Записанные в хранилище данные занимают огромный объем, являются неструктурированными и разнородными (содержат тексты, качественную, количественную информацию), но результаты их обработки должны быть просты и понятны, а инструменты обработки подобных "сырых" данных достаточно просты в применении. В качестве примеров вопросов, на которые хотелось бы получить ответ после проведения анализа данных, можно сформулировать следующие:

Какие товары предлагать данному покупателю?

Можно ли выдавать кредит клиенту?

В чем причины брака при производстве стиральных машин?

Какие схемы покупок характерны для мошенничества с кредитными картами?

Ответы на подобные вопросы могут содержаться в гигабайтах данных. Целью проведения анализа является построение модели или формулировка набора правил, выявляющих закономерности, разбивающих объекты на классы.

9.1 Задачи интеллектуального анализа данных

Интеллектуальный анализ данных (data mining) — это процесс обнаружения в разнородных данных неочевидных, объективных и практически полезных знаний. Методы интеллектуального анализа данных предназначены именно для обнаружения таких закономерностей, которые не очевидны, то есть их не найдет эксперт, основываясь на своем опыте, однако найденные закономерности должны быть объективными, то есть соответствующими действительности, подтверждающиеся примерами, и полезными для деятельности организации.

Инструменты интеллектуального анализа данных, как правило, рассматриваются как помощник аналитика, но не его замена. Автоматически найденная закономерность требует объяснения причин, по которым она появилась. Закономерностей может оказаться несколько, поэтому поле для аналитической работы человека все равно остается.

К основным классам задач, которые решаются методами интеллектуального

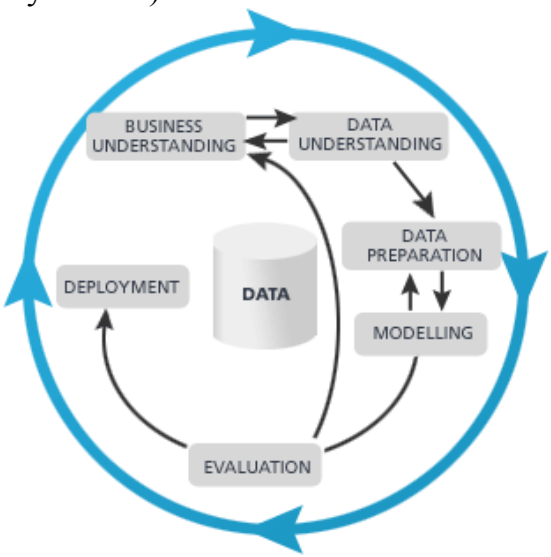
анализа данных, относятся следующие:

- классификация (распределение объектов по заранее известным группам по набору атрибутов)
- кластеризация (разделение объектов на неизвестные заранее группы, например, на основе данных кассы магазина, выделить группы необычных схем покупок)
- выявление отклонений (проверить транзакции, совершенные по кредитным картам за последний месяц, выделить те, которые не вписываются в общую схему)
- анализ связей между событиями (например, обнаружить связь между ремонтом автомобилей и поставкой деталей из определенного региона)

Приведем практический пример приложения, использующего интеллектуальный анализ данных. В 1995 году одна из крупнейших британских торговых сетей Safeway Stores, насчитывающая более 90 000 сотрудников и более 500 магазинов, предложила своим покупателям приложение, разработанное для нее фирмой IBM, формирующее список покупок на персональном электронном устройстве и передающее его в магазин. Оно позволяло покупателю быстро забрать заказанные товары, не разгуливая по торговому залу[12]. Однако вскоре было замечено, что в тех магазинах, где было внедрено использование данного приложения, сократилось количество "случайных" покупок. Чтобы решить данную проблему, на электронных устройствах был введен профиль пользователя, и каждому конкретному пользователю на основе анализа данных о его заказах в прошлом предлагалось приобрести дополнительные к его списку товары. Было отмечено, что в 25% заказов рекомендованные покупки появились.

