ИНФОРМАТИКА

В.П. Зубков
кандидат технических наук, доцент,
А.Н. Назарычев
доктор технических наук
Ивановский филиал СГА

К вопросу о построении плана решения задачи

В статье рассматриваются подходы программного построения плана решения задачи. Это позволяет автоматизировать решение задач с помощью программных систем.

Рассмотрим несколько определений понятия «задача»:

— задача [1] — это необходимость сознательного поиска соответствующего средства для достижения ясно видимой, но непосредственно недоступной цели. Решение задачи означает нахождение этого средства;

— задача [2] — требование определить математический объект, удовлетворяющий заданным условиям;

— задача [3] — это то, что требует исполнения, разрешения.

В приведенных трех определениях понятие «задача» рассматривается с функциональной точки зрения, без определения формы ее изложения. Для использования формулировки задачи в виде некоторого текста программами, умеющими анализировать этот текст, необходимо выделить формы пред-

ставления содержания задачи. Задачи рассматриваются для естественных и технических систем. Можно различать задачи формализованные и неформализованные. В текстах формализованных задач используются языки формальных систем. Тексты неформализованных задач излагаются на научном языке или языке деловой прозы. Такие тексты могут содержать части текстов на формализованном языке.

Человек может решать задачу, основываясь на представлениях, на приобретенном опыте, логическим путем. Когда человек опирается на приобретенный опыт при решении задач — это означает, что происходит обучение. Чтобы логическим путем решить задачу, необходимо обладать знанием. При решении задачи человек рассуждает. Программа при решении задачи, сформулированной на естественном языке, может использовать накопленные и первоначальные знания, а также логические пути решения.

Когда говорят о структуре задачи, то имеют в виду что-то неизменное, присущее многим задачам. Обычно под структурой задачи понимают такие ее неизменные части, как условия, ограничения, цели. Условия описывают данные для решения задачи. Условие — это то, от чего зависит нечто другое. Условие заключает в себе как отношение причины и следствия, так и отношение основания и вывода. Ограничения ставят границы для рассматриваемых данных и результатов, то есть когда задаются ограничения, то не все данные и получаемые результаты учитываются. Цели описывают то, что надо получить в данной задаче. Цель определяет путь деятельности. Цель есть представление, которое надо достигнуть. Для формального рассмотрения структуры задачи введем следующие обозначения: У — условие (символ может задаваться с индексами); О — ограничение; Ц — цель. Символы О и Ц также могут быть с индексами.

При любом рассмотрении структуры задачи возникает вопрос: как выделить в тексте задачи условия, ограничения

и цели. При обработке текста задачи программной системой анализа естественного языка (АЕЯ) на выходе получается семантическая сеть понятий и терминов, содержащихся в тексте задачи. Возникает проблема идентификации структуры текста задачи, а также отыскания точек начала идентификации. Другими словами, необходимо ответить на вопрос, когда происходит наиболее эффективное выделение структуры: в исходном тексте задачи; в сформированной семантической сети после работы АЕЯ; в комплексном определении, используя промежуточные представления.

Рассмотрим определения понятия «план»: план [2] представляет собой систему действий, удовлетворяющих тем или иным условиям, зависящим от конкретной задачи; план [3] — это заранее намеченная система мероприятий, предусматривающая последовательность и сроки выполнения работ.

Согласно приведенным определениям план представляет собой описание последовательности действий, приводящих к решению задачи.

Определения понятия «решение» следующие: решение [2] — это процесс отыскания решения; решение [3] — это заключение, вывод из чего-либо.

Любая задача имеет условия и цели и может иметь ограничения. Значит, план решения — это крупноблочное представление последовательности действий для достижения цели от условий при удовлетворении заданных ограничений. Для составления плана решения необходимо понимание смысла задачи. Понимание [3] возможно только тогда, когда предмету, на который оно направлено, придано смысловое содержание. При понимании или интерпретируется смысл телесно-вещественного (герменевтика, физиогномика), или происходит дальнейшее осмысление того, что уже имеет некоторый смысл (гномика). Понимание [4] — это процесс нахождения существенных признаков и связей исследуемых предметов и явлений, вычленение их из массы несущественного, случайного на осно-

ве анализа и синтеза, применения правил логического умозаключения, установления сходства и различия, причин, вызвавших появление данных объектов и их развитие, сопоставления полученной информации с имеющимися знаниями.

