Долженко А.И.

Управления знаниями

Учебное пособие

Для магистров, обучающихся по направлению подготовки09.04.03
«Прикладная информатика»

Ростов-на=Дону
2015

Оглавление

Тема 1. Основные классы интеллектуальных информационных систем

Прогресс в сфере экономики немыслим без применения современных информационных технологий, представляющих собой основу экономических информационных систем (ИС). ИС в экономике имеют дело с организацией и эффективной обработкой больших массивов данных в компьютеризированных системах предприятий, обеспечивая информационную поддержку принятия решений менеджерами. Глобализация финансовых рынков, развитие средств электронной коммерции и формирование в Интернете доступных для анализа баз данных финансово-экономической информации, снижение стоимости программной реализации ИС привели к беспрецедентному росту их использования в экономике. ИС позволяют объективно оценить достигнутый уровень развития экономики, выявить резервы и обеспечить успех их деятельности на основе применения правильных решений.

Интеллектуальные информационные системы (ИИС) — естественный результат развития обычных информационных систем, сосредоточили в себе наиболее наукоемкие технологии с высоким уровнем автоматизации не только процессов подготовки информации для принятия решений, но и самих процессов выработки вариантов решений, опирающихся на полученные информационной системой данные. ИИС способны диагностировать состояние предприятия, оказывать помощь в антикризисном управлении, обеспечивать выбор оптимальных решений по стратегии развития предприятия и его инвестиционной деятельности. Благодаря наличию средств естественно-языкового интерфейса появляется возможность непосредственного применения ИИС бизнес-пользователем в качестве средств поддержки процессов анализа, оценки и принятия экономических решений. ИИС применяются для экономического анализа деятельности

предприятия, стратегического планирования, инвестиционного анализа, оценки рисков и формирования портфеля ценных бумаг, финансового анализа, маркетинга и т.д.

Современная динамично изменяющаяся бизнес-среда требует профессионалов, способных в дополнение к экономическим знаниям применять современные информационные технологии, чтобы находить инновативные способы реализации бизнес-процессов.

Работы в области искусственного интеллекта в течение довольно длительного времени представлялись многим как причуды оторванных от реальности информатиков-интеллектуалов, обучающих компьютер игре в шахматы или распознаванию сцен, или же пытающихся создать автономно ориентирующиеся в пространстве мобильные роботы. Механизмы, лежащие в основе таких программ и систем, объявлялись неалгоритмизуемыми, эвристическими, считались известными только посвященным, зачастую несли в себе аромат таинственности и волшебства.

Появление экспертных систем MYCIN, DENDRAL, PROSPECTOR, а также обнадеживающие результаты их успешного применения в области медицины, технической диагностики, геофизики, управления непрерывными технологическими процессами решительно изменили ситуацию. Стало очевидным, что методы правдоподобных и дедуктивных выводов могут быть хорошим дополнением или частичной заменой специалиста, ставящего медицинский или технический диагноз и вообще принимающего решения в форме выбора одной из альтернативных гипотез на основании наблюдаемых данных.

Эти успехи стимулировали применение технологий и методов искусственного интеллекта в самых разных отраслях экономики, в первую очередь, для анализа и диагностирования эффективности экономической деятельности предприятий, выбора эффективной стратегии поведения трейдера на рынке ценных бумаг, выбора оптимальных вариантов инвестиционных проектов в условиях неопределенности и при наличии

трудно формализуемых факторов. Первые экспертные системы были оторваны от корпоративных информационных систем и строились как самостоятельные программы, имели собственную организацию хранения данных и знаний. Поэтому их применение к реальным проблемам в сфере экономики первоначально не дало ожидаемого результата. Возникли проблемы, связанные с высокой трудоемкостью создания и реорганизации базы знаний традиционными методами интервьюирования экспертов, а также с загрузкой, хранением и актуализацией больших объемов данных, на которые не были рассчитаны эти экспертные системы.

Новая волна и значительный эффект от применения технологии искусственного интеллекта получены в результате разработки и применения интеллектуальных информационных систем, явившихся синтезом экспертных и информационных систем. Создание ИИС стало естественным продолжением широкого применения информационных систем классического типа. Системы типа BPR (реинжиниринга бизнес-процессов) показали возможность упорядочения информационных потоков и совершенствования структуры предприятия при внедрении информационных технологий, помогли освоить методологию разработки информационной модели предприятия. Интегрированные ИС предприятия обеспечивают информационную поддержку всех производственных процессов и служб предприятия, включая проектирование, изготовление и сбыт продукции, финансово-экономический анализ, планирование, управление персоналом, маркетинг, сопровождение эксплуатации изделий, перспективное планирование. Внедрение информационных систем типа ERP (Enterprise Resource Planing) увеличивает эффективность работы предприятия на 20-30%. В результате появились полностью компьютеризованные информационно-технологические связи между корпорациями (системы B2B или (бизнес-бизнес) и связи корпорации с клиентами (системы B2C или системы (бизнес-клиент).

Успехи повсеместного внедрения интегрированных информационных систем с встроенными жесткими алгоритмами бухгалтерских расчетов, финансового анализа, контроля исполнения документов и графиков производства, контроля исполнения заявок и управления запасами стимулировали интерес информационно-аналитических служб и отделов перспективного планирования и развития к возможностям информационных систем.

В этом случае потребовался значительно больший объем информации как собственно о предприятии, так и о его окружении, т.е. природных, политических, экономических и других факторах, конкурентах, поставщиках и т.д., а также значительно более сложные вычисления, необходимость учета слабо формализуемых факторов, высокий уровень интерфейса. Поставленные задачи реализованы в системах поддержки принятия решений (DSS). Их отличительная черта — значительно более высокий уровень «интеллекта», чем у обычных интегрированных систем управления производством; наличие специальных процедур для отбора и ввода данных, в том числе и по расписанию из различных внешних систем. В DSS производится заблаговременное вычисление (в целях обеспечения уменьшения времени реакции) агрегированных данных, часто используемых в запросах; используется специальная организация хранения данных, обеспечивающая возможность многоаспектного поиска с изменяемой глубиной агрегирования/дезагрегирования данных. Эта технология получила название хранилищ и витрин данных в сочетании с OLAP (оперативная аналитическая обработка данных).

Наиболее мощные фирмы, разрабатывающие системы управления базами данных (СУБД) — ORACLE, SYBASE, Microsoft, — поставляют на рынок системы, в которые DSS входят как компонента. В состав таких систем входят технологии искусственного интеллекта — нейронные сети, интеллектуальный анализ данных. Объектно-ориентированная структура этих баз данных сделала реальностью идеологию фреймов, разработанную в

рамках искусственного интеллекта. Технические решения, необходимые для создания полномасштабных интеллектуальных информационных систем — средства ведения баз знаний на основе объектно-ориентированных баз данных, автоматизации формирования баз знаний на основе методов интеллектуального анализа данных, полнотекстовые системы поиска и семантические анализаторы естественного языка для естественно-языкового интерфейса — стали производиться как серийно выпускаемые программные изделия. Нереализованными в рамках таких СУБД пока остаются технологии реализации правдоподобных (вероятностных) и логических (дедуктивных) выводов.

Наибольший эффект от внедрения ИИС достигается там, где при принятии решений учитываются наряду с экономическими показателями слабо формализуемые факторы — экономические, политические, социальные. Так, в области экономического анализа и управления, менеджмента, антикризисного управления, стратегического планирования, инновационного менеджмента и инвестиционного анализа существует обширное поле деятельности для применения интеллектуальных технологий и систем. Поскольку в основе этих видов деятельности лежит проблема выбора решений.

Интеллектуальные информационные системы (ИИС). Современные базы данных включают в свой состав целый ряд механизмов и технологий, повышающих их интеллектуальные возможности. Это относится прежде всего к многомерной организации данных в хранилищах данных, организации естественноязыкового интерфейса на ограниченном фрагменте языка, реализации сценариев «что если». Все эти механизмы почерпнуты из области исследований по искусственному интеллекту. Системы поддержки принятия решений — квазиинтеллектуальные системы, поскольку они призваны автоматизировать не сам процесс оценки предпочтительности гипотез или выбора варианта решения, а только готовят аналитические

обобщенные данные для окончательного выбора решения специалистом-менеджером.

Важность этих систем для теории и практики и практики искусственного интеллекта определяется двумя обстоятельствами:

- в DSS реализуется поиск аналитических зависимостей или агрегатов, при использовании которых правила принятия решений, т.е. зависимости между наблюдаемыми данными и гипотезами становятся более простыми;
- в структуре специализированных процессоров или архитектур этих систем реализуются некоторые начальные этапы технологии обработки данных, характерных для технологии искусственного интеллекта. Это относится к организации хранения и обработки больших объемов данных в виде многомерных кубов с учетом семантических взаимосвязей.

На начальных стадиях разработки методов искусственного интеллекта создание экспертных систем (ЭС), предназначенных для оценки предпочтительности гипотез на основе наблюдаемых данных, и документальных информационно-поисковых систем (ИПС) и систем управления базами данных (СУБД) шло параллельными путями, затем произошло объединение ИПС как компоненты полнотекстового поиска и реляционных СУБД. В настоящее время постреляционные СУБД включают в себя ряд компонентов из области искусственного интеллекта. Экспертные системы представляли собою автономные программные комплексы не интегрированные с базами данных и системами аналитических вычислений. Понятие «экспертная система» закрепилось за такими автономными программами, ориентированными на определенную достаточно узкую сферу применения. Затем стало ясно, что принципы логического вывода и баз знаний применимы к широкому кругу задач, экспертные системы стали использовать базы данных и оформляться как программные продукты, имеющие развитые средства ввода-вывода данных, хранения и ведения баз

знаний, прикладные задачи и такую новую генерацию систем стали называть интеллектуальные информационные системы.

ИИС объединяют в себе возможности СУБД, лежащих в основе ИС, и технологию искусственного интеллекта, благодаря чему хранение в них экономической информации сочетается с ее обработкой и подготовкой для использования при принятии решений. Вначале ИИС, называемые также системами, основанными на знаниях, рассматривались как средство, позволяющее не экспертам принимать решения с таким же качеством, как один или более экспертов в конкретной области. Однако очень быстро стало ясно, что эта технология в действительности способна к достижению большего объема знаний и более быстрого реагирования, чем группа специалистов.

Первоначально ИИС использовали знания нескольких экспертов в каждой из областей инвестиций. В настоящее время базы знаний частично формируются посредством машинного обучения, используя методы индукции, генетические алгоритмы и некоторые другие методы извлечения знаний. Менеджер, используя такую схему, теоретически может принимать решения более эффективно и с меньшей стоимостью, чем это смог бы сделать любой индивидуальный эксперт в данной области. Наиболее очевидным преимуществом интеграции некоторых форм искусственного интеллекта в процессе принятия решений по сравнению с постоянным консультированием с группой экспертов обычно является более низкая стоимость и большее соответствие результатов задаче. В отличие от обычных аналитических и статистических моделей, ИИС позволяют получить решение трудно формализуемых слабо структурированных задач. Возможность ИИС работать со слабо структурированными данными подразумевает наличие следующих качеств:

– решать задачи, описанные только в терминах мягких моделей, когда зависимости между основными показателями являются не

вполне определенными или даже неизвестными в пределах некоторого класса;

- способность к работе с неопределенными или динамичными данными, изменяющимися в процессе обработки, позволяет использовать ИИС в условиях, когда методы обработки данных могут изменяться и уточняться по мере поступления новых данных;
- способность к развитию системы и извлечению знаний из накопленного опыта конкретных ситуаций увеличивает мобильность и гибкость системы, позволяя ей быстро осваивать новые области применения.

Возможность использования информации, которая явно не хранится, а выводится из имеющихся в базе данных, позволяет уменьшить объемы хранимой фактуальной информации при сохранении богатства доступной пользователю информации. Направленность ИИС на решение слабо структурированных, плохо формализуемых задач расширяет область применения ИИС.

Наличие развитых коммуникативных способностей у ИИС дает возможность пользователю выдавать задания системе и получать от нее обработанные данные и комментарии на языке, близком к естественному. Система естественно-языкового интерфейса (СЕЯИ) транслирует естественноязыковые структуры на внутримашинный уровень представления знаний. Включает морфологический, синтаксический, семантический анализ и соответственно в обратном порядке синтез. Программа интеллектуального интерфейса воспринимает сообщения пользователя и преобразует их в форму представления базы знаний и, наоборот, переводит внутреннее представление результата обработки в формат пользователя и выдает сообщение на требуемый носитель.

Важнейшее требование к организации диалога пользователя с ИИС — естественность, означающая формулирование потребностей пользователя с

использованием профессиональных терминов конкретной области применения. Наибольшее распространение ИИС получили для экономического анализа деятельности предприятия, стратегического планирования, инвестиционного анализа, оценки рисков и формирования портфеля ценных бумаг, финансового анализа, маркетинга.

Применение ИИС совместно со стандартными методами исследования операций, динамического программирования, а также с методами нечеткой логики для планирования при комплексной автоматизации деятельности предприятия, приносит принципиальные выгоды:

- реально снижаются операционные издержки;
- повышается качество управленческих решений.

Детальное обоснование политики предприятия в решении задач расширения и модернизации производства, диверсификации деятельности, повышении эффективности использования производственных мощностей позволяет формировать выполнимые и хорошо контролируемые планы работ. Такое обоснование, базирующееся на текущей производственной загрузке, необходимых материальных и человеческих ресурсах, возможно лишь в хорошо структурированных средах с достаточным информационным базисом.

Для ИИС характерны следующие признаки:

- развитые коммуникативные способности: возможность обработки произвольных запросов в диалоге на языке максимально приближенном к естественному (система естественно-языкового интерфейса — СЕЯИ);
- направленность на решение слабоструктурированных, плохо формализуемых задач (реализация мягких моделей);
- способность работать с неопределенными и динамичными данными;
- способность к развитию системы и извлечению знаний из накопленного опыта конкретных ситуаций;

- возможность получения и использования информации, которая явно не хранится, а выводится из имеющихся в базе данных;
- система имеет не только модель предметной области, но и модель самой себя, что позволяет ей определять границы своей компетентности;
- способность к аддуктивным выводам, т.е. к выводам по аналогии;
- способность объяснять свои действия, неудачи пользователя, предупреждать пользователя о некоторых ситуациях, приводящих к нарушению целостности данных.

Традиционно считается, что ИИС содержит базу данных, базу знаний, интерпретатор правил или машину вывода, компоненту объяснения и естественно-языкового интерфейса, обеспечивающих связный дискурс, т.е. диалог пользователя и системы с попеременным переходом инициативы.

Отличительные особенности ИИС по сравнению с обычными ИС состоят в следующем:

- интерфейс с пользователем на естественном языке с использованием бизнес-понятий, характерных для предметной области пользователя;
- способность объяснять свои действия и подсказывать пользователю, как правильно ввести экономические показатели и как выбрать подходящие к его задаче параметры экономической модели;
- представление модели экономического объекта и его окружения в виде базы знаний и средств дедуктивных и правдоподобных выводов в сочетании с возможностью работы с неполной или неточной информацией;
- способность автоматического обнаружения закономерностей бизнеса в ранее накопленных фактах и включения их в базу знаний.

ИИС особенно эффективны в применении к слабо структурированным задачам, в которых пока отсутствует строгая формализация, и для решения которых применяются эвристические процедуры, позволяющие в большинстве случаев получить решение. Отчасти этим объясняется то, что диапазон применения ИИС необычайно широк: от управления непрерывными технологическим процессами в реальном времени до оценки последствий от нарушения условий поставки товаров по импорту.

По мере совершенствования принципов логического и правдоподобного вывода, применяемых в ИИС за счет использования нечеткой, модальной, временной логики, байесовских сетей вывода, ИИС начинают проникать в высокоинтеллектуальные области, связанные с разработкой стратегических решений по совершенствованию деятельности предприятий. Этому способствуют более современные алгоритмы анализа и синтеза предложений естественного языка, облегчающие общение пользователя с системой.

Включение в состав ИИС классических экономико-математических моделей, методов линейного, квадратичного и динамического программирования позволяет сочетать анализ объекта на основе экономических показателей с учетом факторов и рисков политических и внеэкономических факторов, оценивать последствия полученных их ИИС решений.

Наличие в составе ИИС объектно-ориентированной базы данных позволяет однородными средствами обеспечить хранение и актуализацию как фактов, так и знаний.

Проектирование ИИС как крупного программного комплекса как в отношении его жизненного цикла, так и в отношении технологии проектирования незначительно отличается от технологии проектирования ИС. Основная специфика связана с разработкой базы знаний.

ИИС можно классифицировать по разным основаниям. Мы выберем в качестве оснований классификации следующие: предметная область в

экономике, степень автономности от корпоративной ИС или базы данных, по способу и оперативности взаимодействия с объектом, адаптивности, модели знаний (рис. 1.1).

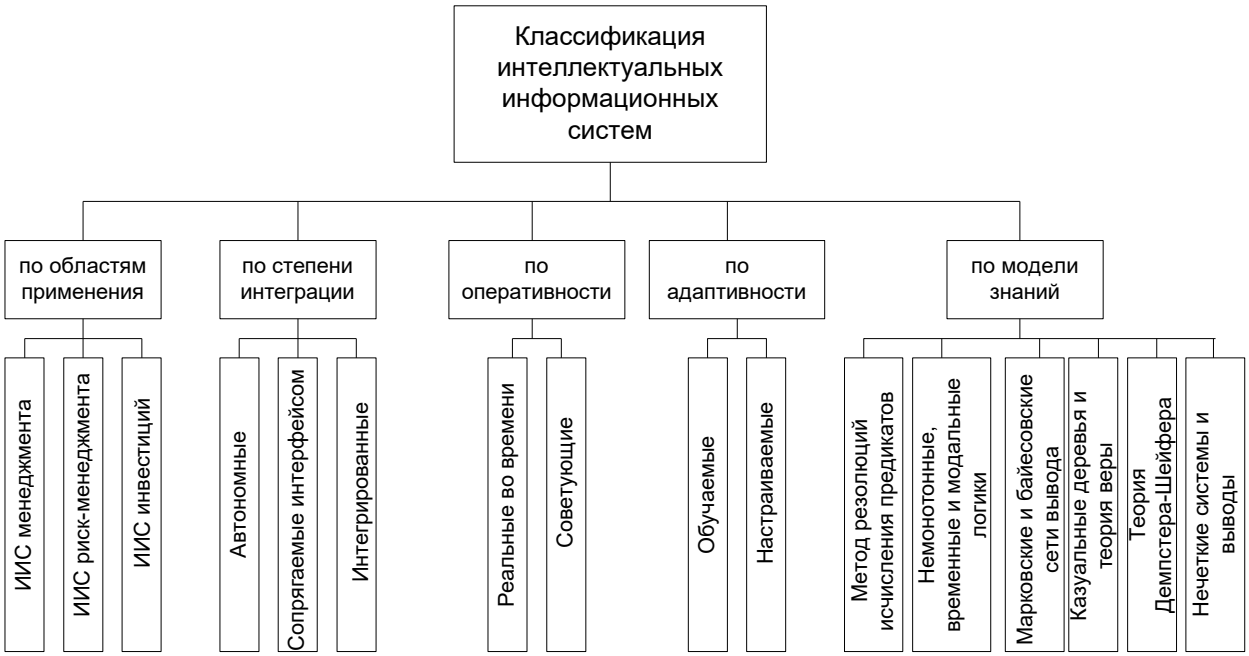


Рис. 1.1. Классификация интеллектуальных информационных систем

На рисунке для примера приведены ИИС из областей менеджмента, риск-менеджмента и инвестиций. По степени интеграции ИИС могут быть: автономные в виде самостоятельных программных продуктов с собственной базой данных; сопрягаемые с корпоративной системой с помощью средств ODBC или OLE DB; полностью интегрированные. По оперативности принято различать динамические и статические ИИС. Однако фактор времени всегда является существенным в ИИС и полностью статических систем не может быть по определению. Предлагается различать ИИС реального времени с собственными сенсорами и эффекторами и советующие, в контур которых вовлечен пользователь.

По адаптивности различаются обучаемые ИИС типа нейронных сетей, т.е. системы, параметры, а возможно и структура которых могут изменяться в процессе обучения или самообучения, и ИИС, параметры которых изменяются администратором базы знаний. Наиболее часто используемые

модели знаний приведены непосредственно на рис. 1.1. Приведем несколько примерен ИИС.