9.2 Стандарт CRISP-DM

В 1996 году группа аналитиков, представляющих компании DaimlerCrysler, SPPS и NCR, предложила стандарт для включения процесса Data Mining в общую стратегию принятия решений организации или ее подразделения. Стандарт был назван *CRISP-DM* (The Cross-Industry Standart Process for Data Mining). Он предусматривает несколько этапов решения задачи интеллектуального анализа данных. Полученная на одном этапе информация служит входными данными для следующего этапа (см. рисунок 9.1).



12 Пресс-релиз о внедрении http://www.prnewswire.co.uk/news-releases/safeway-and-ibm-launch-world-is-first-in-grocery-shopping-technology-156752995.html

Рис. 9.1 Основные фазы процесса анализа данных по стандарту CRISP-DM

В соответствии со стандартом CRISP-DM, анализ данных является непрерывным процессом со многими циклами и обратными связями. Рассмотрим последовательно все этапы процесса анализа данных на примере некоторой абстрактной компании СТ, занимающейся производством и продажей бытовой техники.

- *Осмысление бизнеса* (Business understanding).

В рамках этой фазы нужно обозначить цели и требования проекта, сформулировать из них задачу анализа данных и подготовить предварительную стратегию решения задачи.

Руководство компании СТ хочет уменьшить расходы, связанные с гарантийными претензиями клиентов. После разговора с инженерами службы контроля качества были сформулированы три конкретных вопроса:

1. Есть ли связь между гарантийными претензиями?
2. Есть ли связь между типами поломок и сервисными центрами?
3. Можно ли составить прогноз о наличии будущих претензий по качеству техники?

Планируется применить методы интеллектуального анализа данных, чтобы раскрыть эти и, возможно, дополнительные закономерности.

- *Осмысление данных* (Data understanding).

В рамках данного этапа собираются данные для обработки, оценивается их качество, анализируется структура данных.

В компании СТ имеется база данных, содержащая информацию о всех приборах, сделанных за последние 3 года. В ней содержатся записи о том, где был произведен бытовой прибор, и гарантийные жалобы из сервисных центров, в которые обращались клиенты. Выделено несколько типов поломок, каждая имеет свой идентификатор.

- *Подготовка данных* (Data preparation).

На данном этапе выделяется тот набор данных, который будет применен в последующих фазах. Выбирается подходящий для анализа формат данных и выполняется приведение к такому формату.

Специалисты из аналитического отдела СТ сделали вывод, что в базе информация хранится в виде, не пригодном для анализа. База данных была оптимизирована под выполнение внутренних запросов, а для выяснения, например, "количества дней, прошедших от даты продажи до первого обращения в сервисный центр" требовалось делать сложные запросы. Более того, атрибут "дата" был записан в разных форматах, поэтому потребовалась дополнительная подготовка данных.

- *Моделирование* (Modeling).

На этапе моделирования выбирается конкретный метод анализа данных и подбираются параметры метода. Для его наилучшего применения может потребоваться дополнительная подготовка данных (тогда возвращаемся к предыдущему этапу). Примеры методов анализа будут рассмотрены в разделе 9.3.

- *Оценка результатов* (Evaluation).

На данном этапе оценивается качество и эффективность методов предыдущей фазы, проверяется, что с помощью методов возможно решение задач, поставленных на первоначальном этапе. В зависимости от оценки либо устанавливаются границы исследования, либо (если результат признан неуспешным) происходит возврат к первому этапу.

В результате моделирования в компании СТ обратили внимание, что верность правила

"Если сервисный центр X, то проблема с нагревательным элементом стиральной машины" сильно менялась в зависимости от центра. Было решено, что последующие исследования должны выявить причины таких несоответствий. Аналитики по итогам рассмотрения признали, что не было получено интересных результатов, в связи с чем было принято решение вернуться к первому этапу. Результатом проведенной работы стали рекомендации по улучшению структуры базы данных компании, которое позволило бы ее эффективнее использовать для анализа.

- *Внедрение* (Deployment).

На данном этапе предложенный ранее метод применяется к большим объемам данных, и по итогам применения составляется отчет. Если процесс выполнялся для сторонней компании, то этап внедрения может выполнять непосредственно сама компания-заказчик.

Из рассмотренного примера можно сделать вывод, что процесс анализа данных является непростым, и на каждом этапе требуется участие в нем специалиста-аналитика, чтобы был получен правильный ответ именно на тот вопрос, который поставлен. Гарантии удачного завершения нет, но при проведении анализа грамотными специалистами, понимающими используемые методы, требования к данным и цели проекта, результат будет полезным для организации.

9.3 Методы анализа данных

Среди методов анализа данных можно выделить три группы: статистические методы, индуктивные алгоритмы и использование нейронных сетей.

Исторически *статистические алгоритмы* появились и использовались еще задолго до появления термина Data Mining. Наиболее популярным статистическим методом анализа является *линейная регрессия*. Задача метода состоит в подборе линейной функции, наиболее соответствующей конкретному набору данных. В данной модели одна из переменных (y) определяется как зависимая от одной или нескольких других переменных (факторов) с линейной функцией зависимости. Регрессионная модель

$$y = f(x,b) + \varepsilon$$

где b – параметры модели, называется линейной регрессией, если функция регрессии $f(x,b)$ имеет вид

$$f(x,b) = b_0 + b_1x_1 + b_2x_2 + \ldots + b_kx_k$$

где b_j – коэффициенты регрессии, x_j – факторы модели, k – количество факторов модели, ε - случайная ошибка модели, предполагается, что она является нормально распределенной с нулевым математическим ожиданием и фиксированной дисперсией, которая не зависит от факторов x_i и переменной y. Модельная линейная функция может быть использована для предсказания поведения зависимой переменной для других наборов значений факторов.

Вторую группу методов составляют *индуктивные алгоритмы*. Они основаны на построении деревьев решений (см. стр. 17). В нелистовых вершинах такого дерева находятся факторы, влияющие на принятие решения. Каждый из них принимает конечное число значений. Каждой ветви перехода к дочерней вершине одно из таких значений приписывается. Если же значения какого-либо фактора не являются дискретными либо их количество бесконечно, то происходит разбиение области значений на конечное число отрезков (подмножеств): например, отрезок значений температуры воздуха ниже -10 градусов назовем "очень холодно", от -10 до +5 –

"холодно", от +5 до +15 градусов – "прохладно" и т. д. Таким образом, фактор "температура воздуха" будет принимать конечное множество (новых) значений.

Приведем алгоритм построения дерева решений для данных, описывающих принятие решения об игре в гольф в зависимости от разных погодных факторов, хранящихся в виде таблицы

номер	облачность	температура	влажность	Играть ли в гольф?
1	солнечно	тепло	небольшая	да
2	пасмурно	холодно	высокая	нет
3	дождь	тепло	высокая	нет
4	солнечно	жарко	высокая	нет

Таблица 9.1 Данные для принятия решения об игре в гольф

В правом столбце таблицы записаны интересующие нас решения, они будут появляться в листовых вершинах дерева. Номер, указанный в левом столбце, не является атрибутом, влияющим на принятие решения. Поскольку в примере не задана важность (порядок рассмотрения) факторов, то при построении дерева будем их использовать слева направо: в корень дерева поместим фактор "облачность" (см. рисунок 9.2). У корня будет три дочерние вершины, переход в которые осуществляется по значениям "солнечно", "пасмурно" и "дождь" соответственно.