Intelligent Hedger: основанный на знаниях подход в задачах страхования от риска. Фирма: Information System Department, New York University. Проблема огромного количества постоянно растущих альтернатив страхования от рисков, быстрое принятие решений менеджерами по рискам в ускоряющемся потоке информации, а также недостаток соответствующей машинной поддержки на ранних стадиях процесса разработки систем страхования от рисков предполагает обширную сферу различных оптимальных решений для менеджеров по риску. В данной системе разработка страхования от риска сформулирована как многоцелевая оптимизационная задача. Данная задача оптимизации включает несколько сложностей, с которыми существующие технические решения не справляются. Краткие характеристики: система использует объектное представление, охватывающее глубокие знания по управлению риском и облегчает эмуляцию первичных рассуждений, управляющих риском, полезных для выводов и их объяснений.

Система рассуждений в прогнозировании обмена валют. Фирма: Department of Computer Science City Polytechnic University of Hong Kong. Представляет новый подход в прогнозировании обмена валют, основанный на аккумуляции и рассуждениях с поддержкой признаков, присутствующих для фокусирования на наборе гипотез о движении обменных курсов. Представленный в прогнозирующей системе набор признаков — это заданный набор экономических значений и различные наборы изменяющихся во времени параметров, используемых в модели прогнозирования. Краткие характеристики: математическая основа примененного подхода базируется на теории Демпстера—Шейфера.

Nereid: Система поддержки принятия решений для оптимизации работы с валютными опционами. Фирма: NTT Data, The Tokai Bank, Science University of Tokyo. Система облегчает дилерскую поддержку для

оптимального ответа как один из возможных представленных вариантов; более практична и дает лучшие решения, чем обычные системы принятия решений. Краткие характеристики: система разработана с использованием фреймовой системы CLP, которая легко интегрирует финансовую область в приложение ИИ. Предложен смешанный тип оптимизации, сочетающий эвристические знания с техникой линейного программирования. Система работает на Sun-станциях.

PMIDSS: Система поддержки принятия решений при управлении портфелем. Разработчики: Финансовая группа Нью-Йоркского университета. Решаемые задачи: выбор портфеля ценных бумаг; долгосрочное планирование инвестиций. Краткие характеристики: смешанная система представления знаний, использование разнообразных механизмов вывода: логика, направленные семантические сети, фреймы, правила.

Тема 2
Цели разработки интеллектуальных информационных систем

Реализация в экономической модели объекта информатизации позволяет строить классическую схему управления по следующим этапам:

- планирование работ;
- сбор и анализ данных о происходящих процессах;
- анализ соответствия фактических результатов плановым показателям;
- разработка организационных, финансовых, маркетинговых и иных процедур, снижающих влияние неблагоприятных факторов: снижение рыночного спроса или изменения стоимости комплектующих изделий;
- адаптация дальнейших планов работ с учетом сложившихся условий.

При всей своей очевидности такая схема управления на практике не имеет универсальных решений. Она формируется с учетом специфики и масштаба бизнеса, существующего менеджмента, уровня детализации решаемых задач.

Выработка решений в виде стратегии функционирования и развития производится на основе миссии и целей предприятия с учетом доступных ресурсов и результатов обработки данных обратной связи от объекта управления. Высшее руководство, например, решает задачи определения целей и выработки стратегий развития, формирования и совершенствования организационной структуры, оценки позиций фирмы на рынке и поведения конкурентов, установления ассортимента выпускаемой продукции, организации деятельности аппарата управления компании в целом и т. п. Менеджеры среднего уровня принимают решения, связанные с календарным планированием производства, подбором и расстановкой кадров, реализацией инноваций, систем материального стимулирования и т.д.

Будем считать, что в основе деятельности менеджера-пользователя лежит процесс обнаружения, описания и разрешения проблемных ситуаций

(ПрС). Возникающие в объекте управления ПрС находят свое отражение в базе данных в виде определенных значений атрибутов.

Лицо, принимающее решение (ЛПР), — это субъект решения, т.е. руководитель или менеджер, наделенный надлежащими полномочиями и несущий ответственность за последствия принятого им и реализованного решения.

В своих действиях ЛПР опирается на собственные профессиональные навыки, прошлый опыт, интуицию. Однако при сложных и нечетко сформулированных задачах ЛПР не может использовать опыт, а опора только на интуицию увеличивает риск принятия неверного или неоптимального решения. В подобных ситуациях ЛПР вынужден привлекать к выработке решения экспертов — специалистов в разных областях знаний, для анализа проблемы и подготовки вариантов решений.

Принятие решения — это процесс выбора способа действий, уменьшающего расхождение между существующим (наблюдаемым) и желаемым (возможно, идеальным) состояниями организации. Процесс принятия решения состоит из упорядоченных определенным образом этапов (процедур), содержание которых описывается в терминах цели, проблемы, проблемной ситуации, альтернативы и самого решения как результата выбора альтернативы (варианта действий).

Цель — под этим понимают ожидаемое и желаемое состояние системы, в которое она должна перейти под действием управляющих воздействий и внутренних законов движения экономического объекта.

Различают стратегические и тактические цели. Стратегические цели носят более общий характер и рассчитаны на более длительный период времени, чем тактические.

Проблема — это расхождение между фактически наблюдаемым и желаемым или заданным состоянием управляемого объекта (организации). Проблема возникает, если:

- функционирование организации в данный момент не обеспечивает достижение стоящей перед нею цели;
- функционирование организации в будущем не гарантирует достижения поставленной цели;
- происходит пересмотр целей организации, вызванный, например, изменением общей макроэкономической ситуации, рыночной конъюнктуры и т.п.

Проблемная ситуация (ПРС) — это содержательное описание проблемы совместно с комплексом условий, факторов и обстоятельств, вызвавших её возникновение. Ситуационные факторы, породившие ту или иную проблему, можно подразделить на внутренние и внешние по отношению к организации (объекту управления).

Для каждой группы факторов имеются соответствующие модели анализа и оценок.

Анализ проблемной ситуации — это совместное рассмотрение проблемы в контексте вызвавших ее факторов.

Для краткости изложения будем именовать описывающие проблему признаки, события, оценки и показатели породивших ее причин факторами проблемы (проблемной ситуации). Факторы проблемы могут быть представлены численным значением, логическим высказыванием, лингвистической переменной или текстовым вербальным описанием.

С точки зрения точности формализации описания и последующего выбора решения различают проблемы:

- структурированные;
- слабоструктурированные;
- неструктурированные.

Проблема ***структурированная,*** если удается представить все составляющие её элементы (признаки, проявления, причины, обстоятельства) и зависимости между ними в формализованной (аналитической или логической) форме.

Описание ***слабоструктурированных*** проблем возможно главным образом в виде качественных зависимостей между ее элементами, информация о части которых может отсутствовать. С точки зрения ЛПР, слабоструктурированные проблемы отличаются наличием неопределенностей как в характере зависимостей, так и в значениях их параметров.

Неструктурированной является проблема, для которой могут быть определены зависимости лишь между классами объектов и отношений, к которым они принадлежат.

Решение является реакцией организации на возникшую проблему: оно всегда принимается там, где возникает ПРС. С содержательной точки зрения, решение есть идентификатор программы или плана разрешения проблемной ситуации.

Рассматривая ЛПР как эксперта, чтобы адекватно строить его информационное обслуживание, необходимо учитывать следующие стадии в понимании им сущности ПРС.

1. Интерпретация поступающих данных и выявление проблемы.
2. Структурирование и диагностика проблемы.
3. Классификация ситуации.
4. Проектирование решений.
5. Выбор решения.
6. Реализация решения.

Каждая из стадий, в свою очередь, может состоять из некоторого количества этапов, в зависимости от рассматриваемого вопроса и стадии его решения, ЛПР можно считать находящимся в различных состояниях. Вопросу и фазе его решения соответствует определенное подмножество показателей базы данных. Мгновенное состояние базы данных определяет одно из возможных решений. **Принимаемые решения и состояния базы данных связаны определенными отношениями**. Пользователь может находиться в состоянии неосведомленности относительно значения

некоторых факторов проблемы. В этом случае он генерирует запрос к базе данных и получает ответ в виде значений специфицированных в условиях выдачи запроса атрибутов.

ЛПР осуществляет интерпретацию данных, относящихся к проблемной ситуации. ***Интерпретация*** — это процесс оценки данных; при этом ИС обеспечивает возможность работы ЛПР при наличии неполных и противоречивых данных. Цель данной стадии — выявление существования проблемы, степени ее критичности и приведение ее, если это возможно, к некоторому виду, удобному для понимания и проведения дальнейшего анализа.

На стадии ***структуризации и диагностики*** основная цель ЛПР — выявление основных факторов и зависимостей, обусловивших возникновение ПРС, чему предшествуют уточнение показателей, сопровождающих возникновение проблемы: симптомов проблемы и сбор дополнительной релевантной информации. Диагностика включает в себя выявление отклонений от установленных значений показателей и нарушений функционирования системы.

Классификация ситуации предполагает определение подходящей модели, то есть выбор основных свойств, аналитических зависимостей и/или логических выражений для просчетов вариантов решений.

С целью уточнения значений отдельных параметров ЛПР генерирует запросы в ИС.

На основе понимания ситуации ЛПР переходит к ***проектированию решения***, осуществляя генерацию, анализ и отбор вариантов решения ПРС, используя типовые проблемные ситуации и эвристики (опыт пользователя по разрешению прошлых «подобных» проблемных ситуаций). Процесс проектирования решения включает в себя следующие фазы.

1. *Представление*. ЛПР в процессе разрешения ПРС выдвигает гипотезы относительно подходящей модели. Затем в рамках определенной модели (системы аксиом) осуществляется

декомпозиция, т.е. разложение задачи на подзадачи и выбор конкретных значений переменных, атрибутов и предикатов, входящих в систему аксиом модели ситуации.

2. *Отыскание решения*. Процесс поиска вывода в системе аксиом или нахождение оптимального решения в рамках моделей, построенных ЛПР в результате изучения ситуации. Поиск траектории в пространстве состояний, обеспечивающей достижение заданного состояния с учетом ограничений.

Процесс выбора решения включает в себя следующие фазы.

1. *Оценка решения*. Построение функционала оценки множества вариантов решений, оценка затрат ресурсов. На стадии генерирования альтернативных вариантов ЛПР на основе уточненных им целей определяет критерии, по которым будут оцениваться варианты решений, а также ограничения, которые в соответствии с имеющимися в его распоряжении ресурсами будут накладываться на них.
2. *Прогнозирование* — это предсказание будущих состояний на основе принципов дедуктивного вывода и аналитических оптимизационных моделей. На этом этапе руководитель пытается спрогнозировать последствия принятого решения с использованием таких методов, как прогнозирование на основе временных рядов (методы сглаживания временных рядов, построение тренда). Трудности в этой части заключаются в необходимости учета временных зависимостей, а также модальностей, учета немонотонности теории.

Процесс реализации решения. На завершающей стадии ЛПР осуществляет выбор наилучшего решения и организует реализацию. Процесс реализации включает в себя в качестве этапа: планирование — формирование программы действий по разрешению ПРС, включая выдачу заданий на разработку частных программ, планирование объемов и сроков будущих

работ. План должен учитывать ограниченность ресурсов и противоречивость целей, неполноту данных и возможность их изменения во времени. По сути дела планирование с позиции модальной логики и семантики возможных миров означает описание возможного будущего мира с утверждением, что будущий возможный мир наступит только тогда, когда будет иметь действие ЛПР, которое он намерен осуществить. На этом же этапе определяются структурные подразделения, конкретные исполнители, ответственные за реализацию и контроль исполнения решения.

При решении слабоструктурированных проблем, наибольшие трудности ЛПР испытывает на стадиях выявления, структурирования и диагностики проблемы. Для слабоструктурированных проблем должны быть разработаны и адаптированы к его условиям следующие методы и средства:

- система признаков для регистрации проблемных ситуаций;
- методы оценки степени критичности проблемных ситуаций;
- причинно-следственные диаграммы для диагностирования причин возникновения проблемных ситуаций;
- правила принятия решений (ППР) для формирования и выбора вариантов решений;
- методы прогнозирования результатов решений;
- модели функционирования предприятия и модели внешней среды.

Сложности на стадии выявления слабоструктурированных проблем объясняются отсутствием, неполнотой и противоречивостью данных, необходимых для выявления проблемы, невозможностью установления нормативов для регистрации проблемы в силу быстрых изменений во внешней среде, и постоянной корректировки целей предприятия.

При интерпретации данных сложных слабоструктурированных проблем ЛПР необходимо проанализировать результаты производственно-коммерческой деятельности предприятия, в рамках которого на основе отчетных, плановых и нормативных данных о его состоянии и

функционировании, устанавливаются отклонения и причины их возникновения.

На стадии структурирования и диагностики проблемы необходимо привлечение специалистов различных служб и экспертов. Процесс структуризации подразумевает решение следующих задач.

1. *Классификация* проблем по категориям (таксономия) — определение того, на что влияет существование той или иной проблемы или под влиянием каких сфер деятельности организации и внешних факторов они находятся.
2. *Идентификация* переменных или факторов, составляющих сущность проблемы, воздействующих на проблему или находящихся под ее влиянием.
3. *Оценка зависимостей между переменными* — как они воздействуют друг на друга.

Основной подход для решения задач первой группы — методология ситуационного анализа, т.е. разделение проблем и источников их возникновения на внешние и внутренние. Предлагается использовать в качестве признаков для таксономии проблем основные направления деятельности (функциональные области) конкретного предприятия, а также основные направления воздействия на него внешней среды.

Две другие задачи, которые приходится решать руководителю при структуризации проблемной ситуации, связаны с идентификацией факторов проблемы и определением существующих между ними взаимосвязей. В качестве базовых факторов проблемы целесообразно рассматривать причины и следствия их возникновения, а в качестве отношений между ними — причинно-следственные взаимосвязи. Анализ ситуации позволяет построить причинно-следственные диаграммы («дерево причин») и диаграммы зависимостей. Причинно-следственная диаграмма — это формальное отображение структуры ПРС в виде иерархически незамкнутого графа. Его вершины соответствуют элементам проблемы, отражающим причины ее

возникновения, а дуги — связям между ними. Связь элементов — подпроблем — отображается в виде отношения «причина—следствие».

Выявление проблем осуществляется нахождением отклонения от определенных аналитических зависимостей между этими показателями и дальнейшей их интерпретации.

Для технико-экономических показателей, характеризующих проблемную ситуацию, которые могут быть представлены статистическими рядами данных, используются базовые статистические методы, например, расчет средних, стандартных отклонений, дисперсий.

Так как в качестве основных причин возникновения ПРС выступают состояния и изменения состояний элементов внутренней и внешней среды организации (ресурсы, структурные подразделения, виды продукции, виды затрат, поставщики, потребители), можно говорить о структуризации проблемы в соответствии не только с причинами ее возникновения, но и с элементами внешней и внутренней среды организации, а также характеризующими их показателями. После формирования дерева причин производят построение логических правил, используемых для выбора действий и правил принятия решений (ППР). Первый элемент ППР — таблица условий — описание проблемной ситуации в виде определенной совокупности объектов, их параметров и отношений между ними.

Второй элемент ППР — перечень действий, которые могут быть приняты в той или иной проблемной ситуации. Для комплексного представления результата структуризации проблемы и методов ее решения можно использовать концептуальную модель. В соответствии с этой моделью проблема может быть представлена в виде графа с четырьмя видами вершин:

X, P, Д и C.

X — множество проблем.

P — множество атрибутов описания проблем.

D — множество типовых решений проблем.

C — множество условий, разделяющих проблемы и решения, ориентированные дуги выражают множество отношений, описывающих взаимосвязь проблем.

Логический анализ проблем-причин на низших уровнях иерархии, показывает, что во многих случаях они позволяют сформулировать варианты решения проблем более высокого уровня. В общем случае в качестве вариантов решений можно использовать классы стратегий, предлагаемых в экономической литературе.

Тема 3
Объектно-ориентированное проектирование (ООП) интеллектуальной информационной системы

Объектно-ориентированное проектирование (ООП) — это разработка набора моделей, связанных с понятием объекта, объединяющего состояние и поведение. Краеугольный камень ООП — характеристика программного обеспечения в терминах поведения, т.е. тех действий, которые должны быть выполнены приложением. В практике объектно-ориентированной разработки приложений баз данных концептуальное моделирование должно обеспечить информацию по следующим пунктам:

- объекты и их отношения с другими объектами;
- поведение объектов;
- взаимодействие между объектами.

Концептуальные модели должны использовать обозначения, которые непосредственно выражают семантику реального мира и имеют достаточную и выразительную силу для описания сложных проблем. Модель должна:

- позволять разработчику представлять объекты, классифицировать их на значимые множества объектов;
- описывать отношения между объектами и множествами объектов и группировать отношения в определенные группы;
- позволять описывать поведение объекта, состояния, в которых находится объект, условия при которых происходит переход из состояния в состояние и потенциальные действия, которые являются допустимыми в различных состояниях;
- позволять разработчику определять по отношению к взаимодействию объектов, как объекты общаются и обмениваются информацией друг с другом, чтобы иметь соответствующее формальное обоснование.

Синтаксис и семантика всех конструкций модели должны быть выражены математически. При разработке системы методом ООП первая

задача — составление спецификации, при этом действия, осуществляемые системой, разбиваются на компоненты.

Компонента — это абстрактная единица, могущая выполнять определенную работу. На первом этапе нет необходимости знать в точности, как задастся компонента или как она будет выполнять свою работу. Компонента должна:

- иметь небольшой набор четко определенных обязанностей;
- взаимодействовать с другими компонентами настолько слабо, насколько это возможно.

Чтобы выявить отдельные компоненты и определить их обязанности, необходимо разработать сценарий работы системы. При разработке сценария работы системы составляется спецификация в виде CRC-записей: Компонент (Component), Обязанности (Responsibility), Соисполнитель (Collaborator). Далее прорабатываются вопросы:

- что требуется делать;
- кто это будет делать.

Секрет хорошего объектно-ориентированного проекта в том, чтобы установить агента для каждого действия, могущего реализовать один или несколько из доступных для него методов, которые он выполняет в соответствии с протоколом. В процессе разработки спецификации для каждой компоненты придумывается название, и в ее карточку вписываются действия. Например: просмотр базы знаний, ввод записей, корректирование записей, корректировка выходного отчета, разработка нового отчета.

Важный принцип ООП — готовность к изменениям. Независимо от того, насколько тщательно вы пытаетесь разработать исходные спецификации, неизбежны те или иные изменения. Разработчики должны предвидеть это и планировать свои действия соответствующим образом. Изменения должны затрагивать как можно меньше компонент. Необходимо попытаться предсказать наиболее вероятные источники изменений и

позволить им влиять на возможно меньшее количество компонент программы. Наиболее быстро изменяющиеся элементы системы:

- интерфейс;
- форматы обмена информацией;
- форматы выходных данных.

Необходимо постараться уменьшить также зависимость приложения от характеристик программных средств.

Уменьшение количества связей между фрагментами приложения снижает взаимозависимость между ними и увеличивает вероятность того, что каждую компоненту удастся изменить с минимальным воздействием на другие. Выбрав компоненты и их функции, создаем схему взаимодействия между ними. Для этой цели используются диаграммы взаимодействия. Поведение компоненты — это набор действий, ею осуществляемых. Полное описание компоненты иногда называют протоколом. Говоря о состоянии компоненты, имеют в виду ее внутреннее содержание. Состояние не является статическим и может изменяться с течением времени.

Связность — это мера того, насколько отдельная компонента образует логически законченную осмысленную единицу. Высокая связность называется зацеплением и возникает в результате объединения в одной компоненте связанных друг с другом функций. Наиболее часто оказываются связанными друг с другом функции, для которых необходимо иметь доступ к общим данным. С другой стороны, зацепление характеризует взаимозависимость между компонентами. В общем случае желательно уменьшить степень зацепления, поскольку связи между компонентами препятствуют их модификации и мешают дальнейшей разработке или повторному использованию. В частности, зацепление возникает, когда одна компонента должна иметь доступ к данным (состоянию) другой компоненты. Следует избегать подобных ситуаций и определить, как будет реализована каждая из компонент.

– Компоненты со многими функциями лучше реализовать в виде классов. Каждой обязанности в CRC-записи присваивается имя. Эти имена станут затем названиями функций или процедур. Вместе с именами определяются типы аргументов, передаваемых функциям. Затем описывается вся информация, содержащаяся внутри компоненты. Если компоненте требуются некоторые данные для выполнения конкретного задания, их источник (аргумент функции, глобальная или внутренняя переменная) должен быть явно описан.