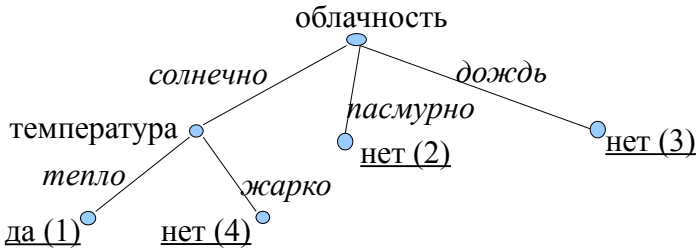


Рис. 9.2 Дерево решений

Далее, если все строки данных при соответствующем атрибуте имеют одинаковое решение (значение правого столбца), то вершина объявляется листовой и решение становится ее значением. Таковыми являются ветви "пасмурно" и "дождь". В листовых вершинах запишем решение и номер соответствующей строки. При значении "солнечно" мы имеем две строки с различными решениями. Для этих строк находится первый атрибут, значения которого у них будут различны (это атрибут "температура"), он становится следующей нелистовой вершиной дерева. Для построенной вершины рассматриваем оставшиеся два значения ("тепло" и "жарко"), ведущие в листовые вершины, соответствующие разным решениям. Поскольку все решения из таблицы оказались рассмотренными, построение дерева завершено.

Структура дерева зависит от порядка рассмотрения факторов. Можно заметить, что при рассмотрении атрибутов в другом порядке, мы получили бы дерево меньшего размера (см. рисунок 9.3).

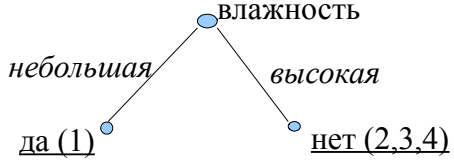


Рис. 9.3 Минимальное дерево решений

Существует общее правило выбора атрибута: он должен разбивать исходное множество данных таким образом, чтобы объекты подмножеств, полученных в результате разбиения, являлись представителями одного класса или были максимально приближены к такому разбиению.

На практике при построении дерева возникает еще одна проблема: работа с огромными объемами данных может порождать огромные деревья, что сделает их практически бесполезными. Поэтому при построении дерева решается вопрос об отсечении некоторых его ветвей. Ограничение может быть задано на глубину дерева, на количество ветвей, выходящих из вершин. Данные ограничения могут приводить к снижению точности классификации.

Метод деревьев решений имеет ряд преимуществ: он позволяет решать задачи, в которых отсутствует априорная информация о виде зависимости между исследуемыми данными, классифицировать объекты в областях, где сложно формализовать знания, а точность моделей оказывается достаточно высокой. Кроме того, при использовании данного метода человек понимает процедуру принятия решения.

Третьим подходом к анализу данных является использование *нейронных сетей*. Идея данного подхода появилась из биологии: для решения задач предложено смоделировать не алгоритмы, которые выполняет человек, а саму структуру мозга, представляющую собой сеть из элементов – нейронов, и организовать взаимодействие элементов этой сети. Основными преимуществами нейронных сетей являются способность к обучению при предоставленном достаточном количестве примеров, и способность выделять кластеры связанной между собой информации. Эти возможности являются полезными для анализа данных, потому что для человека обнаружение связей в огромных объемах данных представляется нереальной задачей, особенно когда такие связи являются неочевидными.

Нейронная сеть состоит из простых элементов – нейронов (см. рисунок 9.4). У каждого нейрона есть набор входов, на которые он получает сигнал либо от внешней среды, либо от других нейронов. Сумма значений входных параметров $(x_1, \ldots, x_n)$ с некоторыми весами $(w_1,\ldots, w_n)$ складываются, и значение суммы с весовыми коэффициентами подается на вход специальной функции, которая называется функцией активации. Когда поданное ей значение больше, чем некоторое пороговое, нейрон "активируется" — передает значение следующему нейрону, который работает по такому же принципу. В качестве примера такого устройства можно привести сенсорный элемент, который вырабатывает сигнал от воздействия какого-либо вида энергии (света, звука, давления, нагревания). Если входной сигнал превышает некоторый порог, то на выходе получаем 1, иначе 0.