При переходе от класса к дочернему классу или экземпляру объекта имеет место так называемое наследование свойств. Существуют следующие формы наследования:

– *специализация* — дочерний класс более конкретный, частный или специализированный случай родительского класса;
– *спецификация* — родительский класс описывает поведение, реализуемое в дочернем классе, но оставленное не реализованным в родительском;
– *конструирование* — дочерний класс использует методы, предоставляемые родительским классом, но не является подтипом родительского класса;
– *обобщение* — дочерний класс модифицирует или переопределяет некоторые методы родительского класса с целью получения объекта более общей категории;
– *расширение* — дочерний класс добавляет новые функциональные возможности к родительскому классу, но не меняет наследуемое поведение;
– *ограничение* — дочерний класс ограничивает использование некоторых методов родительского класса;
– *варьирование* — дочерний и родительский классы — варианты на одну и гуже тему, и связь «класс-подкласс» произвольна;
– *комбинирование* — дочерний класс наследует черты более чем одного родительского класса. Это множественное наследование.

Действие в объектно-ориентированном программировании инициируется путем передачи сообщений агенту (объекту), ответственному за действие. Сообщение содержит запрос на осуществление действия и сопровождается дополнительной информацией (аргументами), необходимой для его выполнения.

Получатель — это агент, которому посылается сообщение. Если он принимает сообщение, то на него автоматически возлагается ответственность за выполнение указанного действия. В качестве реакции на сообщение пользователь запускает некоторый метод, чтобы удовлетворить принятый запрос.

Принцип маскировки: клиенту, посылающему запрос, не требуется знать о фактических средствах, с помощью которых его запрос будет удовлетворен. Если имеется работа, которую нужно выполнить, первая мысль клиента — найти кого-либо, кому эту работу можно поручить.

Фундаментальной концепцией в объектно-ориентированном программировании является понятие ответственности за выполнение действия. Полный набор обязанностей, связанных с определенным объектом, определяют с помощью понятия протокол. Все объекты — представители или экземпляры классов. Метод, активизируемый в ответ на сообщение, определяется классом, к которому принадлежит получатель сообщения. Все объекты одного класса используют один и тот ответ на одинаковые сообщения. Принцип, в соответствии с которым знание о более общей категории разрешается использовать для более узкой категории, называется наследованием. Классы могут быть организованы в иерархическую структуру с наследованием свойств. Дочерний класс (или подкласс) наследует атрибуты родительского класса или подкласса, расположенного выше в иерархическом дереве.

Абстрактный» родительский класс — это класс, не имеющий экземпляров; используется только для порождения подклассов. Информация,

содержащаяся в подклассе, может переопределить информацию, наследуемую из родительского класса.

При разработке системы выделяют следующие типовые категории классов: класс управления (администрирования) данными, класс источников данных или посредников в передаче данных, классы для просмотра данных, вспомогательные классы.

Классы—администраторы данных (Date Managers) часто получают имена, например, Date или State-классы, основной обязанностью которых является поддержка данных или информации о состоянии чего-либо. Источники данных (Date Sources) генерируют данные по запросу. Классы для просмотра данных View или Observer осуществляют вывод информации на экран.

Основная компонента ООП — *объект*. Объект может быть атомарным или молекулярным, имеющим составные части или компоненты, которые в совокупности составляют объект. Объект ООП — лицо, место или предмет — может быть физическим или концептуальным. Объект, представление которого неотличимо от самого объекта, называется *лексическим*. Имя уникальным образом идентифицирует множество объектов.

Учитывая, что основным принципом ООП подхода является итеративное прототипирование, разработчику приходится постоянно проектировать, генерировать код, модифицировать, развивать прототип, снова генерировать, модифицировать и развивать, каждый раз реализуя все большее количество требований. Более того, изменения вносятся в коды программы, поэтому инструмент должен обеспечивать реинжиниринг, чтобы восстановить по коду диаграммы моделей и сравнить их с предыдущим вариантом.

Наиболее характерный представитель средств ООП — система Rational Rose. Эта система предназначается для автоматизации этапов анализа и проектирования, а также для генерации кодов на различных языках и выпуска проектной документации. Все CASE-средства, входящие в состав

Rational Rose, используют методы Гради Буча (научного руководителя разработок) и систему OMT Джеймса Рамбо, поддерживая единый графический интерфейс с пользователем. Различия между этими системами минимальны и определяются языками, на которых генерируются коды программ.

Все перечисленные выше компоненты общие для всех систем семейства. К ним следует добавить генератор кодов, индивидуальный для каждой системы, и анализатор — отдельный программный модуль, обеспечивающий реинжиниринг.

Репозитарий — это объектно-ориентированная база данных. Средства просмотра обеспечивают «навигацию» по проекту, в том числе перемещение вверх/вниз по иерархиям классов и подсистем, переключение из одного вида диаграмм к другому и т.д. Средства контроля и сбора статистики дают возможность находить и устранять ошибки по мере развития проекта, а не после завершения его описания. Генератор отчетов формирует тексты выходных документов на основе содержащейся в репозитарий информации.

Объектно-ориентированный язык UML (Unified Modeling Language) предназначен для специфицирования, визуализации и документирования систем программного обеспечения. UML сосредотачивается на тех явлениях, которые не описываются существующими языками, например на моделировании конкурентных распределенных систем. UML — объектно-ориентированный язык, и не является языком визуального программирования. UML является развитием языков OMT, OOSE и других методов.

Главная цель моделирования — подготовить описания для многих конкретных объектов, используя метод дихотомии тип-экземшяр. Роли, как и экземпляры идентифицируют различные элементы, принадлежащие одному классификатору. Как и типы, они описывают элемент, который может иметь много различных экземпляров и допускает возможность повторного использования. Схема объекта — это граф экземпляров, включающий

объекты и значения данных. Статическая диаграмма объекта — это экземпляр класса. Классификатор — это метамодель суперкласса, включающего класс, тип данных и интерфейс. Класс — дескриптор для множества с аналогичной структурой, поведением и отношениями. Классы имеют структуру данных, поведение и связи с другими элементами.

Инварианты и наборы правил. При выполнении объектно-ориентированного анализа и моделирования интеллектуальных ИС необходимо использовать результаты, полученными и области искусственного интеллекта, семантических сетей и семантического моделирования данных Спецификации, которые отражают инкапсуляцию атрибутов и операций, очень хороши, но не обязательно четко описывают аспекты, которые имели в виду аналитики или пользователи. Для многократного использования спецификация объекта должна содержать знания объекта (атрибуты), его действия (операции), а также мотивы этих действий и связь с интерфейсами других объектов. Семантика частично содержится в структурах ассоциаций, классификации, композиции и использования, а также в утверждениях, инвариантах и наборах правил, которые описывают поведение объекта. Введение семантики ограничивает многократное использование, но делает его более безопасным.

В процессе проектирования приходится решать, принадлежат ли правила отдельным операциям или объекту в целом. Правила, связывающие несколько операций, не определяют зависимости между атрибутами и относятся к объекту в целом. Правила, которые касаются инкапсуляции состояния объекта, находятся в пределах одной из его операций. Наиболее важный вид правил "объекта в целом" — это правила управления, которые описывают поведение объекта, его участие в структурах, к которым он принадлежит. Это правила управления значениями по умолчанию, множественного наследования, исключительных ситуаций и ассоциаций с другими объектами. Введение инкапсулированного набора правил является новым аспектом и расширяет объектные модели и добавляет множество

наборов правил к каждому объекту. Таким образом, если обычный объект состоит из идентификатора (identifier), атрибутов (attributes) и операций (operations), то объект, который используется при моделировании интеллектуальной ИС, состоит из идентификатора, атрибутов, операций и наборов правил. Наборы правил составлены из неупорядоченного набора утверждений и правил или в форме "если..., то...". Введение правил приводит к тому, что объекты, которые являются локальными сущностями, могут формировать правила для глобального системного поведения. Это приводит к тому, что объекты с набором правил могут рассматриваться в качестве интеллектуальных агентов для экспертных систем.

При моделировании ИИС считается, что задавать только атрибуты и операции (сигнатуру) объекта недостаточно. Необходимо определять каким образом объект взаимодействует с другими объектами и каким правилам и ограничениям подчиняется он для обеспечения целостности. В некоторых системах это достигается путем использования формальных утверждений. Утверждения в таких системах делятся на два вида, утверждения об операциях и утверждения о целом объекте. Первые — это типичные предусловия и постусловия, в то время как последние называют инвариантами класса. Также имеются утверждения, представляющие ограничения атрибутов.

Утверждения атрибутов делятся на следующие виды.

1. Ограничения диапазона описывают пределы изменения допустимых значений.
2. Ограничения перечня перечисляют допустимые значения.
3. Ограничения типа определяют класс, которому должны принадлежать значения Ограничения типа всегда представляют и обобщают первые два случая.

Операционные утверждения делятся на следующие:

- **предусловие** — это отдельное логическое утверждение, которое должно быть истинным до выполнения соответствующей операции.

- **постусловие** — это отдельное логическое утверждение, которое должно быть истинным после того, как соответствующая операция закончила выполнение.
- **условие инвариантности** — это отдельное логическое утверждение, которое, быть постоянным при выполнении операции. Это имеет значение только для параллельных систем обработки (включая модели бизнес-процессов). Различают два вида условий инвариантности: гарантия и выражение уверенности;
- **выражение уверенности** устанавливает предварительное условие, которое должно оставаться истинным в течение выполнения операции, к которой это относится. Если оно нарушено, спецификация не устанавливает результата. Сервер не отвечает за поддержку этого условия.
- **гарантия** — это утверждение, которое сервер должен поддерживать как истину в течение выполнения операции.

Аспекты операции могут включать более одного утверждения любого из этих типов утверждения могут быть представлены в соответствии с диаграммами состояний.

Объектные утверждения и наборы правил делятся на следующие виды:

- **инвариант класса** — это одно логическое утверждение о любом подмножестве возможностей объекта, которое должно быть истинным всегда. Ограничения кратности для атрибутов являются инвариантами. Инварианты можно также назвать правилами.
- **набор правил** — это неупорядоченный набор инвариантов класса (или правил) и утверждений об атрибутах с определенным режимом логического вывода, который обеспечивает объединение правил. Внешние наборы правил выражают информацию второго порядка, такую как стратегия управления. Внутренние наборы правил - это

множества инвариантов (первого порядка). Они могут быть написаны на естественном языке или на исполняемом языке правил.

- **эффект** — это постусловие, объединенное со всеми другими постусловиями операций типа. Эффект обычно имеет такую форму: (любое изменение f(x, x@ pre) => условие). Эффекты могут быть выражены в виде правил. Они полезны для проектирования супертипа и наложения ограничений на операции.

Обычно, помимо сказанного выше, предполагается, что логика, лежащая в основе вышеупомянутых определений, является стандартным исчислением предикатов первого порядка.

Интерфейс — это список сообщений, которые можно передать объекту, со своими параметрами и типами возвращаемых значений. В зависимости от интерпретации, он может включать операции получения и присваивания значении атрибутам. Это понятие иногда рассматривается как сигнатура типа. С другой стороны, тип — это полная спецификация, включающая все утверждения, которые могут быть сделаны о типе и его экземплярах, и все наборы правил.

Инвариант или **ограничение** — это отдельное правило, которому всегда подчиняется объект. Оно выражается с использованием терминологии, обеспечиваемой моделью типов. Примерные инварианты для системы управления авиалинией могли бы включать следующее: Каждый пилот должен быть капитаном или вторым пилотом одного полета в день", "Капитаном и вторым пилотом не может быть один и тот же человек" или "Каждый полет должен осуществляться капитаном, который имеет квалификацию для управления этим типом самолета". Инвариант типа может иметь отношение к общим интерфейсам других типов. Инварианты могут быть выражены неформально, как в приведенном выше примере, или с использованием формализованного языка или других формальных логических систем.

Наборы правил обобщают инварианты класса и позволяют объектам выполнять следующие функции:

- логически выводить значения атрибутов, которые не хранятся явно;
- представлять триггеры базы данных;
- представлять операции непроцедурным способом;
- представлять режимы управления.

Правила определяют информацию второго порядка, такую как зависимости между атрибутами (например, зависимость между возрастом служащей и ее правом на отпуск) Глобальные пред- и постусловия, которые применяют ко всем операциям, могут быть определены как правила. Типичное бизнес-правило могло бы предполагать «изменение продолжительность отпуска до шести недель, если стаж службы превышает пять лет» в качестве одного из правил для типа Employee (Служащий). Наборы правил позволяют решать проблемы анализа в тех случаях, когда в качестве целевой среды предусмотрена активная база данных.

Правила и инварианты используются для уточнения семантики объекта. Это помогает при описании информации, которая обычно постоянно хранится в архиве, например информации о деловой активности предприятия. Это также обеспечивает функциональную совместимость на довольно низком уровне. Например, имеется объект, который вычисляет кубические корни. Клиенту этого объекта недостаточно знать только его операции; необходимо также знать что возвращаемое значение является кубическим, а не квадратным корнем. В этом простом случае решение очевидно, поскольку кубический корень можно уникально охарактеризовать с помощью одного простого правила: ответ, дважды умноженный на себя, равен переданному параметру. Если это правило является частью интерфейса, тогда все другие системы и системные компоненты могут понять функции объекта только из его интерфейса. Таким образом устраняются некоторые сложности технологии хранения информации за счет ее перемещения в объектную модель.

Описанные правила могут быть классифицированы на несколько, не обязательно взаимоисключающих типов.

- бизнес-правила;
- правила обработки исключительных ситуаций;
- триггеры;
- правила управления.

Спецификаторы (qualifier) языка UML могут быть выражены как правила. В качестве примера можно рассмотреть ассоциации "многие ко многим" между файлами и каталогами DOS. Спецификатор Filename (имя_файла) сводит это к ассоциации «многие к одному». Явная квалификация только расчленяет множества. Квалификация относительна – она имеется степень уточнения.

В языке UML инварианты и наборы правил можно записать в (дополнительном) четвертом именованном разделе, расположенном подразделом операций в изображении типа.

Основанные на правилах расширения объектно-ориентированного анализа помогают обогатить семантику моделей для интеллектуальных информационных систем. Обогащенная семантика делает эти модели удобочитаемыми и в большей степени обратимыми; большее количество замыслов аналитика становится в модели более наглядным. Это обеспечивает подлинно объектно-ориентированный подход к спецификации расширенных систем баз данных и систем, основанных на использовании знаний.

В приложениях содержатся не только правила управления, но и бизнес-правила. В обоих случаях инкапсуляция правил в объектах имеет смысл и увеличивает возможность многократного использования и расширяемость. Правила системного уровня подходят для большинства общих объектов в системе, которые являются вершиной иерархии (или иерархий, если отсутствует единый базовый класс, такой, например как Object в C#). Они распространяются на другие части системы через наследование. Все виды

правил сохраняются в наборах правил дополнительного четвертого раздела в изображении объекта.

Тема 4. Интеллектуальные системы поддержки принятия решений и экспертные системы

4.1. Интеллектуальные информационные системы поддержки принятия решений

Опыт эксплуатации информационных систем в организационных и экономических системах показал, что наиболее важное значение должен иметь в этих системах и в контуре управления — человек (управленец; лицо, принимающее решение— ЛПР).

Не следует забывать, что управление в экономических и организационно-технических системах является сложным творческим процессом, нуждающимся в различных формах обеспечения интеллектуальной деятельности. Преуменьшение значения творческого элемента (опыта, интуиции) и, наоборот, преувеличение возможностей формализации ряда управленческих задач, неизбежно ведет к тому, что реальные результаты далеко не полностью оправдывают ожидания, которые связывались и связываются с компьютеризацией управления и принятия решений.

Видимо, здесь кроется причина недостаточно эффективного использования в ИС и системах поддержки решений методов оптимизации. Говоря о взаимодействии пользователя с оптимизационными моделями априорно подразумевают адекватность этих моделей реальному объекту. Однако, сложность, существенная нелинейность, слабая структурированность задач, неясность предпочтений, нечеткость исходной информации не позволяют в большинстве случаев разработчикам создавать адекватные модели объектов. «Ключом» в этом направлении должны стать и уже активно становятся методы и модели искусственного интеллекта (ИИ), в частности прикладные системы, базирующиеся на знаниях (или интеллектуальные системы).

Большинство имеющихся объектов управления относятся к слабоструктурированным или плохо определяемым объектам, которые

обладают рядом неожиданных для традиционного управления свойств, таких, как уникальность, отсутствие формализуемой цели существования, отсутствие оптимальности, высокая динамичность, неполнота описания объекта, и, наконец, индивидуальность поведения лица, принимающего решения в процессе принятия решений,

Практика показала, что трудности, практически непреодолимые для «управленца-компьютера» оказываются под силу «управленцу-человеку». Квалифицированный эксперт после определенного времени работы по управлению уникальным объектом справляется и с неполнотой описания объекта, и с нечеткостью исходной информации, и с отсутствием формализуемых целей (разумеется, имеется в виду управление основными управляющими параметрами).

Следовательно, в процессе практической деятельности по управлению объектом ЛПР приобретает некоторый инструмент, который помогает ему в решении задач управления плохо определенными объектами. Этот инструмент есть не что иное, как знание. Таким образом, возникла идея необходимости автоматизации интеллектуальной деятельности человека.

Основное назначение информационных систем в экономике — это современное представление необходимой информации ЛПР для принятия им адекватных и эффективных решений при управлении процессами, ресурсами, финансовыми транзакциями, персоналом или организацией в целом. Однако в процессе развития информационных технологий, исследования операций и технологий моделирования, а также с возрастанием потребителей информационно-аналитической поддержки самих ЛПР, все больше проявлялась потребность в системах, не только представляющих информацию, но и выполняющих некоторый ее предварительный анализ, способных давать некоторые советы и рекомендации, осуществлять прогнозирование развитие ситуаций, отбирать наиболее перспективные альтернативы решений, т.е. поддерживать решения ЛПР, взяв на себя

значительную часть рутинных операций, а также функции предварительного анализа и оценок.

Информационная система поддержки принятия решений (ИСППР) решений связывает интеллектуальные ресурсы менеджера со способностями и возможностями компьютера для улучшения качества решений. ***Эти системы предназначены для менеджеров, принимающих управленческие решения в условиях полуструктурированных и слабо определенных задач***.

Таким образом, дальнейшее развитие ИСППР привело к созданию интеллектуальной ИСППР.

Интеллектуальная ИСППР — это компьютерная система, состоящая из 5 основных взаимодействующих компонентов:

- языковой подсистемы (механизм обеспечения связи между пользователем и другими компонентами интеллектуальной ИСПР);
- информационной подсистемы (хранилище данных и средств их обработки);
- подсистемы управления знаниями (хранилище знаний о проблемной области, таких как процедуры, эвристики и правила, и средства обработки знаний);
- подсистемы моделей (хранилище моделей, языки моделирования, средства управления моделированием);
- подсистемы обработки и решения задач (связующее звено между другими подсистемами).

Подсистема обработки и решения задач распределена и функционально встроена в другие подсистемы, реализуя свои отдельные специфические функции в их рамках. Эта подсистема обладает основными способностями по манипуляции и обработке задач для принятия решений.

На рис. 4.1 представлен вариант структуры интеллектуальной ИСППР.

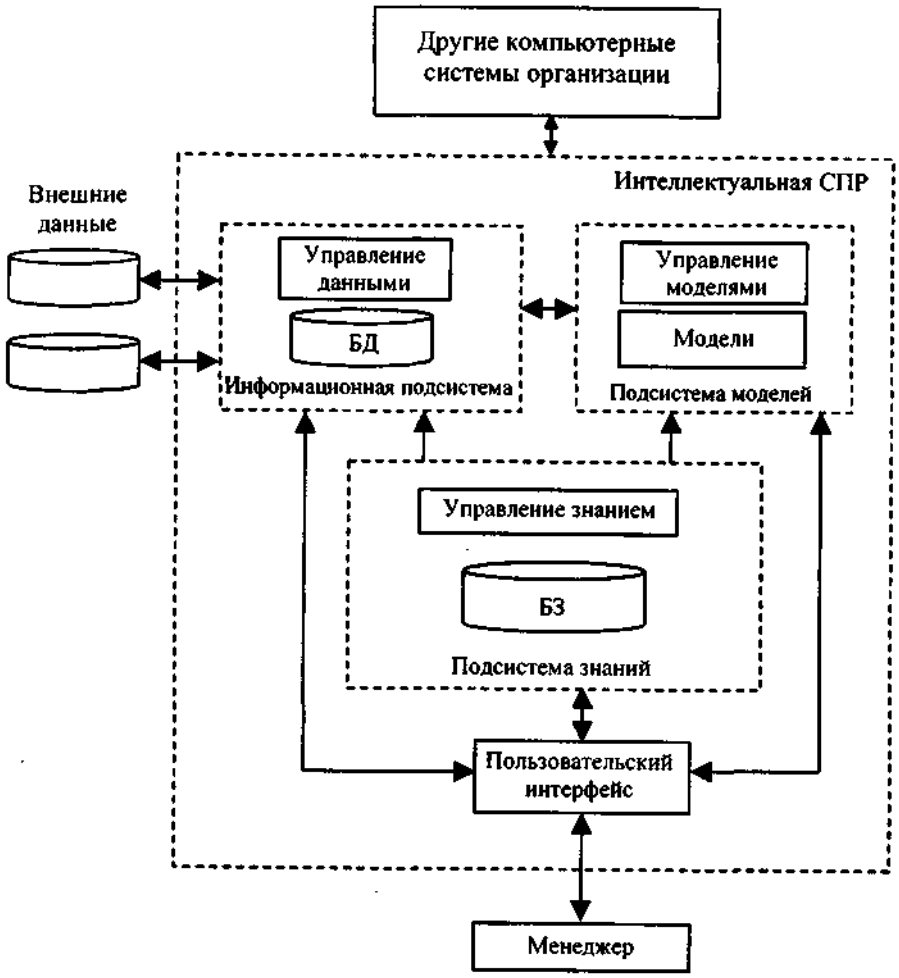


Рис. 4.1. Схематическое представление интеллектуальной СПР

Информационная подсистема состоит из БД, системы управления БД, (средств организации запросов, справочника данных, внешних источников данных.

Подсистема моделей состоит из базы моделей, системы управления моделями, языков моделирования, справочника моделей и процессора, который осуществляет реализации на модели, интегрирует модели и осуществляет руководство процессом моделирования.

База моделей содержит обычные и специальные статические, финансовые, прогнозирующие, управленческие и другие количественные модели, которые обеспечивают аналитические способности интеллектуальной ИСПР. Способность обращаться к моделям, реализовывать их прогоны, вносить изменения, комбинировать и проверять модели

являются ключевой способностью интеллектуальной ИСПР, которая отличает их от обычных информационных систем.

Модели в базе моделей могут подразделяться на стратегические, тактические, операционные и составные стандартные блоки моделей.

Функциями системы управления моделями являются создание моделей с использованием стандартных модельных модулей, генерация новых стандартных модулей и отчетов, дополнение и модернизация моделей, их изменения и манипулирование с данными модели.

Процессор подсистемы моделей обычно реализует следующие действия:

- исполнение модели, т.е. процесс управления текущим прогоном или реализацией модели;
- интеграция модели, т.е. совмещение операций нескольких моделей, когда это необходимо;
- подтверждение и интерпретация инструкций моделирования, поступающих от диалогового компонента системы и проведение их в систему управления моделями.

Пользовательский интерфейс реализует все аспекты коммуникации между пользователем и ИСПР.

Подсистема управления знаниями. Многие неструктурированные и слабоструктурированные задачи являются такими сложными, что они требуют для своего решения экспертизы, дополнительно к обычным способностям ИСППР.

Такая экспертиза может быть обеспечена ЭС или другой интеллектуальной системой.

Поэтому большинство первых интеллектуальных ИСППР оснащены системной компонентой для управления знаниями. Такая компонента может обеспечить требуемую экспертизу для решения некоторых видов задач и обеспечивать действие других составных частей ИСППР.

Возможны различные способы интеграции интеллектуальных систем, основанных на знаниях, с математическим моделированием.

Например, часто решения, основанные на знаниях, помогают поддерживать шаги в процессе получения решения без математической поддержки; интеллектуальные системы моделирования решений могут помочь пользователям строить, использовать и управлять библиотекой или базой моделей; аналитические экспертные системы (ЭС) принятия решений могут интегрировать теоретически строгие методы неопределенности в базу знаний ЭС.

Компонента знаний состоит из одной или нескольких интеллектуальных программных составляющих. Как СУБД и система управления моделями, программное обеспечение управления знаниями обеспечивает требуемое исполнение и интеграцию в интеллектуальных системах.

Информационные СППР, которые включают такую составляющую, называются интеллектуальными информационными СППР, интеллектуальными СППР, экспертными СППР, экспертными системами или СППР, базирующимися на знаниях.

Необходимость использования интеллектуальных систем. Существует множество доводов в пользу того, что интеллектуальные системы могут и должны стать важнейшей составной частью в системах поддержки решений, при управлении сложными объектами в технологии современных производств и решении широкого спектра экономических задач.

Если в качестве примера объекта взять предприятие, то здесь при управлении возникают следующие проблемы:

- преодоление сложности (сложности управления возникают тогда, когда приходится делать выбор из множества возможных решений);
- организация больших объемов информации;

- как уменьшить информацию до того уровня, который необходим для принятия решения (потеря информации, поступающей от объектов, работающих в реальном режиме времени, может существенно сказаться на результате);
- нехватка времени на принятие решения (проявляется по мере усложнения производства);
- координация (решения необходимо координировать с другими звеньями процесса или объекта);
- необходимость сохранения и распространения знаний очень опытных экспертов, полученных ими в процессе многолетней работы и большого практического опыта. Проблема извлечения знаний и их распределения — сегодня одна из главных проблем.

В процессе своей управленческой (а вообще говоря, любой) деятельности человек получает и осознает огромное количество информации. Однако ограниченные возможности человеческого мозга заставляют его осуществлять вербальное перекодирование исходной информации в сгустки насыщенной информации, используя при этом уникальные возможности человеческого языка. Едва ли не все рассуждения человека по своей природе являются приближенными. При этом, используя простые эвристические правила вывода, человек легко справляется с нечеткими рассуждениями.

Специалисты в области ИИ всегда старались разработать программы для компьютеров, которые могли бы в некотором смысле «думать», т.е. решать задачи таким способом, который мы бы сочли разумным, если бы его применил человек.

В процессе исследований и 20-летних поисков они пришли к выводу, что эффективность программы при решении задач зависит от знаний, которыми она обладает, а не только от формализмов и схем вывода, которые она использует. То есть, чтобы сделать программу интеллектуальной, ее нужно снабдить множеством высококачественных специальных знаний о некоторой предметной области.

Понимание этого факта привело к созданию специальных систем, каждая из которых является экспертом в некоторой узкой предметной области. Эти программы получили название экспертных систем.

4.2. Экспертные системы — основная разновидность интеллектуальных систем

Экспертная система— это система, которая использует человеческие знания, встраиваемые в компьютер, для решения задач, которые обычно требуют человеческой экспертизы. Хорошо разработанные системы имитируют процесс рассуждения экспертов, используя это для решения специфических задач.

Такие системы могут использоваться не экспертом для улучшения их способностей и возможностей в решении задач определенного класса в конкретной предметной области. ЭС могут быть также использованы для распространения источников редких знаний.

В конечном счете, такие системы могут функционировать лучше, чем некоторые отдельные эксперты — люди при выработке решения или суждения в специфической, обычно узкой области экспертизы.

Эта возможность может иметь значительное влияние как на деятельность таких профессиональных консультантов, как финансовые аналитики, юристы, аудиторы и др., так и на организации и их менеджмент.

Технологию построения ЭС часто называют ***инженерией знаний***. Этот процесс требует специфической формы взаимодействия создателя ЭС, которого называют ***инженером знаний***, и одного или нескольких экспертов в некоторой предметной области. Инженер знаний «извлекает из экспертов стратегии, эмпирические правила, которые они используют при решении задач, описания, и встраивает эти знания в ЭС (рис. 4.2).

Рис. 4.2. Процесс построения ЭС

ЭС — это сложные программы, которые манипулируют знаниями в целях получения эффективного решения в узкой предметной области. Как и настоящий человек-эксперт, эти системы используют символическую логику и эвристики (эмпирические правила) чтобы найти решения. Они могут ошибаться, но обладают способностью учиться на своих ошибках.

Основные понятия ЭС. Основными понятиями ЭС являются:

- экспертиза;
- эксперты;
- проведение экспертизы;
- вывод и объяснительные способности.

Экспертиза — это специфическое знание необходимое для решения задачи, извлеченное из обучения, чтения и опыта. Следующие типы знаний являются примерами того, что включает в себя экспертиза:

- теории о проблемной области;
- правила и процедуры относительно проблемной области;
- правила (эвристики) о том, что делать в данной проблемной ситуации;
- глобальные стратегии для решения таких типов задач;
- мета-знания (знания о знаниях);
- факты о проблемной области.

Эти типы знаний дают возможность ЛПР принимать решения лучше и быстрее при решении сложных задач.

Эксперты. Трудно дать определение понятию эксперт, так как мы в действительности говорим о разных степенях или уровнях экспертизы. Возникает также вопрос: каким объемом экспертных знаний в данной области и какими навыками должен обладать человек, чтобы стать квалифицированным экспертом?

Обычно, человеческая экспертиза включает многогранное интеллектуальное поведение, которое вовлекает в процесс следующие виды деятельности:

- выявление и формулировка проблемы и задачи;
- решение задачи быстро и надлежащим образом;
- объяснение решения;
- обучение из опыта;
- реструктуризация знаний;
- отказ при необходимости от устоявшихся правил и шаблонов;
- определение уместности и соответствия;
- осознание ограничений.

Для имитации эксперта-человека необходимо создать компьютерную систему, проявляющую все эти характеристики. Однако в современных ЭС прежде всего исследованы и разработаны вторая и третья из этих видов деятельности (решение задач и объяснение решений).

Проведение экспертизы. Целью ЭС является проведение экспертизы путем аккумуляции знаний от экспертов и предоставлению их другим людям (не экспертам). В этот процесс вовлечены четыре вида деятельности:

- извлечение знаний (из экспертов или других источников);
- представление знаний (в компьютере);
- вывод знаний;
- передача знаний пользователю.

Знания хранятся в компьютере в базе знаний (БЗ).

Вывод. Уникальной чертой ЭС является их способность рассуждать («думать»). Имеется в виду, что необходимые знания для экспертизы хранятся в БЗ, программа может иметь доступ к соответствующим данным в БД, а ЭС может делать логический вывод, получая нужное знание, зачастую не хранящееся в явном виде в БЗ. Процесс вывода осуществляется составляющей системы, которая называется машина вывода.

Способность объяснять. Другой уникальной чертой ЭС является ее способность объяснять свои советы или рекомендации. Объяснение и обоснование производятся подсистемой объяснений. Она дает возможность системе проверять свои рассуждения и объяснять их действия.

4.3. Функциональные возможности и характеристика экспертных систем

Основными характеристиками ЭС являются:

- накопление и организация знаний;
- знания — основа ЭС, они являются явными и доступными, что отличает эти системы от большинства традиционных программ;
- применение для решения проблем высококачественного опыта квалифицированных экспертов. Именно высококачественный опыт в сочетании с умением его применять делает систему рентабельной. Этому также способствует гибкость системы;
- наличие прогностических способностей. ЭС может объяснить каким образом новая ситуация привела к изменениям;
- ведущие специалисты уходят, но их опыт остается и используется в ЭС;
- ЭС можно использовать для обучения и тренировки.

Преимущества ЭС.

Зачем разрабатывать ЭС? Не лучше ли обратиться к человеческому (опыту, как это было в прошлом. Приведем доводы в пользу ЭС (табл. 4.1).

Таблица 4.1. Сравнение человеческой и искусственной компетентности

Человеческая компетентность	**Искусственная компетентность**
Непрочная	Постоянная
Трудно передаваемая	Легко передаваемая
Трудно документируемая	Легко документируемая
Непредсказуемая	Устойчивая
Дорогая	Приемлемая по затратам

Здесь очевидны преимущества искусственной компетентности. Кроме того, эксперт-человек может принимать различные решения в тождественных ситуациях из-за эмоциональных факторов (влияние дефицита времени, стресса).

Необходимость человека в контуре управления. Почему необходимо ставить для человека место в системе?

Если искусственная компетентность настолько лучше человеческой, почему бы полностью не отказаться от экспертов-людей, заменив их ЭС? (Однако, такая постановка вопроса несостоятельна в силу ряда причин):

- хотя ЭС хорошо справляются со своей работой, но в некоторых областях деятельности человеческая компетентность превосходит любую искусственную. Это не есть отражение фундаментальных ограничений ИИ, но характерно для современного его состояния. Например, область творчества;
- обучение: человеческая компетентность пока превосходит искусственную. Эксперты адаптируются к изменяющимся условиям, приспосабливают свои стратегии к новым обстоятельствам. ЭС мало приспособлены к обучению новым концепциям и правилам. Обучающие программы разработаны для простых задач и мало пригодны, когда требуется учитывать всю сложность реальных задач; эксперты могут непосредственно воспринимать комплекс входной сенсорной информации (визуальной, звуковой, осязательной, обонятельной и тактильной). ЭС — только символы. Хотя в отдельных направлениях разработки инженерных и производственных интеллектуальных

систем получены реальные результаты определенной обработки сенсорной информации;

- эксперты-люди могут охватить картину в целом, все аспекты проблемы и понять, как они соотносятся с основной задачей. ЭС стремится сосредоточиться на самой задаче, хотя смежные задачи могут повлиять на решение основной;
- люди, эксперты и не эксперты, обладают тем, что мы называем здравым смыслом, или общедоступными знаниями. Это широкий спектр общих знаний о мире, о том, какие законы в нем действуют, т.е. знания, которыми каждый из нас обладает, приобретает из опыта и которыми постоянно пользуется. Из-за огромного объема знаний, образующих здравый смысл, не существует легкого способа встроить их в интеллектуальную программу. Знания здравого смысла включают знания о том, что вы знаете и чего не знаете.

Поэтому ЭС наиболее часто используются как советчики, в качестве консультантов или помощников ЛПР.

Функциональные возможности ЭС определяются двумя ее главными системными частями: средой развития и средой рекомендаций (рис. 4.3). Среда развития используется разработчиком ЭС для построения компонентов и размещения знаний в БЗ. Среда рекомендаций используется неэкспертами для получения экспертных знаний и советов.

Три главных компонента, которые проявляются виртуально в каждой ЭС — это БЗ, механизм вывода и пользовательский интерфейс.

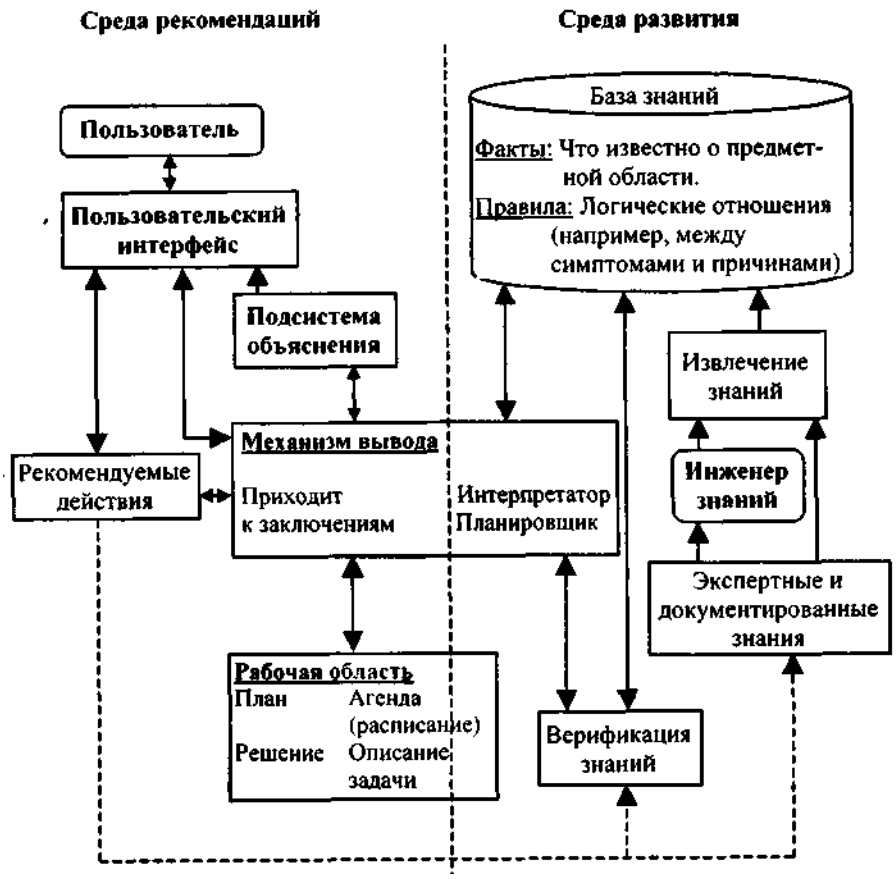


Рис. 4.3. Структура ЭС и ее окружение

Хотя вообще ЭС могут содержать следующие компоненты:

- подсистема извлечения знаний;
- базу знаний;
- механизм вывода;
- пользовательский интерфейс;
- рабочая область;
- подсистема объяснения;
- подсистема верификации знаний.

Обычно, большинство ЭС не содержат подсистему верификации знаний. Существует также большие колебания в содержании и способностях каждой компоненты.

Извлечение знаний представляет собой накопление, передачу и преобразование экспертиз решения задачи от экспертов или документированных источников знаний компьютерной программой для

конструирования или расширения БЗ. Потенциальные источники знаний включают экспертов, учебники, справочники, мультимедийные документы, базы данных (общественные или частные), специальные исследовательские отчеты и информацию, доступную через Интернет.

Извлечение знаний из экспертов является сложной задачей, которая часто создает узкое место при построении ЭС.

Современные условия требуют от инженера знаний и способностей взаимодействовать с одним или более людьми-экспертами при построении БЗ. Инженер знаний помогает эксперту структурировать проблемную область путем интерпретации и объединения ответов человека на вопросы, проводя аналогии, предлагая контрпримеры и выявляя концептуальные трудности.

База знаний содержит знания, необходимые для понимания, формулирования и решения задач. Она включает два основных элемента: факты, такие как проблемная ситуация и теоретические знания о проблемной области, и специальные эвристики или правила, которые направляют использование знаний при решении специфических задач в отдельной области. Кроме того, механизм вывода, тесно связанный с БЗ, содержит стандартные правила решения задач и принятия решений. Эвристики выражают неформальные знания, мнения и суждения в прикладной области. Глобальные стратегии, которые могут быть как эвристиками, так и частью теории проблемной области, обычно включаются в БЗ. Знания, а не просто факты, являются первоначальным необработанным материалом экспертных систем. Информация и знания в БЗ представлены и включены в компьютерную программу путем реализации процесса, называемого представление знаний

Механизм вывода является мозгом ЭС, его также называют управляющая структура или интерпретатор правил (в ЭС, основанных на правилах).

Эта компонента является в основном компьютерной программой, которая обеспечивает методологию для рассуждения об информации в БЗ и в рабочей области, а также для формулирования заключений. Она обеспечивает указания о том, как использовать знания системы при реализации агенды (расписания запланированных действий в рабочей области), которая организует и управляет шагами, предпринимаемыми для решения задачи.

Механизм вывода имеет два главных элемента:

- интерпретатор, который выполняет выбранные позиции агенды, используя соответствующие правила БЗ;
- планировщик, который поддерживает управление агендой. Он оценивает результаты используемых правил вывода в свете их приоритетов или других критериев в агенде.

Пользовательский интерфейс ЭС содержат языковой процессор для дружественного, проблемно-ориентированного общения между пользователем и компьютером. Общение наилучшим образом выполняется на естественном языке. Иногда оно дополняется меню и графикой.

Рабочая область — это область, расположенная отдельно для описания текущей задачи, как определено входными данными. Она также исг пользуется для запоминания промежуточных результатов. В рабочей области запоминаются промежуточные гипотезы и решения. 90

Могут быть запомнены три типа решений: план (как атаковать задачу), агенда (потенциальные действия, ожидающие выполнения) и решение (гипотезы — кандидаты и альтернативные направления действий, которые система сгенерировала до сих пор).

Подсистема объяснения. Способность отслеживать ответственность и соответствие заключений их источникам является решающей и при проведении экспертизы, и при решении задачи. Подсистема объяснения мо-кет отслеживать такую ответственность и объяснять поведение эксперт-рой системы интерактивно отвечая на вопросы.