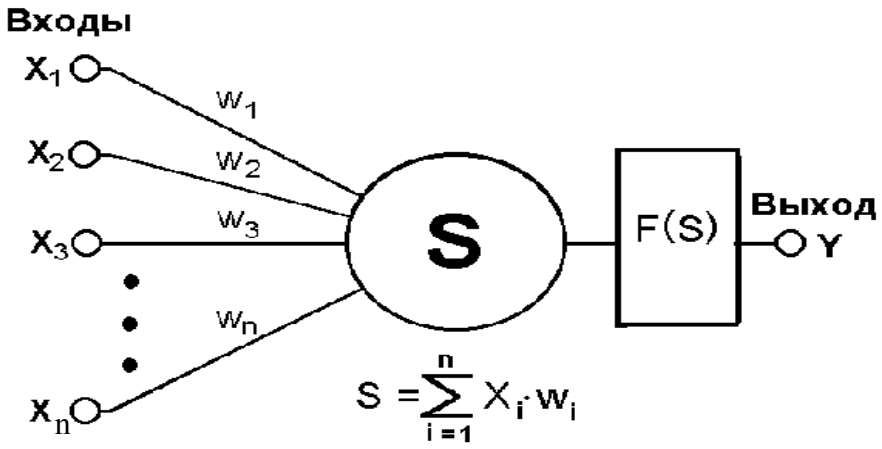
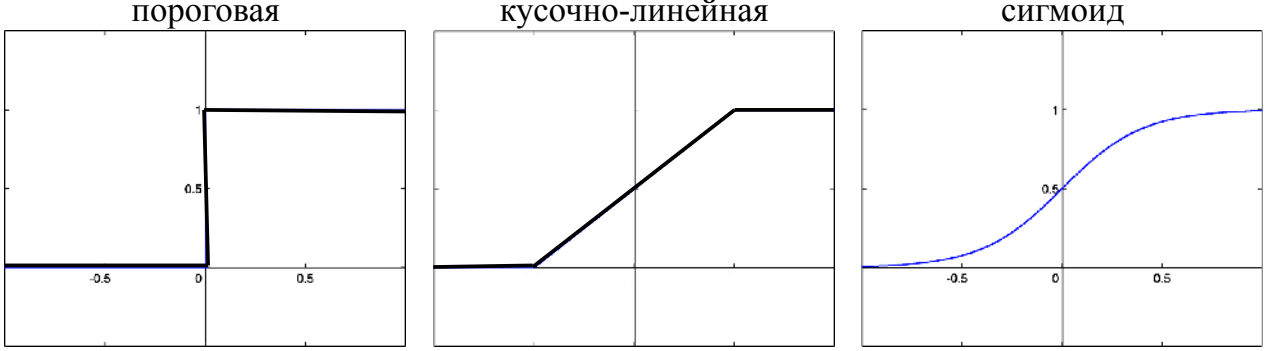


Рис. 9.4 Структура нейрона

Как правило, выходом нейрона является число от 0 до 1.0, чтобы при передаче следующему нейрону не оказалось непропорционально большого значения на каком-то из входов. В качестве функции активации нейрона наиболее часто используется один из трех видов функций:

пороговая кусочно-линейная сигмоид



Отдельно взятый нейрон очень простой элемент и сам по себе много задач не решает. Работать же для практически полезных задач он начинает, будучи объединенным с другими нейронами в сеть. Сети бывают одноуровневыми (см. структуру сети на рисунке 9.5) и многоуровневыми (пример двухуровневой сети изображен на рисунке 9.6).

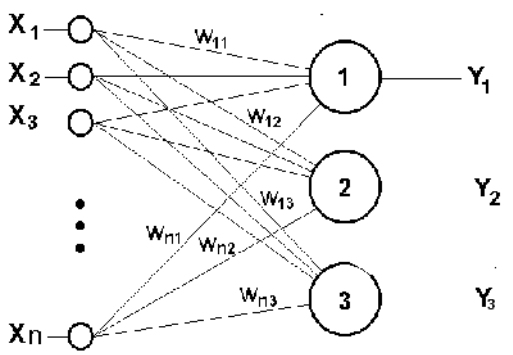


Рис.9.5 Одноуровневая сеть

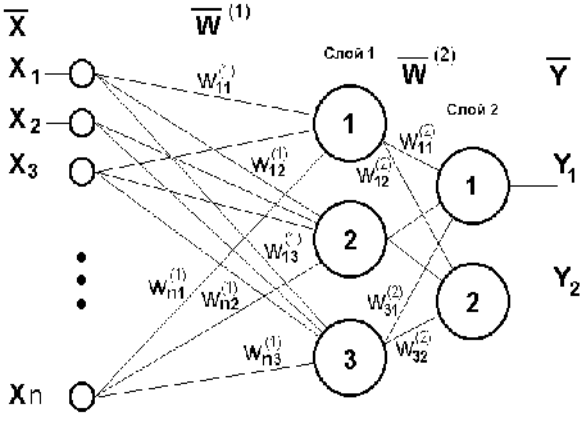


Рис. 9.6 Двухуровневая сеть

Сеть бывает полностью связанной, когда все узлы одного уровня связаны со всеми узлами следующего уровня, и слабо связанной, когда таких связней между нейронами разных уровней меньше. Количество нейронов, количество уровней определяется разработчиком сети в зависимости от задачи. Чем больше в сети количество скрытых уровней, тем более сложные закономерности сеть потенциально будет способна выявить. Однако если уровней становится слишком много, возникает эффект "перетренированности" сети, когда она делает лишние неверные обобщения и не корректирует свое поведение во время обучения.
На решение конкретной задачи влияет структура сети, заданные весовые коэффициенты и пороговые значения функций активации. Структура и функции активации обычно задаются до решения задачи, а вот весовые коэффициенты подбираются в процессе обучения сети.

Для нейронных сетей не пишутся программы в традиционном смысле слова. Обучение сети заключается в подборе весов (w_i) — коэффициентов связи между нейронами. В процессе обучения сеть выявляет зависимости между входными и выходными данными, а также выполняет обобщение. В случае успешного обучения сеть сможет работать и на неполных, зашумленных, частично искаженных данных. Обучение должно проводиться на подходящей выборке примеров. Известна история, когда нейронную сеть обучали распознавать образы танков по фотографиям, однако позднее выяснилось, что все предъявляемые ей изображения имели один и тот же фон, и именно этот тип ландшафта сеть и научилась распознавать. Существуют два разных подхода к обучению нейронных сетей — это обучение с учителем (supervised learning, управляемое обучение) и обучение без учителя (unsupervised learning, неуправляемое обучение).

При обучении с учителем нейронной сети нужно задать множество примеров с ожидаемым результатом работы. Рассмотрим, например, задачу классификации объектов "круг" и "квадрат". Для начала случайным образом выбираются пороговые значения для функций активации, устанавливаем связи между элементами. Начальные весовые коэффициенты полагаем равными 0. Далее сети последовательно подаются на вход объекты "круг" и "квадрат" из обучающей выборки. При появлении квадрата некоторые из нейронов активизируются, ставим им веса `1`. Коэффициенты для нейронов, которые активизируются при появлении круга, устанавливаем равными `-1`. Процесс корректировки коэффициентов продолжается до тех пор, пока не удается стабилизировать ошибку.

В рассмотренном примере для заданной обучающей выборки мы заранее знали, что в ней два класса объектов. Однако подобная информация не всегда является известной: есть задачи, в которых требуется найти заранее неизвестные зависимости. Например, подобная задача возникает при разведочном анализе данных. Целью в этом случае является понимание структуры данных, для того, чтобы потом их анализировать с учетом найденной структуры, в зависимости от нее выбрать нейросетевую модель. К задачам такого рода относится и поиск сходства между различными классами данных. Примером сети, которая способна обучаться без учителя (без заранее заданных примеров и результатов работы), является сеть Кохонена. В данной сети имеется два слоя нейронов – входной и выходной. Каждый объект, который требуется классифицировать, представляется в виде некоторого вектора, подающегося на вход нейронной сети. Входные элементы предназначены

только для того, чтобы распределять данные входного вектора между выходными элементами. Количество нейронов во входном слое определяется количеством компонент входного вектора. Каждый входной элемент связан со всеми выходными, и общее число весовых значений, влияющих на выходной элемент, тоже оказывается равным размерности входных векторов. Часто удобно интерпретировать весовые значения выходных элементов как значения координат, описывающих позицию вектора в пространстве выходных данных.