Подсистема верификации и совершенствования знаний Эксперты обладают способностями верифицировать и совершенствовать знания. То есть, они могут анализировать свои собственные знания и их использование, обучаться от них и улучшать их для будущих консультаций. Аналогично, такая эволюция необходима в компьютеризованном обучении, так, чтобы программа могла анализировать рассуждения под углом зрения их успеха или неудачи. Это может привести к улучшениям, и как результату, более точным БЗ и более эффективному рассуждению. Такой составляющей в настоящее время пока нет в коммерческих ЭС, но она разрабатывается в экспериментальных ЭС.

4.4. Области применения экспертных систем

ЭС могут быть классифицированы несколькими путями. Одним из них является классификация по основным проблемным областям, на которые они ориентированы. При этом проблемные области определяются основными классами задач, эффективно решаемыми методами ЭС. Например, диагностика может быть определена как «выявление неисправностей системы через наблюдения. Диагностика является общей по своей сути деятельностью, совершаемой в медицине, организационных исследованиях, компьютерных операциях, контроле за оборудованием. Основные классы задач, для решения которых создаются экспертные системы, перечислены в таблице 4.2.

Таблица 4.2. Основные классы задач, решаемые ЭС

Классы задач	**Решаемые задачи**
Интерпретация	Выявление описаний ситуации из наблюдений
Предсказание	Выявление похожих последствий в данной ситуации
Диагностика	Выявление неисправности системы через наблюдения
Проектирование	Конфигурирование и разработка объектов, удовлетворяющих определенным требованиям
Планирование	Разработка планов для достижения целей.
Мониторинг	Сравнение наблюдений с планами, сигнализируя об отклонениях и исключениях
Отладка	Выявление и устранение неисправностей

Управление	Интерпретирование, предсказывание, восстановление и мониторинг поведения системы

Некоторые ЭС принадлежат к двум или более из этих категорий. Дадим краткое описание каждой их этих категорий.

Системы интерпретации выявляют описания ситуации из наблюдений. Это категория включает наблюдения, понимание речи, анализ образов, интерпретацию сигналов и многие другие виды интеллектуального анализа. Системы интерпретации объясняют наблюдаемые данные путем присвоения им символических значений, описывающих ситуацию.

Системы предсказания включают прогнозирование погоды, демографические предсказания, экономическое прогнозирование, оценки урожайности, а также военное, маркетинговое и финансовое прогнозирование.

Системы диагностики включают диагностику в медицине, менеджменте, электронике, механике и программном обеспечении. Диагностирующие системы обычно соотносят наблюдаемые поведенческие отклонения с причинами, лежащими в их основе.

Системы проектирования разрабатывают конфигурации объектов, которые удовлетворяют определенным требованиям задачи проектирования. Такие задачи включают конструирование зданий, планировку расположения оборудования и др. Эти системы конструируют различные взаимосвязи описаний объектов друг с другом и проверяют, удовлетворяют ли эти конфигурации установленным ограничениям и требованиям.

Системы планирования специализируются на задачах планирования, например, такой как автоматическое программирование. Они также работают с кратко- и долгосрочным планированием, управлением проектами, маршрутизацией, разработкой продуктов, военными приложениями, производственным и финансовым планированием.

Системы мониторинга сравнивают наблюдения поведения системы со стандартами, которые представляются определяющими для достижения

цели. Эти решающие выявления соответствуют потенциальным недостаткам на предприятии. Существует много компьютерных систем мониторинга: от контроля движения воздушных потоков до задач управления сбором налогов.

Системы управления и контроля адаптивно управляют всеобщим поведением системы. Для осуществления этого система управления должна периодически интерпретировать текущую ситуацию, предсказывать будущее, диагностировать причины ожидаемых проблем, формулировать план устранения этих проблем и осуществлять мониторинг его выполнения для обеспечения успеха.

Не все задачи, которые обычно образуются в каждом из этих классов, подходят для ЭС. Однако есть тысячи задач, которые подходят к этим классам.

Рассмотренные классы задач ЭС, определяющие проблемные области, решаются в различных предметных областях. Области применения существующих на сегодняшний день ЭС охватывают: медицину, геологию, научные исследования в области химии и биологии, военное дело, инженерное дело, космическую технику, метеорологию, экологию, производство, управление процессами, юриспруденцию, маркетинг, финансы, банковское дело и др.

Сегодня ЭС используются многими большими и средними организациями как главный инструмент для улучшения производительности и качества. Они являются также важным инструментом для поддержания стратегических решений и реинжиниринга бизнес-процессов.

4.5. Статические и динамические экспертные системы

При классификации ЭС по проблемным областям на основе классов и типов задач, важно исследовать и оценивать характер проблемной и предметной областей с позиций динамики решаемых задач, важности временного фактора и темпоральной информации.

То есть, если исходная информация о предметной области или окружающем мире, на основе которой решается задача, не изменяется за время решения задачи, то такую предметную область можно условно назвать статической предметной областью, и ее представление в ЭС будет статическим. Если информация о предметной области изменяется за время решения задач, то такую предметную область можно назвать динамической предметной областью

Если задачи, решаемые ЭС, явно не учитывают фактор времени и не изменяют в процессе решения данные о реальной действительности, то это статические задачи. Если задачи при решении требуют учета фактора времени или изменяют данные о реальных внешних процессах, то это динамические задачи.

То есть, ЭС работает в статической проблемной среде, если она использует статическое представление и решает статические задачи. Если ЭС использует динамическое представление или решает динамические задачи, то, соответственно, она работает в динамической проблемной среде. Важность времени в динамических проблемных средах определила название таких ЭС, как систем, работающих в реальном времени.

Среди специализированных систем, основанных на знаниях, наиболее значимыми являются ЭС реального времени или динамические ЭС. Исследования по разработке таких систем с целью их практического использования ведутся достаточно давно, с середины 80-х годов прошлого века.

Следует отметить ряд работ в этой области. В 1985 г. фирма Lisp Machine Inc. (LMI) выпустила систему PICON (Process Intelligent Control — интеллектуальное управление процессом). Система применялась для управления нефтеперерабатывающим предприятием. Система обеспечивает контроль 20 тысяч точек. Имеется возможность динамически изменить программу контроля, концентрировать внимание на «горячих точках»,

параметры которых выходят за рамки допусков. Позже LMI разработала также пакет RTIME.

Успех системы PICON привел к тому, что в 1986 группа ведущих разработчиков системы образовала фирму Gensym, которая, значительно развив идеи, заложенные в PICON, в 1988 г. вышла на рынок с инструментальным средством G2, версия 1.0.

С этого времени работы по созданию инструментальных средств для ЭС реального времени стали вестись более активно. С отставанием от Gensym на 2-3 года другие фирмы начали создавать свои инструментальные средства для ЭС реального времени.

Значимость инструментальных средств реального времени определяется не столько их бурным коммерческим успехом (хотя и это достойно тщательного анализа), но, в первую очередь, тем, что только с помощью подобных средств создаются стратегически значимые приложения в таких областях, как управление непрерывными производственными процессами, аэрокосмические исследования, транспортировка и переработка нефти и газа, управление атомными и тепловыми электростанциями, финансовые операции и многие другие.

Экспертные системы реального времени, решают следующие классы задач:

- мониторинг в реальном масштабе времени;
- обнаружение неисправностей;
- диагностика;
- оперативное планирование и др.

Статические экспертные системы не способны решать подобные задачи, так как они не выполняют требования, предъявляемые к системам, работающим в реальном времени:

- представлять изменяющиеся во времени данные, поступающие от внешних источников, обеспечивать хранение и анализ изменяющихся данных;

- выполнять временные рассуждения о нескольких различных асинхронных процессах одновременно (т.е. планировать в соответствии с приоритетами обработку поступивших в систему процессов);
- обеспечивать механизм рассуждения при ограниченных ресурсах (время, память). Реализация этого механизма предъявляет требования к высокой скорости работы системы, способности одновременно решать несколько задач;
- осуществлять постоянный мониторинг процесса, и при необходимости автоматически запускать механизм логического вывода решений по устранению критических ситуаций с одновременным информированием ЛПР;
- моделировать «окружающий мир», рассматриваемый в данном приложении, обеспечивать создание различных его состояний;
- протоколировать свои действия и действия персонала, обеспечивать восстановление после сбоев;
- обеспечивать наполнение базы знаний для приложений реальной степени сложности с минимальными затратами времени и труда (необходимо использование объектно-ориентированной технологии, общих правил, модульности и т.д.);
- обеспечивать настройку системы на решение задачи (проблемная/предметная ориентированность);
- обеспечивать создание и поддержку пользовательских интерфейсов для различных категорий пользователей;
- обеспечивать уровень защиты информации (по категориям пользователей) и предотвращать несанкционированный доступ.

Специфические требования, предъявляемые к экспертной системе реального времени, приводят к тому, что их архитектура отличается от архитектуры статических систем. Появляются две новые подсистемы: моделирования внешнего окружения и сопряжения с внешним миром

(датчиками, контроллерами и т.п.). Оставшиеся подсистемы подвергаются значительным изменениям.

При создании ЭС реального времени, приобретают важное значение несколько новых по сравнению с обычными ЭС соображений. Главное из них — эффективность исполнения. В обычных ЭС факты и знания, на которых основываются рассуждения, носят статический характер. В производственных системах факты, или показания технологических датчиков, являются динамическими. В таких ЭС может существовать до нескольких тысяч показаний приборов и аварийных сигналов, заметно меняющих величину или состояние в течение нескольких минут.

Задача системы-советника оператора или ЛПР — поставить экспертные диагнозы состояния производства и рекомендовать неотложные аварийные мероприятия или операции по обеспечению экономически оптимальных режимов процесса.

Например, возможны следующие производственные ситуации:

- отказ важного датчика и передача вследствие этого ложной информации. ЭС должна при помощи БЗ о процессе обнаружить противоречия и послать оператору аварийный сигнал;
- нарушение хода процесса. ЭС должна найти причины возникших нарушений, отделить их от следствий, и помочь оператору в устранении неполадок. Для этого она могла бы использовать эвристические правила оптимизации.

В приведенных примерах ЭС работает по правилам экспертизы, заложенным в нее при разработке. Потенциальное преимущество системы — советника оператора — заключается в том, что она проводит экспертизу во всех отношениях достаточно быстро, обеспечивая постоянную организованную помощь оператору.

Таким образом, основными принципами построения ЭС реального времени, в дополнение к требованиям, рассмотренным выше, являются следующие:

- доступ к данным. Необходим эффективный интерфейс передачи данных в реальном масштабе времени между ЭС и распределенной измерительной системой;
- концепция рассуждений. Базовые механизмы прямой и обратной цепочек рассуждений должны быть «встроены в программное окружение, работающее в реальном времени;
- вычислительная эффективность. Эффективность рассуждений зависит от структуры программы и БЗ, а также от быстродействия компьютера. Кроме того, дедуктивные процедуры обычных рассуждений могут быть дополнены эвристическими процедурами, подобными тем, которыми пользуются эксперты.

Тема 5.
Представление знаний в интеллектуальных системах

5.1. Проблемы представления и моделирования знаний

Важное место в теории искусственного интеллекта занимает проблема представления знаний.

Знания представляют собой совокупность сведений (у индивидуума, общества или у системы ИИ) о мире (конкретной предметной области, совокупности объектов или объекта), включающих в себя информацию о свойствах объектов, закономерностях процессов и явлений, правилах использования этой информации для принятия решений.

Первоначально вычислительная техника была ориентирована на обработку данных. Это было связано как с уровнем развития техники и программного обеспечения, так и со спецификой решаемых задач. Дальнейшее усложнение решаемых задач, их интеллектуализация, развитие вычислительной техники ставят задачу создания машин обработки знаний. ***Существенным отличием знаний от данных является их интерпретируемость.*** Если для интерпретации данных необходимы соответствующие программы и сами по себе они не несут содержательной информации, то знания всегда содержательны. Другой отличительной чертой знаний является наличие отношений, например, вида «тип — подтип», «элемент—множество» и т.д. Знания характеризуются наличием ситуативных связей, определяющих ситуативную совместимость отдельных событий и фактов, позволяющих устанавливать причинно-следственные связи.

Типы знаний, которые представляются в системах ИИ, охватывают следующее:

- структуру, форму, свойства, функции и возможные состояния объекта;

- возможные отношения между объектами, возможные события, в которых эти объекты могут участвовать;
- физические законы;
- возможные эффекты действий и состояний, причины и условия возникновения событий и состояний;
- возможные намерения, цели, планы, соглашения и т.д.

Э. Фейгенбаум, в свою очередь, выделяет следующие типы знаний:

- об объектах и категориях окружающего мира;
- о событиях, определяющих временные последовательности и причинно-следственные связи;
- о деятельности, т.е. о способности выполнять какие-либо действия;
- метазнания, т.е. «знания о размере наших знаний или о границах наших способностей.

Можно выделить ряд общих черт для всех систем представлений знаний (СПЗ):

1) все СПЗ имеют дело с двумя мирами — представляемым и представляющим;

2) вместе они образуют основу для представления, если решены следующие вопросы:

- Чем является представляемый мир?
- Чем является представляющий мир?
- Какие аспекты представляемого мира смоделированы?
- Какие аспекты представляющего мира смоделированы?
- Каково соответствие между этими мирами?

Существует также ряд общих для всех СПЗ проблем. К ним можно отнести, в частности, проблемы:

- приобретения новых знаний и их взаимодействие с уже существующими;

- организации ассоциативных связей;
- выбора диапазона в размере элементов представления, связанной с тем, насколько «детально» могут быть описаны объекты и события, и какая часть внешнего мира может быть представлена в конкретной системе;
- неоднозначности и выбора семантических примитивов;
- модульности и понимания;
- явности знаний и доступности;
- выбора соотношения декларативной и процедурной составляющих представления, что влияет на экономичность системы, полноту, легкость кодировки и понимания.

В общем виде модели представления знаний могут быть условно разделены на концептуальные и эмпирические.

Концептуальная модель дает эвристический метод для решения некоторой проблемы. Метод эвристичен, поскольку концептуальное описание не дает гарантии того, что он может быть применен во всех соответствующих практических ситуациях. Концептуальная модель делает возможным распознавание проблемы, позволяет уменьшать время для ее предварительного анализа.

Практическое использование концептуальной модели влечет за собой необходимость преобразования ее в эмпирическую. Знания могут быть накоплены в виде эмпирических моделей, как правило, описательного характера. Эти модели могут варьировать от простого набора правил до полного описания того, как ЛПР решает задачу.

Модели представления знаний можно условно разделить на *декларативные* и *процедурные*.

Декларативная модель представления знаний основывается на предположении, что проблема представления некоей предметной области решается независимо от того, как эти знания потом будут использоваться. Поэтому модель как бы состоит из двух частей: статических описательных

структур знаний и механизма вывода, оперирующего этими структурами и практически независимого от их содержательного наполнения. При этом в какой-то степени оказываются раздельными синтаксические и семантические аспекты знания, что является определенным достоинством указанных форм представления из-за возможности достижения их определенной универсальности.

В декларативных моделях не содержатся в явном виде описания выполняемых процедур. Эти модели представляют собой обычно множество утверждений. Предметная область представляется в виде синтаксического описания ее состояния (по возможности полного). Вывод решений основывается в основном на процедурах поиска в пространстве состояний.

В ***процедурном представлении знания*** содержатся в процедурах — небольших программах, которые определяют, как выполнять специфичные действия (как поступать в специфичных ситуациях). При этом можно не описывать все возможные состояния среды или объекта для реализации вывода. Достаточно хранить некоторые начальные состояния и процедуры, генерирующие необходимые описания ситуаций и действий.

При процедурном представлении знаний семантика непосредственно заложена в описание элементов базы знаний, за счет чего повышается эффективность поиска решений. По сравнению с процедурной частью статическая база знаний у них мала. Она содержит не «неизменные аксиомы, а лишь так называемые «утверждения», которые приемлемы в данный момент, но могут быть изменены или удалены в любое время. Общие знания и правила вывода представлены в виде специальных целенаправленных процедур, активизирующихся по мере надобности.

Процедуры могут активизировать друг друга, их выполнение может прерываться, а затем возобновляться. Возможно использование процедур — «демонов», активизирующихся при выполнении операций введения, изменения или удаления данных.

Средством повышения эффективности генерации вывода в процедурных моделях является добавление в систему знаний о применении, т.е., знаний о том, каким образом использовать накопленные знания для решения конкретной задачи. Эти знания, как правило, тоже представляются в процедурной форме.

Главное преимущество процедурных моделей представления знаний заключается в большей эффективности механизмов вывода за счет введения дополнительных знаний о применении, что однако снижает их общность. ***Другое важное преимущество*** заключено в выразительной силе. Процедурные системы способны смоделировать практически любую модель представления знаний. Выразительная сила процедурных систем (проявляется в расширенной системе выводов, реализуемых в них.

Необходимо отметить, что деление моделей представления знаний на декларативные и процедурные весьма условно, так как в реальных системах представления знаний используются в равной мере элементы и сочетания всех указанных выше форм моделей представления знаний.

5.2. Фреймы

Для представления и описания стереотипных объектов, событий или ситуаций были введены понятия «фреймы», которые являются сложными структурами данных.

Фреймы были впервые предложены в качестве аппарата для представления знаний М. Минским в 1975 г. Согласно его определению, ***фреймы — это минимальные структуры информации, необходимые для представления класса объектов, явлений или процессов***. В общем виде фрейм может быть представлен в виде, показанном на рис. 5.1 и описан строкой:

<ИФ, (ИС, ЗС, ПП),..., (ИС, ЗС, ПП)>,

где *ИФ* — имя фрейма; *ИС* — имя слота; *ЗС* — значение слота; *ПП* — имя присоединенной процедуры.

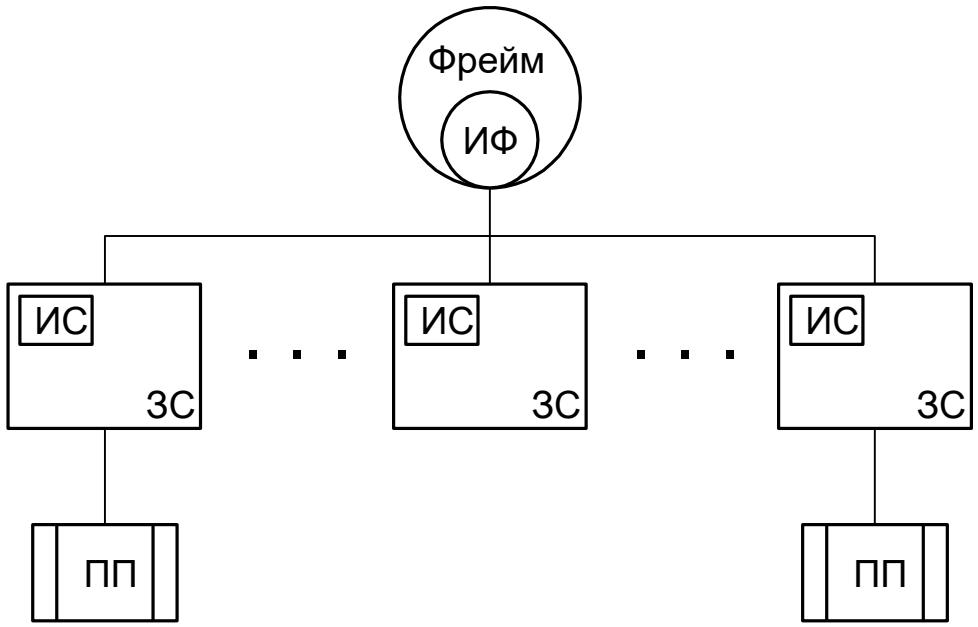


Рис. 5.1. Схема фрейма

Слоты — это некоторые незаполненные подструктуры фрейма, заполнение которых приводит к тому, что данный фрейм ставится в соответствие некоторой ситуации, явлению или объекту.

С каждым фреймом связана информация: как использовать фрейм; что делать, если происходит что-либо непредвиденное; недостающие значения для слотов. Фрейм с заполненными слотами называется экземпляром фрейма. ***Для организации связи между объектами предметной области строится сеть фреймов***. Связь может быть организована путем указания в качестве значений некоторых слотов одного фрейма имен других фреймов.

В качестве данных фрейм может содержать обращения к процедурам (так называемые присоединенные процедуры). Выделяют два вида процедур: процедуры-демоны и процедуры-слуги. ***Процедуры-демоны*** активизируются при каждой попытке добавления или удаления данных из слота (по умолчанию). ***Процедуры-слуги*** активизируются только при выполнении условий, определенных пользователем при создании фрейма.

Для уменьшения информационной избыточности во фреймовых системах реализуют принцип наследования информации, позволяющий

общую (глобальную) для системы информацию хранить в отдельном фрейме, а во всех остальных фреймах указывать лишь ссылку на место хранения этой информации.

Основные свойства фреймов.

1. ***Базовый тип***. При эффективном использовании фреймовой системы, можно добиться быстрого понимания сущности данного предмета и его состояния, однако для запоминания различных позиций в виде фреймов необходимы большие объемы памяти. Поэтому только наиболее важные объекты данного предмета запоминаются в виде базовых фреймов, на основании которых строятся фреймы для новых состояний. При этом каждый фрейм содержит слот, оснащенный указателем подструктуры, который позволяет различным фреймам совместно использовать одинаковые части.
2. ***Процесс сопоставления***. Процесс, в ходе которого проверяется правильность выбора фрейма, называется процессом сопоставления. Обычно этот процесс осуществляется в соответствии с текущей целью и информацией (значениями), содержащийся в данном фрейме. Т.е., фрейм содержит условия, ограничивающие значения слота, а цель используется для определения, какое из этих условий, имея отношение к данной ситуации, является существенным.
3. ***Иерархическая структура***. Фрейм обычно соответствует представлению общего понятия с классификационной иерархической структурой. Особенность иерархической структуры заключается в том, что информация об атрибутах, которую содержит фрейм верхнего уровня, совместно используется всеми фреймами нижних уровней, связанных с ним.
4. ***Сети фреймов***. Если процесс сопоставления не привел к успеху,

возникает необходимость поиска фрейма, подобного предыдущему. Такой поиск, осуществляемый с использованием указателей различия, возможен благодаря соединению фреймов, описывающих объекты с небольшими различиями, с данными указателями и образованию сети подобных фреймов.

5. ***Отношения «абстрактное — конкретное» и «целое — часть»***. Рассмотренная иерархическая структура основывается на отношениях «абстрактное – конкретное», однако помимо такого типа структур существуют и другие, основанные на отношениях «целое – часть».

Отношения «абстрактное – конкретное» характерны тем, что на верхних уровнях расположены абстрактные объекты, а на нижних — конкретные объекты, при чем объекты нижних уровней наследуют атрибуты объектов верхних уровней.

Еще одно отношение «целое — часть» касается структурированных объектов и показывает, что объект нижнего уровня является частью объекта верхнего уровня.

Наибольшее практическое применение во фреймовых системах получили лишь отношения «абстрактное – конкретное». Но в некоторых областях иногда требуется описывать и управлять структурированным объектом. Поэтому в таких случаях не обойтись без обработки отношений типа «целое - часть».

5.3. Семантические сети

Важной схемой представления знаний являются семантические сети. Семантические сети не являются однородным классом схем представления. Имеется лишь несколько общих черт, объединяющих ряд механизмов представления, называемых семантическими сетями. Часто общей основой являются лишь сходство формального обозначения (направленный граф с помеченными вершинами и ребрами) и основной принцип, заключающийся в

том, что элементы знаний должны храниться смежно, если они семантически связаны.

Семантическая сеть - это направленный граф с помеченными вершинами и дугами, в котором вершины соответствуют конкретным объектам, а дуги, их соединяющие, отражают имеющиеся между ними отношения.

Отношения, используемые в семантических сетях, можно разделить на следующие:

- ***лингвистические*** включающие в себя отношения типа «объект», «агент», «условие», «место», «инструмент», «цель», «время» и др.;
- ***атрибутивные***, к которым относят форму, размер, цвет и т.д.;
- ***характеризации глаголов***, т.е. род, время, наклонение, залог, число;
- ***логические***, обеспечивающие выполнение операций для исчисления высказываний (дизъюнкция, конъюнкция, импликация, отрицание);
- ***квантифицированные***, т.е. использующие кванторы общности и существования;
- ***теоретико-множественные***, включающие понятия «элемент множества», «подмножество», «супермножество» и др.

Если имеется конечное множество атрибутов $A=\{A_i, i=\overline{1,n}\}$ и конечное множество отношений $R=\{R_j, j=\overline{1,m}\}$, то под интенсионалом отношения R_j понимают набор пар вида:

$$INT(R_j)=\{...,[A_i, DOM(A_i)],...\},$$

в которых $DOM(A_i)$ означает домен A_i , т.е. множество значений атрибута A_i соответствующего отношения R_j.

Под экстенсионалом отношения R_j понимают множество

$$EXP(R_j)=\{F_1,...,F_p\}$$

где F_k — факт отношения, задаваемый в виде совокупности пар вида («атрибут-значение».

Таким образом, интенсиональная семантическая сеть описывает предметную область на обобщенном, концептуальном уровне, в то время как в экстенсиональной сети производятся конкретизация и наполнение фактическими данными.

Выразительная сила семантических сетей несколько слабее, чем в логике предикатов. В частности, представляет определенную сложность отображение квантификаторов. Некоторые недостатки могут быть устранены с помощью реализации механизма наследования: субконцепции наследуют свойства суперконцепций если только это явно не запрещено.

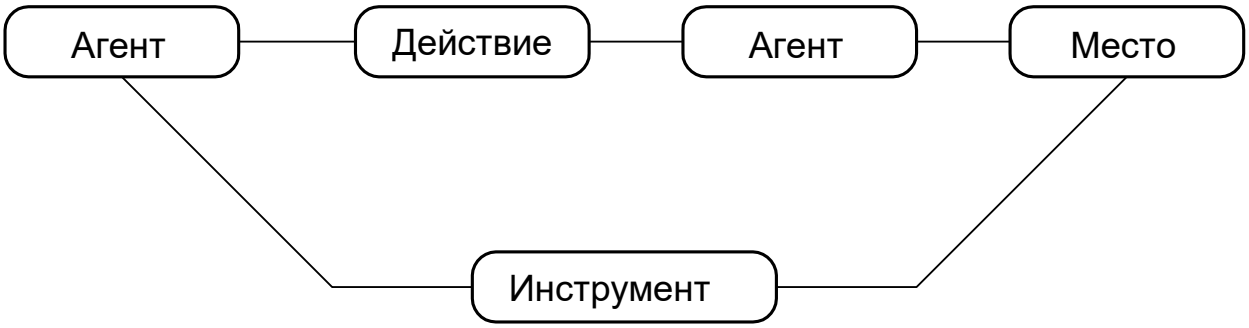
Статические базы знаний, представленные с помощью семантических сетей, могут быть объектом действий, производимых активными процессами. Стандартные операции включают в себя процессы поиска и сопоставления, с помощью которых определяется, представлена ли в семантической модели (и где именно) специфическая информация.

По сравнению с логикой предикатов семантические сети имеют то важное преимущество, что вся точно известная информация о той или иной концепции расположена в базе знаний вокруг соответствующей вершины.

Семантические сети нашли применение в основном в системах обработки естественного языка, частично в вопросно-ответных системах, а также в системах искусственного видения. В последних семантические сети используются для хранения знаний о структуре, форме и свойствах физических объектов. В области обработки естественного языка с помощью семантических сетей представляют семантические знания, знания о мире, эпизодические знания (т.е. знания о пространственно-временных событиях и состояниях).

В качестве примера, рассмотрим представление знаний, содержащихся в высказывании «Поставщик N отгрузил товар склада M автотранспортом. На рис. 5.2 представлена интенсиональная, а на рис. 5.3 — экстенсиональная

семантическая сеть. Факты обозначим овалом, а понятия и объекты прямоугольником.



Инструмент Рис. 5.2. Интенсиональная семантическая модель

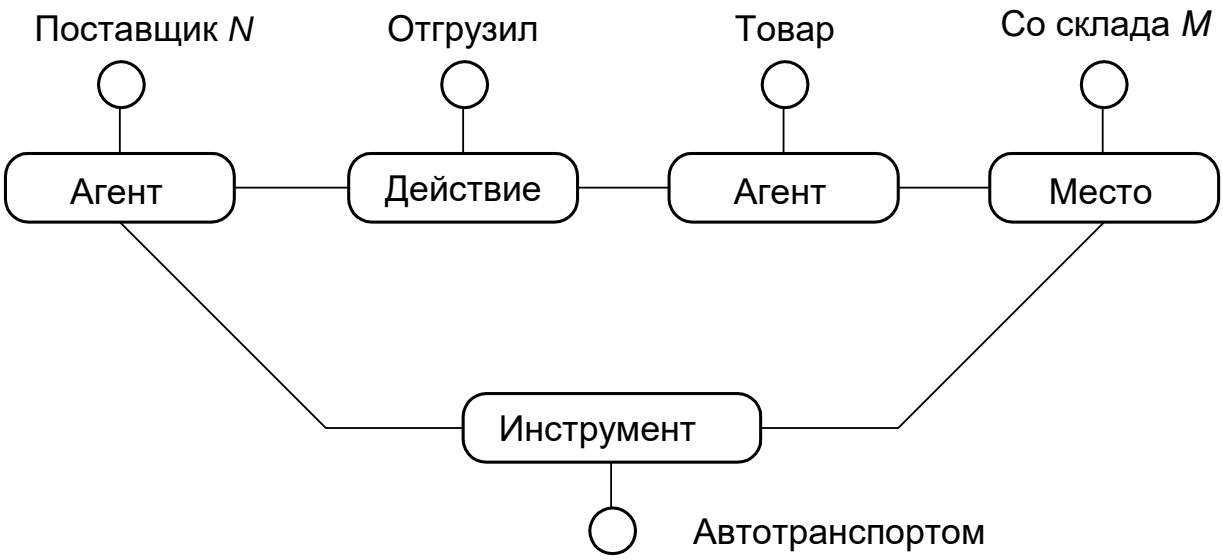


Рис. 5.3. Экстенсиональная семантическая сеть

Тема 6.
Представление знаний в интеллектуальных системах (продлолжение)

5.4. Продукционные модели

Продукционные модели в последнее время широко используются в системах представления знаний.

Продукционные модели могут быть реализованы как процедурно, так и декларативно. Их простота и строгая форма послужили основой ряда интересных свойств, что сделало их удобным средством представления знаний.

Продукционные модели — это набор правил вида «условия — действие», где условиями являются утверждения о содержимом некой базы данных, а действия представляют собой процедуры, которые могут изменять содержимое БД.

В продукционных системах можно выделить три основные компоненты:

- неструктурированная или структурированная БД;
- некоторое число продукционных правил или просто продукций. Каждая продукция состоит из двух частей:
    - условий (антецедент) — в этой части определяются некоторые условия, которые должны выполняться в БД для того, чтобы были выполнены соответствующие действия;
    - действий (консеквент) — эта часть содержит описание действий, которые должны быть совершены над БД в случае выполнения соответствующих условий. В простейших продукционных системах они только определяют, какие элементы следует добавить (или иногда удалить) в БД;

– интерпретатор, который последовательно определяет, какие продукции могут быть активированы в зависимости от условий, в них содержащихся; выбирает одно из применимых в данной ситуации правил продукций; выполняет действие из выбранной процедуры.

Продукционные модели в основном находят применение в качестве решателей или механизмов выводов.

В БД системы хранятся известные факты о некоторой предметной области. Продукции содержат специфические для данной области знания о том, какие дополнительные факты могут быть допущены, если специфические данные найдены в БД.

Действия продукций могут состоять из активных процедур, которые автоматически производят необходимые операции над содержимым БД (либо подобно «демонам» проверять самих себя на предмет того, выполняются ли их условия активации). В этом случае форма представления знаний является процедурной, хотя и в весьма ограниченном виде. В последующих итерациях факты, добавленные в БД, могут подключать (активировать) другие продукции и т.д.

В классических продукционных системах БД представляют собой переменную часть системы, в то время как правила и интерпретатор чаще всего не меняются. Будучи реализованы процедурно, классические продукционные модели обладают весьма привлекательным свойством модульности. Поэтому правила могут быть добавлены или удалены без возникновения неожиданных побочных эффектов. Причина этого заключается также в том, что в классических системах вызов процедур осуществляется только в зависимости от состояния данных; процедуры, как правило, не активируются другими процедурами. Поэтому продукционные системы могут быть с большим успехом использованы для областей знаний, о которых располагаем только некоторым набором независимых правил

(эвристик), а не четкой теорией, вполне завершенной и последовательной, и где поэтому нет алгоритмов, прямо приводящих к цели.

Продукционные системы все более широко используются для реализации продукционных ЭС.

Как правило, классические продукционные системы не содержат сведений о применении, т.е. знаний о том, например, какие продукции использовать для достижения цели. Это ведет к значительному снижению эффективности их работы: хотя в каждой итерации только одна продукция может быть активирована, должны быть проверены условия всех продукций. Для большого числа правил это может потребовать значительного расхода ресурсов. Более того, последовательность выполнения продукций зависит на каждой итерации от состояния всех переменных системы. В этом случае появляется проблема комбинаторного «взрыва.

Для решения этих проблем предлагались подходы, связанные с методами структурного совершенствования БД и условий в продукциях, что позволило бы повысить эффективность функционирования. Предпринимаются также попытки повлиять на ход управления.

В каждом цикле все правила, условия которых удовлетворены содержимым БД, считаются определенными. Если существует несколько таких правил, то вопрос о том, какое из правил выбрать, решается с помощью какой-либо приемлемой стратегии «разрешения конфликтов» (например, выбирается правило с наивысшим приоритетом из заранее определенного перечня приоритетов). Все действия, связанные с выбранным правилом, выполняются и вызывают соответствующие изменения в БД.

Другая возможность заключается в осуществлении точного контроля последовательности выполнении продукций. В простейшем случае продукции могли бы формировать специальные сигналы в БД, которые подключали бы соответствующие продукции в других циклах. В некоторых системах сами продукции могут активировать или дезактивировать другие продукции и даже влиять на работу интерпретатора.

Рассмотрим более подробно некоторые основные формы представления знаний.

В настоящее время разработано множество моделей представления знаний, используемых для реализации систем, основанных на знаниях, в которых знания представлены с помощью правил вида *ЕСЛИ-ТОГДА* (явление - реакция, условие - действие). Систему продукций можно считать наиболее распространенной моделью представления знаний.

Если систему продукций рассматривать как модель представления знаний, то правилам, рассматриваемым с точки зрения человека как средства прямого описания способа логического вывода для решения задач в предметной области, можно придать ясный смысл. При этом отличительной чертой представления знаний с высокой модульностью является простота дополнения, модификации и аннулирования.

Кроме того, со стороны компьютера имеется возможность определения простого и точного механизма использования знаний с высокой однородностью, описанных по одному синтаксису. Эти две отличительные черты, по-видимому, являются причинами столь широкого распространения метода представлений знаний правилами.

5.5. Логические модели представления знаний

Исчисление предикатов

Классическим механизмом представления знаний в системах является исчисление предикатов. В системах, основанных на исчислении предикатов, знания представляются с помощью перевода утверждений об объектах некоторой предметной области в формулы логики предикатов и добавления их как аксиом в систему. Рассмотрим основные положения логики предикатов.

Пусть имеется некоторое множество объектов, называемых ***предметной областью M.*** Знаки, обозначающие элементы этого множества, называют ***предметными константами***, а знак, обозначающий

произвольный элемент этого множества, — ***предметной переменной***. ***Терм*** — это всякая предметная область или предметная константа.

Если f — функциональная n-местная функция, и $t_1, t_2, \ldots, t_n$ — термы, то $f(t_1, t_2, \ldots, t_n)$ есть терм.

Выражение $P(x_1,x_2,\ldots,x_n)$, где $x_i = \overline{1,n}$ — предметные переменные, а P принимает значения 0 и 1, называется *логической функцией* или *предикатом*. Переменные принимают значения из произвольного конечного и бесконечного множества M.

Предикатом или ***логической функцией*** называется функция от любого числа аргументов, принимающая истинные значения 1 и 0. Если в данном выражении заменить x_i на y_i где y_i, — предметные константы, то получим элементарную формулу, т.е. предикатные буквы применимы также и к предметным константам. Элементарные формулы иногда называют атомными. Из элементарных формул с помощью логических связок $\vee$ (или), $\wedge$ (и), $\neg$ (отрицание), $\rightarrow$ (импликация) строят предметные формулы (иногда их называют ***правильно построенными формулами*** — ППФ). ППФ — один из важных классов выражений в исчислении предикатов. Кроме логических связок в рассмотрение вводят кванторы общности $\forall$ или существования $\exists$.

Если P — предикатная формула, а x — предметная переменная, то выражения $\forall xP(x)$ и $\exists xP(x)$ также считаются предметными формулами. В логике предикатов для компактной записи высказываний типа: «для любого x истинно $P(x)$» и «существует такое x, для которого истинно $P(x)$» вводятся две новые дополнительные операции: квантор общности $\forall$ и квантор существования $\exists$. Посредством этих операций приведенные выше высказывания записываются в виде $\forall xP(x)$ и $\exists xP(x)$. Выражение $\forall xP(x)$ обозначает высказывание истинное, когда $P(x)$ истинно при всех $x \in M$ и ложно в противном случае.

Если $P(x)$ в действительности не зависит от x, то выражения $\forall xP(x)$ и $\exists xP(x)$ обозначают то же, что и $P(x)$.

Конкретное вхождение переменной *x* в формулу *P* называется связанным, если оно либо непосредственно следует за каким-либо квантором, либо содержится в области действия некоторого квантора $\forall$ или $\exists$. Вхождение переменной является свободным, если оно не является связанным. В выражении $\forall x P(x,y)$ *x*— связанная, y— связанная.

Связанной переменной называется переменная, если в *P* имеется вхождение этой переменной.

Под *интерпретацией* предикатных формул понимают конкретизацию предметной области, соответствующей данной предметной формуле, и установке соответствия между символами, входящими в предмет, и элементами (а также функциями и отношениями), определяемыми в данной предметной области.

Вывод на предикатах

Выводом системы представления знаний на предикатах являются формулы, выводимые из аксиом с помощью правил вывода. Для организации логического вывода могут использоваться правила.

Для решения конкретной задачи начальное состояние и доступные операторы действий переводятся в формулы исчисления предикатов и добавляются к множеству аксиом. Целевое состояние также выражается формулой и рассматривается как теорема, которая должна быть выведена из аксиом с помощью активного механизма вывода.

Определим основные формы логического вывода.

Индукция — это форма мышления, посредством которой мысль наводится на какое-либо общее правило, общее положение, присущее всем единичным предметам какого-либо класса.

Дедукция— такая форма мышления, когда новая мысль выводится чисто логическим путем (т.е. по законам логики) из предшествующих мыслей. Такая последовательность мыслей называется выводом, а каждая компонента этого вывода является либо ранее доказанной мыслью, либо

аксиомой, либо гипотезой. Последняя мысль данного вывода называется заключением.

Последовательность дедукции определяет «план» того, как достигнуть цели из начального состояния.

Дедукция обычно выполняется с помощью попытки вывести противоречие из получаемого в результате преобразований множества аксиом. Либо для того, чтобы показать, что некоторое множество ППФ неудовлетворимо, надо доказать, что нет такой интерпретации, при которой каждая из ППФ в этом множестве имеет значение 1 (истинно). Хотя эта задача и кажется трудоемкой, существуют довольно эффективные процедуры ее решения. Для выполнения этих процедур требуется представить ППФ данного множества в специальном удобном виде — в виде предложений.