Обучается сеть Кохонена методом последовательных приближений. Сначала выходные элементы располагаются случайным образом. Последовательно на вход сети подаются входные векторы. Проход по всем векторам называется эпохой обучения. Когда в процессе обучения подается новый входной вектор, то тот их выходных элементов, который оказался ближе всего к нему, объявляется победителем. Для элемента-победителя производится корректировка весов так, чтобы он стал еще ближе к учебному вектору. Также корректируются и веса выходных элементов, находящихся близко к элементу-победителю. Обычно весовые значения элемента нужно обновлять, если он лежит внутри круга заданного радиуса с центром в элементе-победителе. В ходе обучения радиус постепенно уменьшается.

Выходной слой еще называется слоем топологической карты, элементы которой располагаются в двумерном пространстве. Задачу можно рассматривать как отображение n-мерного пространства входных векторов в двумерное пространство выходов. Идея сети Кохонена возникла по аналогии с некоторыми известными свойствами человеческого мозга. Кора головного мозга представляет собой большой плоский лист, свернутый складками, с известными топологическими свойствами: например, участок, ответственный за кисть руки, примыкает к участку, ответственному за движения всей руки, и таким образом все изображение человеческого тела непрерывно отображается на эту двумерную поверхность.

В настоящее время существует специальные пакеты программ, позволяющие моделировать нейронную сеть, проводить ее обучение и использовать для решения задач, например: ST Neural Networks, Matlab Toolbox, Nestor, Cascade Correlation, Neudisk, Mimenice, Nu Web, Brain, Dana, Neuralworks Professional II Plus, Brain Maker, HNet, Neuro Solutions.

Применении технологий интеллектуального анализа данных имеет ряд ограничений. Во-первых, инструменты для анализа данных не являются заменой аналитику-человеку, хотя при правильном использовании и оказывают ему существенную помощь. Во-вторых успешный анализ требует качественной предварительной обработки данных: по утверждению аналитиков и пользователей баз данных этот этап может занимать до 80% времени всего процесса. Также известно, что с помощью анализа данных возможно получение большого процента ложных, недостоверных или бессмысленных результатов. Чтобы этого избежать необходима проверка адекватности выбранных моделей на тестовых данных. Ряд вопросов возникает также с возможностью сохранения данных и их автоматизированной обработки, если речь идет о персональных данных клиентов или сотрудников организации, поскольку использование таких данных регулируется законодательно. Но несмотря на все сложности применения технологии, она является очень востребованной и перспективной для разных организаций.

Задания к главе 9

9.1 Построить минимальное дерево решений для диагностики неисправностей в автомобиле, используя данные из таблицы:

свет	звук	зажигание	топливный бак	запах	ПРОБЛЕМА
слабый	рев	работает	непустой	обычный	**аккумулятор**
в норме	хрип	нет	непустой	обычный	**стартер**
в норме	щелчки	нет	непустой	обычный	**соленоид**
в норме	нормальный	работает	пустой	обычный	**нет топлива**
в норме	нормальный	работает	непустой	бензин	**утечка топлива**

9.2 Построить на основе приведенной ниже таблицы дерево классификации южных деревьев. Есть ли в таблице информация, которая для классификации не является существенной?

листья	размер листьев	плод	древесина	сбрасывает листья?	тип дерева
широкие	большие	коричневые шары	тяжелая	да	**платан**
иголки	короткие	шишка	хорошая	нет	**сосна**
тонкие	длинные	орех	волокнистая	нет	**кокос**
простые	маленькие	желудь	тяжелая	нет	**дуб виргинский**
дольчатые	большие	крылатка	тяжелая	да	**клен**

Заключение

На сегодняшний день область называемая "Управление знаниями" является активно развивающейся. Особенную позицию в управлении знаниями занимают методы представления знаний, их автоматизированной обработки и анализа, помогающие сделать компьютеры интеллектуальными помощниками людей в современных бизнес-организициях. Информационные технологии не решают всех проблем управления знаниями. Напротив, они могут породить и дополнительные сложности при разработке и внедрении методов и систем. Однако при умелом использовании, программные средства управления знаниями дают доступ к новому мощному ресурсу — инфраструктуре знаний, способному перевести работу организации на качественно новый уровень.