Процесс стандартизации

Любую ППФ исчисления предикатов можно представить в виде предложения, применяя к ней последовательность простых операций. Задача состоит в том, чтобы показать, как придать произвольной ППФ форму предложения. Этот процесс (преобразования ППФ в форму предложения) состоит из следующих этапов:

1) ***исключение знаков импликации***. В форме предложения в исчислении предикатов явно используются лишь связки $\vee$ и $\neg$. Знак импликации можно исключить в исходном утверждении вместо $A \to B$ записать $\neg A \vee B$;

2) ***уменьшение области действия знаков отрицания***. Необходимо, чтобы знак отрицания $\neg$ применялся не более чем к одной предикатной букве;

3) ***стандартизация переменных***, при которой осуществляется переименование переменных с тем, чтобы каждый квантор имел свою переменную. Так, вместо $(\forall x)\{P(x) \to (\exists x)Q(x)\}$ следует написать $(\forall x)\{P(x) \to (\exists y)Q(y)\}$;

4) ***исключение кванторов существования;***

5) ***приведение к предваренной нормальной форме*** (ПНФ), . На этом этапе уже не осталось кванторов существования, а каждый квантор общности имеет свою переменную. Теперь можно перенести все кванторы общности ($\forall$) в начало ППФ и считать областью действия каждого квантора часть ППФ. Про полученную таким образом ППФ говорят, что она имеет ***предваренную нормальную форму*** (ПНФ). ППФ в ПНФ состоит из цепочки кванторов, называемой префиксом, и расположенной за ней формулы, не содержащей кванторов, называемой ***матрицей***;

6) ***приведение матрицы к конъюнктивной нормальной форме*** (КНФ). Любую матрицу можно представить в виде конъюнкций конечного множества дизъюнкций предикатов и (или) их отрицаний. Говорят, что такая матрица имеет КНФ. Заменить $A \vee \{B \wedge C\}$ на $\{A \vee B\} \wedge \{A \vee C\}$;

7) ***исключение кванторов общности***;

8) ***исключение связок.***

Система представления знаний на основе исчисления предикатов имеет ряд достоинств:

- они достаточно хорошо исследованы как формальная система;
- существуют ясные правила, т.е. результаты операций над БЗ также достаточно ясно определены.

Недостатками являются ограниченная выразительность, так как существует большое число факторов, которые тяжело или даже невозможно выразить средствами исчисления предикатов.

Тема 7.
Представление и формализация нечетких знаний

Понятия, которыми оперирует человек в различных областях знаний, являются по своей природе слишком сложными и многоплановыми для того, чтобы использовать для их представления только традиционные, точные, хорошо определенные модели и алгоритмы. Многие понятия вследствие субъективности человеческого мышления, приблизительного характера умозаключений и лингвистического их описания оказываются нечеткими по своей природе и требуют для своего представления соответствующего аппарата. Этот аппарат стал мощным инструментом для решения широкого круга проблем, в которых важное место занимают субъективные, трудно формализуемые знания человека. Особый интерес теория нечетких множеств вызывает в связи с исследованиями и разработками человеко-ориентированных социальных и управленческих систем, в частности, экспертных систем. Рассмотрим основы нечетких множеств.

5.6. Основные определения нечетких множеств

Рассмотрим универсальное множество $U=\{u\}$.

Нечетким подмножеством A на множестве U называется совокупность пар

$$A=\{<\mu_a(u),u>\},$$

где μ_a: $U \rightarrow [0,1]$ — отображение множества U в единичный отрезок $[0,1]$, называемое ***функцией принадлежности нечеткого подмножества*** A.

Значение функции принадлежности $\mu_a(u)$ для элемента $u \in U$ будем называть ***степенью принадлежности***. Переменная u называется ***базовой.***

Интерпретацией степени принадлежности $\mu_A(u)$ является субъективная мера того, насколько элемент $u \in U$ соответствует понятию, смысл которого формализуется нечетким множеством A.

Таким образом, нечеткое множество A области рассуждений U характеризуются ***функцией принадлежности*** μ_A: $U \rightarrow [0,1]$, которая каждому элементу и множества U ставит в соответствие число $\mu_A(u)$ из

отрезка *[0,1]*, описывающее степень принадлежности элемента и подмножеству *A*.

Носителем нечеткого подмножества (далее: множества) *A* называется множество таких элементов *U*, для которых μ_A положительна.

Точкой перехода *A* называется такой элемент множества *U*, степень принадлежности которого множеству *A* равна 0,5.

Пример 7.1. Рассмотрим нечеткое множество A_3, соответствующее нечеткому понятию «небольшой запас деталей на складе». Носителем данного нечеткого множества является конечное множество:

S = {10,11,...,40}, каждый элемент которого представляет собой определенное количество деталей.

A_3*={0.05/10; 0,1/11; 0,2/12; 0,3/13; 0,4/14; 0,5/15; 0,7/16; 0,8/19; 1,0/20;... 1,0/33; 0,9/34; 0,8/35; 0,6/36; 0,4/37; 0,3/38; 0,2/39; 0,1/40}*

Отсюда следует, что в решаемой задаче управления запасами для конкретного ЛПР понятию «небольшой запас деталей на складе» полностью соответствует запас объемом от 20 до 33 деталей, в меньшей степени — запасы от 10 до 19 и от 34 до 40 деталей. Запас объемом меньше 10 и больше 40 деталей понятием «небольшой» охарактеризован быть не может.

Далее для краткости нечеткое подмножество *A* множества *U* будем называть нечетким множеством *A*.

Одноточечным нечетким множеством называется множество, носитель которого состоит из единственной точки. Если *A* — одноточечное нечеткое множество, носителем которого является точка *u*, то записывается это как:

$$A = \mu/u,$$

где μ — степень принадлежности и множеству *A*. Определенное (четкое) одноточечное множество обозначают через *1/u*.

Нечеткое множество можно рассматривать как объединение составляющих его одноточечных множеств. Имея это ввиду, множество *A* можно представить в следующем виде:

$$A = \int_U \mu_A(u)/u$$

где символ $\int$ (интегрирование) обозначает операцию объединения одноточечных нечетких множеств μ_A/u, $u \in U$. Если носитель A состоит из конечного числа элементов, то интегрирование в можно заменить суммированием:

$$A = \mu_1/u_1 + ... + \mu_n/u_n \text{ или } A = \sum_{i=1}^{n} \mu_i / u_i$$

где число $\mu_i \left(i = \overline{1,n}\right)$ — степень принадлежности элемента U_i множеству A. Знак плюс в обозначает объединение, а не арифметическое суммирование.

Пример. Если универсальное множество состоит из чисел от 1 до 10, т.е.

$$U=l+...+10,$$

то нечеткое множество A множества U, описываемое понятием «несколько» можно определить в виде

$$несколько \stackrel{\Delta}{=} 0{,}5/3 + 0{,}8/4 + 1/5 + 1/6 + 0{,}8/7 + 0{,}5/8$$

(символ $\stackrel{\Delta}{=}$ обозначает равенство по определению).

Степень принадлежности к нечеткому множеству может сама представлять собой нечеткое множество.

Пример 7.1. Если есть множество

$$U = Юлия + Анна + Мария + Настя$$

и A — нечеткое множество *«привлекательная»*, то можно написать

«привлекательная» = средне/Юлия +мало/Анна+сильно/Мария+ +мало/Настя.

Нечеткие степени принадлежности *«мало», «средне»* и *«сильно»* являются при этом нечеткими подмножествами полного множества V, определяемого следующим образом:

$$V=0+0,1+0,2+... +0,9+1$$

Сами эти подмножества определяются так:

$$мало=0,.5/0,2+0,7/0,3+1/0,4+0,7/0,5+0,5/0,6$$

средне=0,5/0,4+0,7/0,5+1/0,6+0,7/0,7+0,5/0,8

сильно=0,5/0,7+0,7/0,8+0,9/0,9+1/1

5.7. Операции с нечеткими множествами

Приведем некоторые из основных операций, которые можно осуществлять над нечетким множествами.

1. ***Дополнение*** нечеткого множества обозначается символом $\neg A$ и определяется следующим образом:

$$\neg A = \int_U (1 - \mu_A(u))/u .$$

Операция дополнения соответствует логическому отрицанию. Например, если A — название нечеткого множества, то *«не A»* понимается как $\neg A$.

2. ***Объединение*** нечетких множеств A и B обозначается $A+B$ (или $A \cup B$) и определяется:

$$A + B = \int_U (\mu_A(u) \vee \mu_B(u))/u \qquad (7.1)$$

Объединение соответствует логической связке *«или».* Например, если A и B — названия нечетких множеств, то запись *«A или B»* понимается как $A + B$.

При определении степени принадлежности элементов и новому нечеткому множеству, выбирают большее из μ_A и μ_B.

Замечание: следует иметь в виду, что логическая связка $\vee$ в данном контексте означает по определению max (т.е. $\stackrel{\Delta}{=}$ max); $\wedge$ означает min (т.е. $\stackrel{\Delta}{=}$ min).

3. ***Пересечение*** A и B обозначается $A \cap B$ и определяется следующим образом:

$$A \cap B = \int_U (\mu_A(u) \wedge \mu_B(u))/u \qquad (7.2)$$

Пересечение соответствует логической связке *«и»*, т.е.

$$A \text{ и } B = A \cap B \qquad (7.3)$$

При определении степени принадлежности элементов u новому нечеткому множеству выбирают меньшее из μ_A и μ_B..

4. ***Произведение*** A и B обозначается AB и определяется формулой

$$AB = \int_U (\mu_A(u)\mu_B(u))/u \text{ , если } \alpha \geq 0, \text{ то } \alpha A \stackrel{\Delta}{=} \int_U \alpha\mu_A(u)/u$$

Пример 7.2. Если

$U = 1+ 2 + ...+ 10$

$A=0,8/3+1/5+0,6/6,$

$B=0,7/3+1/4+0,5/6,$

то $\neg A = 1/1+1/2+0,2/3+1/4+0,4/6+1/7+1/8+1/9+1/10,$

$A+B=0,8/3+1/4+1/5+0,6/6$

$A \cap B = 0,7/3+0.5/6$ *(берется min из двух значений* μ*),*

$AB=0,56/3+0,3/6$

$0.4A=0,32/3+0,4/5+0,24/6$

Декартово произведение нечетких множеств $A_1, ..., A_n$ универсальных множеств $U_1,..., U_n$ соответственно обозначается $A_1 \times ... \times A_n$ и определяется как нечеткое подмножество множества $U_1 \times ... \times U_n$ с функцией принадлежности.

$$\mu_{A_1 \times .... \times A_n}(u_1,...,u_n) = \mu_{A_1}(u_1) \wedge ... \wedge \mu_{A_n}(u_n) \qquad (7.4)$$

таким образом

$$A_1 \times ... \times A_n = \int_{U_1 \times .. \times U_n} (\mu_{A_1}(u_1) \wedge ... \wedge \mu_{A_n}(u_n))/(u_1,...u_n) \qquad (7.5)$$

Пример 5. 6. Если

$U_1=U_2=3+5+7,$

$A_1=0,5/3+1/5+0,6/7,$

$A_2=1/3+0,6/5,$ *то*

$A_1 \times A_2 = 0,5/3;3+1/5;3+0,6/7;3+0,5/3;5+0,6/5;5+0,6/7;5$

5.8. Нечеткая и лингвистическая переменные

Нечеткая переменная определяется кортежем

$$\langle X, U, \tilde{X} \rangle$$

|где X — наименование нечеткой переменной,

$U = \{u\}$ — область ее определения или универсальное множество

$\tilde{X} = \underset{u \in U}{\cup} \mu_u / u$ — нечеткое множество на U, описывающее ограничения на возможные числовые значения нечеткой переменной.

Лингвистическая переменная определяется кортежем

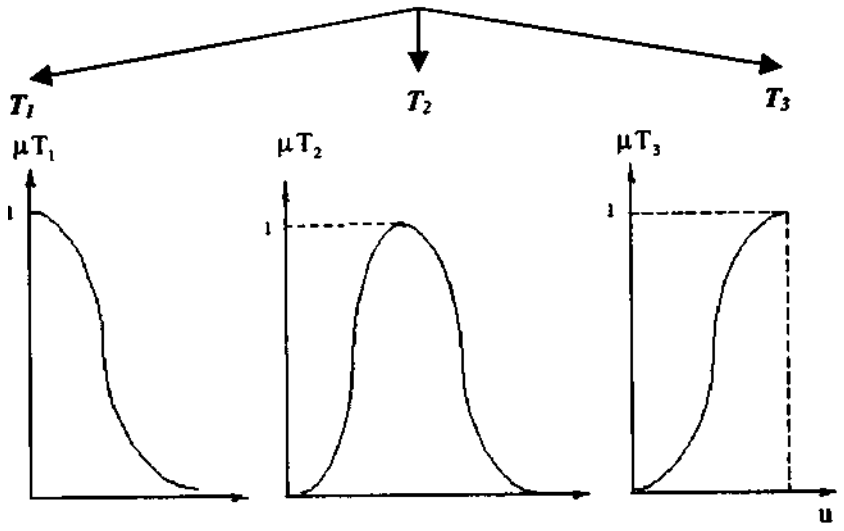
$$\langle X, T, U, G, M \rangle$$

|где X— наименование лингвистической переменной,

T — множество ее значений или ***термов***, представляющее собой наименования нечетких переменных, областью определения каждой из которых является U.

Например,

$T=\{T_1T_2,T_3,\}$, $u_0<u_2<u_1<u_4<u_3<u_t$, $U=[u_0,u_t]$.



u_0 u_1 u u_2 u_3 u u_4 u

Рис. 7.1. Графическое представление распределения терм-множеств лингвистической переменной X

Пару точек (u_0,u_t) будем называть граничной парой. Без особой необходимости не различают переменную и ее наименование. Множество T будем называть ***базовым терм-множеством лингвистической переменной***;

G— ***синтаксическая процедура***, описывающая процесс образования из множества T новых, осмысленных для данной задачи принятия решений значений лингвистической переменной.

Множество $T^*=T\cup G(T)$ назовем расширенным терм-множеством лингвистической переменной.

M — ***семантическая процедура***, позволяющая приписать каждому новому значению, образуемому процедурой G, некоторую семантику путем формирования соответствующего нечеткого множества, т.е. отобразить новое значение в нечеткую переменную. Пример лингвистической переменной.

Пусть ЛПР оценивает посадочную скорость летательных аппаратов с помощью понятий «малая», «небольшая», «средняя», «невысокая». При этом максимальная посадочная скорость равна 300 км/час. Формализация такого описания может быть приведена с помощью лингвистической переменной

< скорость, {малая, небольшая, средняя, высокая}, [0,300], G, M>,

где G — процедура перебора элементов базового терм- множества,

M— процедура экспертного опроса.

5.9. Нечеткие числа и функции

В зависимости от характера множества U лингвистические переменные могут быть разделены на числовые и нечисловые.

Числовой называется лингвистическая переменная, у которой

$$U \subset R^1,\ где\ R^1 = (-\infty, \infty)$$

и которая имеет измеримую базовую переменную.

Нечеткие переменные, соответствующие значениям числовой лингвистической переменной, называются ***нечеткими числами***.

Если $|U| < \infty$, то нечеткие числа будем считать дискретными, если же $|U| = |R^1|$ — то непрерывными. Приведенная выше лингвистическая пере- . менная *СКОРОСТЬ* является числовой, а нечеткие переменные из ее терм-множества— непрерывными нечеткими числами.

Примером нечисловой лингвистической переменной может служить переменная *СЛОЖНОСТЬ,* формализующая понятие «*сложность разработки*», со значениями *НИЗКАЯ, СРЕДНЯЯ, УМЕРЕННАЯ, ВЫСОКАЯ*.

Использование основных понятий лингвистического подхода — лингвистической переменной и нечеткого множества — с целью формализации нечетких описаний элементов задач принятия решений, а именно, критериев, предпочтений ЛПР, случайных подходов, качественных зависимостей между параметрами альтернатив и оценками исходов приводит к необходимости рассмотрения лингвистических критериев и лингвистических отношений предпочтения.

5.10. Лингвистические критерии и отношения предпочтения

Лингвистическим назовем критерий *К*, оценки по шкале которого являются значениями одноименной лингвистической переменной $<K, T(K), U_K, G_K, M_K>$. Согласно этому критерию обеспечивается переход от словесного к числовому описанию лингвистического критерия.

Пример. Рассмотрим критерий прироста прибыли как лингвистический со шкальными значениями *НИЗКИЙ, СРЕДНИЙ, ВЫСОКИЙ* прирост и числовой областью определения $U_k = [10, 100]$ тыс. руб. Примеры функций ринадлежности нечетких множеств, формализующих указанные лингвистические значения критерия для конкретного ЛПР, приведены на рис. 7.2, где величина $\mu_{низкий}(20)$ представляет субъективную оценку того, насколько числовое значение прироста прибыли, равное 20 тыс. руб., соответствует идеальному лингвистическому значению *НИЗКИЙ* прирост.

Рис. 7.2. Примеры функций принадлежности нечетких множеств, формализующие лингвистические значения критерия

Лингвистические критерии, как и соответствующие лингвистические переменные, можно подразделить на числовые или с измеримой базовой переменной, и нечисловые, не имеющие вой переменной. Примером нечислового является критерий профессиональной пригодности со значениями *ХОРОШО, ПЛОХО, НЕДОСТАТОЧНО СООТВЕТСТВУЕТ* В данном случае точно неизвестно как выражается профессиональная пригодность в виде функции тех или иных физических величин.

Тема 8.
Обработка знаний и вывод решений в интеллектуальных системах

5.1. Методы вывода и поиска решений в продукционных системах

5.1.1. Методы вывода на основе прямой и обратной цепочек

При продукционном представлении область знаний представляется множеством продукционных правил *ЕСЛИ — ТОГДА*, а данные представляются множеством фактов о текущей ситуации.

Механизм вывода сопоставляет каждое правило, хранящееся в БЗ с фактами, содержащимися в БД. Когда часть правила *ЕСЛИ* (условие) подходит факту, правило срабатывает и его часть *ТОГДА* (действие) исполняется. Срабатывающее правило может изменить множество фактов путем добавления нового факта, как показано на рис. 5.1. Буквы в БД и БЗ используются для представления ситуаций и понятий.

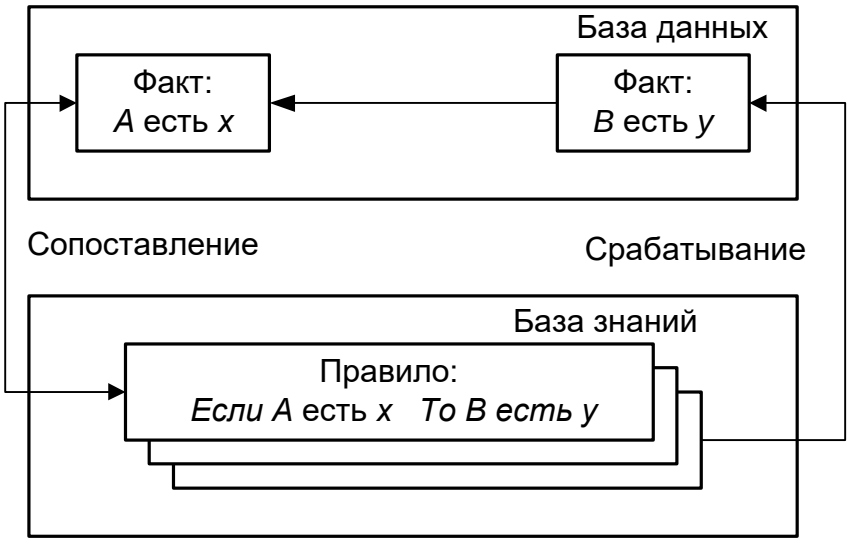


Рис. 8.1. Цикл механизма вывода через процедуру «сопоставление - срабатывание»

Сопоставление частей *ЕСЛИ* правил с фактами создает цепочку вывода. Цепочка вывода показывает как ЭС применяет правила для

получения заключения. Для иллюстрации метода вывода на основе цепочки, рассмотрим простой пример.

Допустим, БД первоначально включает факты A,B,C,D и E, а БЗ содержит только три правила:

Правило 1. Y&D →Z

Правило 2. X&B&E →Y

Правило 3. A → X.

Цепочка вывода на рис. 8.2 показывает, как ЭС применяет правила для вывода факта Z.

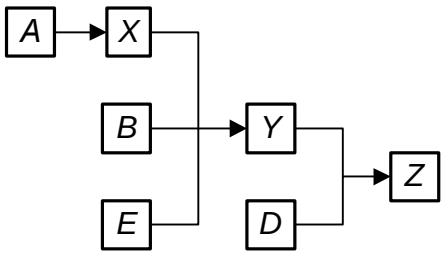


Рис. 8.2. Пример цепочки вывода

Сначала срабатывает Правило 3 для вывода нового факта X из данного факта A. Тогда Правило 2 выполняется для вывода факта Y из первоначально фактов B и E, а также уже известного факта X. И наконец, Правило 1 применяет первоначально известный факт D и только что полученный факт Y для прихода к заключению Z.

ЭС может отразить свою цепочку вывода для объяснения, как было достигнуто отдельное решение; это является основной частью ее объяснительных способностей.

Механизм вывода должен решать, когда правила должны сработать. Существует два принципиальных способа, которыми правила могут быть выполнены. Один называется прямая цепочка (условно-выводимая), а другая обратная цепочка (целе-выводимая).

Рассмотренный пример использует прямую цепочку вывода.

Продукционные системы, в которых сначала анализируется антецедентная часть (условия), имеют так называемую условно-выводимую

архитектуру. Примером экспертной системы такой архитектуры является META-DENDRAL.

Альтернативным типом архитектуры, которая достаточно часто используется в экспертных системах, являются целе-выводимые (действие-выводимые или консеквент-выводимые) продукционные системы. Например, правило вида

A&B&C→D может быть интерпретировано, как «Логическая конъюнкция A, B и C влечет D» или «Чтобы доказать D, необходимо установить A, B, C». В последнем случае цель должна быть достигнута дедуктивным выводом. Для этого исследуются консеквенты правил для нахождения такого правила, которое позволило бы достичь цели. Когда такое правило найдено, проверяются на истинность все его условия. Если условия истинны, продукция активируется. В противном случае продолжается поиск подходящей продукции.

Рассмотрим упрощенный пример продукционной системы с консеквент-выводимой архитектурой. Буквами здесь обозначены элементы БД и они считаются истинными, если содержатся в ней.

БД:A F

Правило 1: A&B&C→D

Правило 2: D&F→G

Правило 3:A&J→G

Правило 4: B→C

Правило 5: F→B

Правило 6: L→J

Правило 7: G→H

Предположим, цель состоит в том, чтобы вывести истинность H. В первую очередь проверяется, находится ли H в БД? Так как в данном случае это не так, то система пытается вывести истинность H, используя правила, имеющие H в правой части. Таким является правило 7. Теперь система пытается вывести истинность G, так как истинность последнего влечет за

собой истинность H. Снова проверяется БД: в БД нет G, следовательно, организуется полис правила, содержащего G в правой части. Таких правил несколько (два). В качестве стратегии «разрешения конфликта будем считать, что правила упорядочены по приоритету, причем правилу с наименьшим номером соответствует больший приоритет.

В данном случае выбирается правило 2, поэтому целью теперь становится вывести истинность D и F. Для этого достаточно показать, что A — истинно (так как находится в БД), B — истинно (согласно правилу 5), C — истинно (согласно правилу 4). Так как истинность D и F доказана, то из правила 2 следует истинность G, а из истинности G — следует истинность H (правило 7). Таким образом цель достигнута. Элементы, истинность которых доказана, добавляются в БД. В данном случае это — элементы H, G, D, C. B. Примерами целе-выводимой архитектуры является MYCIN.

5.2. Общие методы поиска решений в пространстве состояний

Методы перебора. Решение многих задач в интеллектуальных системах можно определить как проблему поиска, где искомое решение — это цель поиска, а множество возможных путей достижения цели представляет собой пространство поиска (или пространство состояний). Поиск решений в пространстве состоит в определении последовательности операторов, которые преобразуют начальное состояние в целевое.

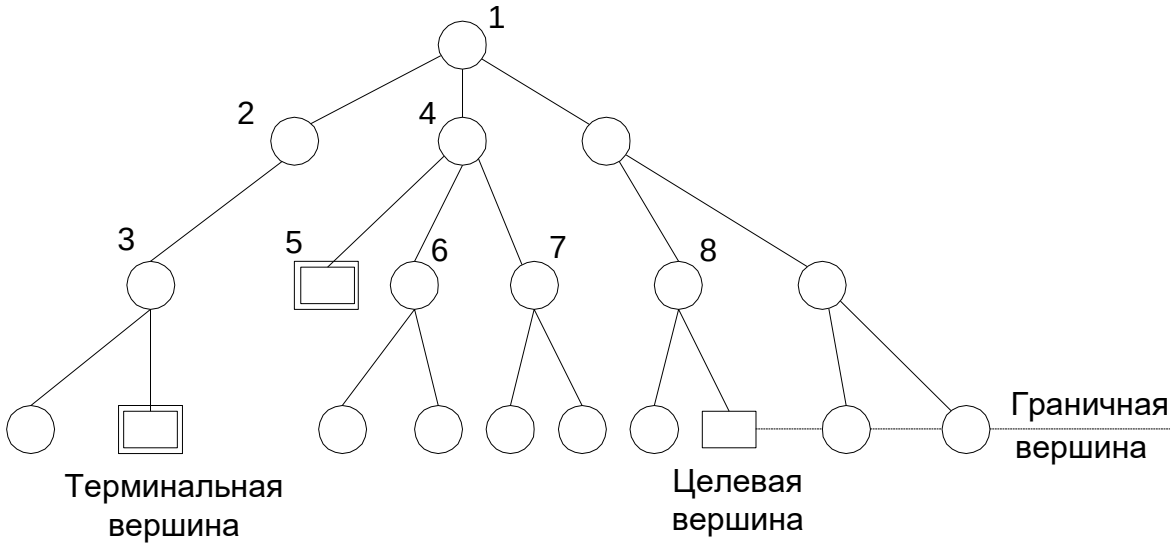
Задачу поиска в пространстве состояний можно сформулировать в общем виде так: пусть исходная задача описывается тройкой (*S*, *F, T*), где *S* — множество начальных состояний; *F* — множество операторов, отображающих одни состояния в другие; *T*— множество целевых состояний. Решение задачи состоит в нахождении последовательности операторов $f_1,f_2,\ldots,f_k$ $(f_i \in F)$, которые преобразуют начальные состояния в конечные. Задача поиска в пространстве состояний описывается с помощью понятий теории графов. Задается начальная вершина *S* (исходное состояние) и множество целевых вершин *T*. Для построения пространства состояний используются

операторы *F*. Последовательно от корня к вершинам дерева применяются операторы, относящиеся к этому уровню, для построения вершин-преемников следующего уровня (раскрытие вершин). Дуги идентифицируются как операторы преобразования. Получаемые вершины — преемники проверяются: не являются ли они целевыми. Далее переходят к следующему уровню. Терминальные вершины — это те, к которым нельзя применить никаких операторов. Раскрытие продолжается до тех пор, пока не получена целевая или терминальная вершина. Поиск на графе состояний — это процесс построения графа *G*, содержащего целевую вершину. Этот граф называется графом поиска. Различают нескольких вариантов реализации метода перебора, которые отличаются заданным порядком, в котором будут перебираться вершины.

Поиск в глубину. При поиске в глубину прежде всего раскрывается та вершина, которая имеет наибольшую глубину. Из вершин, расположенных на одинаковой глубине, выбор вершины для раскрытия определяется произвольно. Для сдерживания возможности следования по бесперспективному пути вводится ограничение на глубину. Вершины, находящиеся на граничной глубине, не раскрываются.

Поиск в ширину. Вершины раскрываются в последовательности их порождения. Поиск идет по ширине дерева, так как раскрытие вершины происходит вдоль одного уровня. Целевая вершина выбирается сразу же после порождения. При поиске в ширину возможно нахождение наиболее короткого пути к целевой вершине, если такой путь есть. На рис. 8.3 представлены графы поиска, построенные при поиске в глубину и в ширину.

а)



б)

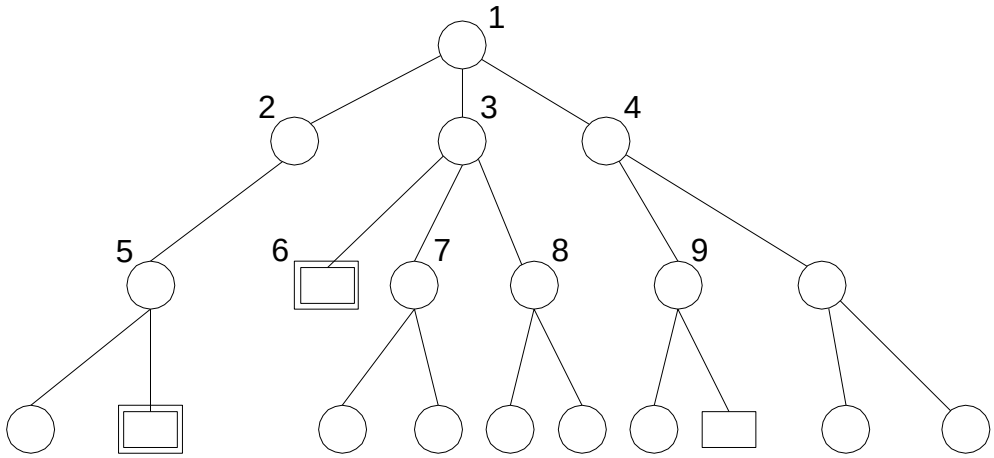


Рис. 8.3. Графы поиска, построенные при поиске в глубину (а) и ширину (б)

Поиск на основе стоимости дуг. Во многих случаях дугам ставят в соответствие некоторую стоимость, чтобы внести оценку для использования соответствующего правила. При поиске целевой вершины стремятся найти путь минимальной стоимости. Раскрытие вершин производится в порядке возрастания их стоимости. Для каждой вершины нужно помнить минимальную стоимость пути, построенного от начальной вершины до нее.

Поиск с возвратом (бэктрекинг). При реализации такого поиска при выборе правила определяется точка возврата, т.е. если дальнейший поиск в выбранном направлении приведет к сложностям или будет

бесперспективным, то осуществляется переход к точке возврата, пройденной на ранних этапах поиска. Далее применяется другое правило, и процесс поиска продолжается. Здесь все неудачные итерации, приведшие к тупиковой ситуации, забываются, как только применяется новое правило, и переходят к другому направлению поиска. Такой метод поиска с возвратом называется хронологическим возвратом. Он часто малоэффективен, так как не запоминает неудачные состояния и шаги поиска, встретившиеся на некотором пути. Многие из них впоследствии окажут свое отрицательное влияние при реализации других путей. Таким образом, много полезной информации отбрасывается и не используется при анализе дальнейших направлений поиска. Необходимо анализировать шаги вывода, приведшие к тупиковым ситуациям и ошибкам. Для этого надо запоминать шаги вывода. Кроме того, возвращаться надо не к точке возврата, предшествующей данному состоянию, а к состоянию, которое вызвало неудачный ход поиска.

Для методов поиска решения при представлении пространства состояний в виде графа характерно то, что в них в основном предусматривается запоминание результатов применения нескольких последовательностей правил. Другая особенность — они работают в пробном режиме, т.е. при выборе и использовании применимого правила для какой-либо ситуации предусматривается возможность возврата к этой ситуации для применения другого правила. Достоинствами методов перебора является достаточно простая их реализация и возможность в принципе находить решение, если оно существует. Однако на практике всегда есть ограничения по времени и объему памяти на процесс реализации поиска. Для больших пространств и сложных массивов задач методы перебора неприемлемы — существует реальность комбинаторного взрыва. Для сокращения поиска необходима информация, называемая эвристической.

Эвристические методы поиска. Эти методы поиска возможно использовать тогда, когда располагают некоторыми эмпирическими правилами, которые позволяют сокращать объем просматриваемых

вариантов решений. Эвристическая информация основывается на опыте, здравом смысле, допущениях разработчика.

При использовании эвристических методов поиска открытые вершины стремятся упорядочить таким образом, чтобы процесс поиска распространился в наиболее перспективных направлениях. Для определения направления поиска используется некоторая мера, характеризующая перспективность вершины или пути, где эта вершина находится. Эту меру называют *оценочной функцией f(n)*. Эта функция является оценкой стоимости кратчайшего пути из начальной вершины в целевую при условии, что он проходит через вершину *n*. При раскрытии вершины или определении пути выбирается вершина с минимальным значением оценочной функции. Оценочная функция должна адекватно характеризовать пространство поиска, т.е. необходим достаточно большой объем знаний о проблемной области и тщательный анализ пространства состояний. На практике использование количественных характеристик и весовых коэффициентов для представления этих знаний себя не оправдывает, так как применение не позволяет эффективно вести поиск решений (могут потребоваться большие объемы вычислений). Кроме того, эвристический поиск с использование оценочной функции предполагает достоверное знание пространства состояний. Однако в реальной практике при принятии решений сталкиваются с фактами и знаниями недостаточно полными и определенными. Кроме того, часто на процесс поиска влияет дефицит времени. В этих условиях люди используют методы, отличные от формального математического рассуждения. Формальное математическое рассуждение является монотонным, т.е. каждое заключение следует из предыдущего. (Монотонность — свойство некоторых логических и математических операций (функций), которое, говоря обобщенно, состоит в том, что направление возможного изменения результата операций зависит только от направления изменения того, над чем эти операции производятся.)

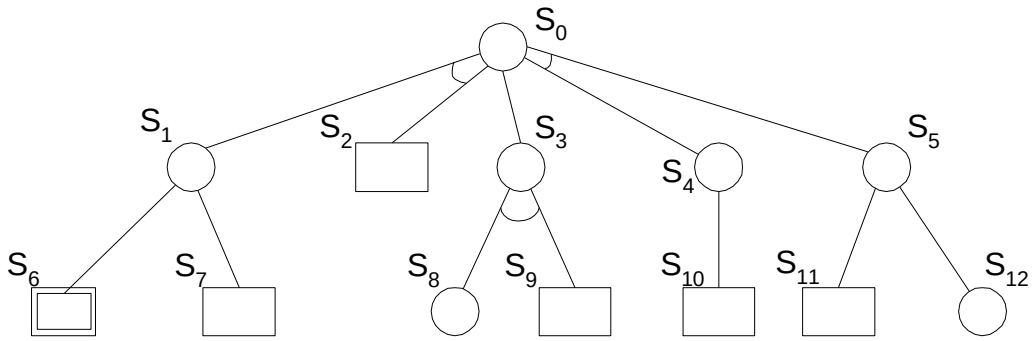
Специалисты, принимающие решения, используют немонотонные рассуждения, или рассуждения здравого смысла (основывающиеся на общих элементарных знаниях). Несмотря на то, что рассуждения здравого смысла являются довольно обыденными для людей, очень трудно достигнуть требуемого уровня реализации подобных рассуждений в ИИ. (Тем не менее, в некоторых классических ЭС, таких как MYCIN, PROSPECTOR, этот метод довольно успешно реализован для сокращения пространства поиска.)

При рассуждениях здравого смысла процедура поиска строится на некоторых предположениях при отсутствии информации, противоречащей этим предположениям. Предположения могут изменяться при поступлении дополнительной проясняющей информации, т.е. в системах поиска, основывающихся на предположениях, необходим просмотр предположений о характере ситуации и направлении поиска при получении новых фактов и знаний. Необходим также пересмотр выводов, полученных на основании этих предположений (совмещение с процедурами возврата).

Метод редукции. Поиск необходимой совокупности данных для решения задачи сводится к решению составляющих подзадач. Задачи описываются различными способами: списки, деревья, массивы. Рассмотрим форму описания задачи как поиск в пространстве состояний. В данном представлении подзадачи рассматриваются как задачи нахождения связи определенными состояниями в пространстве поиска. Для представления входной задачи в виде совокупности подзадач используется оператор перехода к новому описанию. Этот оператор преобразует исходную задачу таким образом, что при решении всех подзадач - преемников обеспечивается решение исходной задачи. Для каждого представления исходной задачи может существовать некоторое множество таких операторов, каждой из которых порождает свою совокупность подзадач. Часть этих подзадач может оказаться неразрешимой. Для других подзадач процесс повторяется, т.е. их в свою очередь, также сводят к подзадачам. Это продолжается до тех пор, пока каждая подзадача не будет иметь очевидное решение.

Процесс преобразования также удобно описывать с помощью графовых структур. Процесс поиска решения исходной задачи при таком описании представляет собой направленный граф редукции задач. Этот граф называется графом *И/ИЛИ*. Вершины этого графа представляют описания задач и подзадач. Граф *И/ИЛИ* содержит вершины двух типов. Тип *«И»* - соответствует задаче, решаемой при условии реализации всех ее подзадач в соответствующих вершинах - преемниках. Тип *«ИЛИ»* — соответствует задаче, решение которой возможно получить при решении одной из альтернативных подзадач в соответствующих вершинах - преемниках. На рис. 8.4 представлены фрагменты графов типа И/ИЛИ.

а)



б)

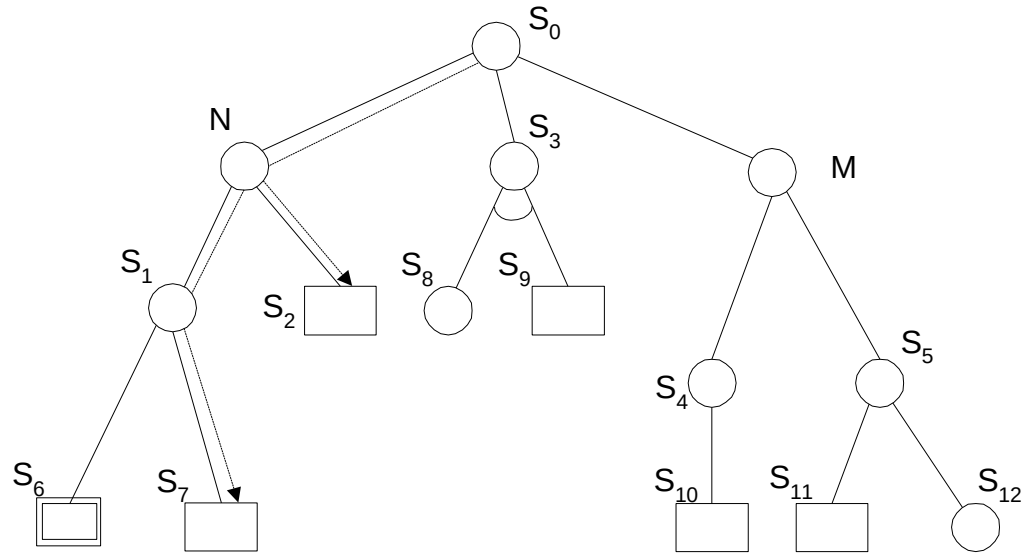


Рис. 8.4. Представление разбиения исходной задачи на подзадачи в виде графа *И/ИЛИ* (а) и преобразование графа *И/ИЛИ* (б)

Исходная задача S_0 разбивается на группы подзадач. Она может быть решена путем решения подзадач либо S_1 и S_2; либо S_3; либо S_4 и S_5. Вершины N, S_3, M, S_5 — это вершины типа *«И»*. Штриховой линией показан вариант решающего графа исходной задачи. Решения подзадач S_2, S_7, S_9, S_{10}, S_{11} — предполагают известными. Решения других подзадач неизвестны. В структуру обычно вводятся дополнительные вершины, чтобы каждое множество подзадач формировалось под своей собственной родительской вершиной. Вершина графа *И/ИЛИ* может принадлежать к типу *И* либо *ИЛИ*. Поэтому исходный граф преобразуется и вводятся дополнительные вершины N и M, которые служат отдельными родительскими вершинами для подзадач $\{ S_1 , S_2\}$ и $\{S_4 , S_5\}$. Таким образом, вершина S_0 преобразуется в вершину *ИЛИ*.

Реализация графа редукции аналогична реализации графа поиска ре-нений в пространстве состояний. В частном случае, если вершин *И* нет, получается обычный граф пространства состояний. Поэтому метод редукции является в какой-то степени обобщением подхода с использованием пространства состояний.

Процесс поиска на графе *И/ИЛИ* заключается в построении решающего графа (или дерева решений), который является подграфом графа редукции.

Литература

1. Ясницкий Л. Н. Искусственный интеллект: учеб. пособие - Москва: БИНОМ. Лаборатория знаний, 2011.
2. Б. Я. Советов, В. В. Цехановский, В. Д. Чертовской Представление знаний в информационных системах: Москва: Издательство "Академия", 2012
3. Долженко, А.И. Нечеткие модели – эффектвный инструмент для анализа потребительского качества информационных систем: монография/ А.И. Долженко; Ростовский государственный экономический университет «РИНХ». – Ростов-н/Д., 2008.
4. Гаскаров Д.В. Интеллектуальные информационные системы. Учеб. Для вузов. – М.: Высш. Шк., 2003.
5. Александров А. CMDB: досье для управления ИТ / Открытые системы, №10, 2006, С.29 – 35.
6. Кожухов А. Управление непрерывностью ИТ-услуг / Корпоративные системы, №9, 2006 // http://www.iemag.ru/?ID=608550
7. ITSM Reference Model / http://h20219.www2.hp.com/services/cache/78360-0-0-225-121.aspx
8. Колесов А. HP ITSM и эффективность обслуживания информационных систем предприятий / http://www.bytemag.ru/?ID=602758
9. Албахари, Дж. C# 3.0 Справочник: Пер. с англ./ Дж. Албахари, Б. Албахари. – 3-е изд. – СПб.:БХВ-Петербург, 2009.