Список литературы и Интернет-источников

1. A. Aamodt, E. Plaza Case-Based Reasoning: Foundational Issues, Methodological

Variations, and System Approaches. AI Communications, 7(I), 1994: pp. 39-59

2. K. D. Ashley "Modeling Legal Arguments. Reasoning with Cases and Hypotheticals" – Cambridge, Massachusetts, The MIT Press, 1991
3. E.Awad, H. Ghaziri "Knowledge Management" – Prentice Hall, 2004
4. I. Becerra-Fernandez, A. Gonzalez, R. Sabherwal "Knowledge Management: Challenges, Solutions, and Technologies" – Prentice Hall, 2004
5. R.J. Brachman, H.J. Levesque "Knowledge Representation and Reasoning" — Elsevier & Morgan Kaufmann, 2004
6. S. Dutta "Knowledge Processing and Applied Artificial Intelligence" – Butterworth-Heinemann, 1993
7. K. Hammond "A model of case-based planning", in Proceedings of the 5th National Conference on Artificial Intelligence, Menlo Park, Calif.: American Association for Artificial Intelligence, 1986, pp. 65-95
8. P. Harmon, D. King "Expert Systems. Artificial Intelligence in Business" – John Willey & Sons, Inc., 1985
9. И. Братко "Программирование на языке Пролог для искусственного интеллекта": Пер. с англ. – М.: Мир, 1990
10. К.Дж. Дейт "Введение в системы баз данных", 6 издание: Пер. с англ. – К.; М.; СПб.: Издательский дом "Вильямс", 1999
11. Д. Джарратано, Г. Райли "Экспертные системы: принципы разработки и программирование": Пер. с англ. — М.: "Вильямс", 2007
12. П. Джексон "Введение в экспертные системы" — М: Издательский дом "Вильямс", 2001
13. Р. Каллан "Основные концепции нейронных сетей": Пер. с англ. — М: Издательский дом "Вильямс", 2001
14. Л. Е. Карпов, В. Н. Юдин "Методы добычи данных при построении локальной метрики в системах вывода по прецедентам" — Препринт ИСП РАН, Москва, 2007 г. http://citforum.ru/consulting/BI/data_mining/
15. С.В. Козлов, А.Ф. Тузовский, С.В. Чириков, В.З. Ямпольский "Использование онтологий в системах управления знаниями организаций" – Известия Томского политехнического университета, Т. 309, № 3, 2006
16. Ж.Л. Лорьер "Системы искусственного интеллекта": Пер. с франц. — М.: Мир, 1991
17. Д.И. Муромцев. "Онтологический инжиниринг знаний в системе Protégé". – СПб: СПб ГУ ИТМО, 2007
18. С. Рассел, П. Норвиг "Искусственный интеллект: современный подход", 2-е изд.: Пер. с англ. – М.: Издательский дом "Вильямс", 2006
19. П. Уинстон "Искусственный интеллект":Пер. с англ. — М.: Мир, 1980
20. Д. Уотермен "Руководство по экспертным системам": Пер. с англ. — М.: Мир, 1989
21. А.П. Частиков, Т.А. Гаврилова, Д.Л. Белов «Разработка экспертных систем.

Среда CLIPS» — СПб.: «БХВ-Петербург», 2003.

22. Ч. Чень, Р. Ли "Математическая логика и автоматическое доказательство теорем": Пер. с англ. – М.: Наука, Главная редакция физико-математической литературы, 1983.
23. Среда CLIPS: http://clipsrules.sourceforge.net
24. Ресурс, посвященный машинному обучению, распознаванию образов и интеллектуальному анализу данных http://www.machinelearning.ru
25. Web-страница системы Protégé: http://protege.stanford.edu/
26. Swi-prolog: http://www.swi-prolog.org

Содержание