Для программы понимание соотносится с формой представлений знаний и методами их обработки. Знание о предметной области задачи хранится в основных семантических сетях: семантической сети определений (СемСО); классификационной семантической сети (СемКлС); каузальной семантической сети (СемКауС); семантической функциональной сети (СемФС); семантической сети задачи первого уровня (СемСЗ1); семантической сети задачи второго уровня (СемСЗ2); семантической сети плана решения задачи (СемСПРЗ), семантической сети действий (СемСД); семантической сети объектов (СемСОб).

Дополнительно знания хранятся в следующих справочниках: справочнике ассоциаций (СПРАСС); справочнике по анализу естественного языка (СПРАЕЯ); справочнике модальных глаголов (СПРМГ); справочнике признаков целевого понятия (СПРПЦП); справочнике сокращений (СПРСКР).

Семантическая сеть плана решения задачи (СемСПРЗ) является семантической функциональной сетью, отражающей предметную область для текста данной задачи.

Кроме рассмотренной общепринятой классической структуры задачи различают функциональную структуру задачи, логическую структуру задачи и прочие структуры.

Смысл [4] — это конкретизация значения предмета в речевом или непосредственно деятельностном соотнесении его со значением слов или с предметной ситуацией. Средством выражения смысла являются знаковые системы, которые изучает семантика. Кроме того, семантика [5] означает определенные общие отношения между символами и объектами, представленными этими символами, и изучает свойства семантических отношений, под которыми понимаются отношения, определяемые сигнификативной функцией. Сигнификативная функция

[6] материального объекта состоит в таком соотнесении материального объекта с другими объектами, которое используется для переработки информации об этих объектах в целях осуществления процессов ориентации в окружающей среде или в целях управления.

Текст задачи на естественном языке представляет собой совокупность знаков, которые выражают определенный смысл. Текст задачи может быть переработан человеком в формальное представление, которое может быть понято человеком или программой, предназначенной для решения данного класса задач. Способы задания смысла задачи показаны в таблице.

Таблица

Способы задания смысла задачи

№ п/п	Форма представления задачи	Составитель задачи	Адресат, понимающий смысл задачи	Возможность дальнейшего уточнения
1	Текст на естественном языке	Человек	Человек, программная система	Преобразование в формальное представление первого уровня
2	Графическое представление	Человек, программная система	Человек, программная система	Преобразование в формальное представление первого уровня
3	Звуковое представление	Человек	Человек, программная система	Преобразование в формальное представление первого уровня
4	Смешанное представление	Человек, программная система	Человек, программная система	Преобразование в формальное представление первого уровня
5	Формальное представление i-го уровня (i≥1)	Человек, программная система	Человек, программная система	Преобразование в формальное представление (i+1)-го уровня (i ≥1)

Для раскрытия смысла задачи используются разные виды семантик. Можно выделить денотационную, алгебраическую, аксиоматическую, операционную, трансформационную и ряд других семантик. Наиболее часто используются денотационная, алгебраическая, аксиоматическая, операционная семантики.

Семантика [7] — часть определения языка, касающаяся указания смысла и действия текста, составленного в соответствии с синтаксическими правилами этого языка.

Денотационная семантика в основном используется для определения смысла языков программирования. Согласно денотационной семантике смысл программы, написанной на некотором языке программирования, задается некоторой оценочной функцией, которая присваивает каждой законченной синтаксической конструкции языка определенное абстрактное значение. Оно может быть либо числом, либо значением истинности, либо функцией. Оценочные функции по своей природе могут быть композиционными или рекурсивными. Значение программы определяется как функция значений, соотнесенных с отдельными синтаксическими элементами.

В операционной семантике используется концепция абстрактной машины, характеризуемой своим состоянием и совокупностью элементарных команд, которые она способна выполнять. Абстрактная машина определяется путем указания, каким образом компоненты состояния изменяются под воздействием каждой из команд. При этом не предполагается, что абстрактная машина должна быть моделью реальной машины. Имеется в виду упрощенный ее вариант, с тем чтобы язык мог быть определен однозначно и не было никаких неоднозначностей в интерпретации примитивных команд и кодов операций.

Семантическое описание языка программирования определяет правила перевода его выражений на язык кодов операций. В отличие от денотационной семантики, в операционной семантике нет гарантии того, что смысл всей программы будет

определен смыслом ее частей. Примером операционной семантики является описание возможностей языка ПЛ/1 с использованием определений Виена, а также описание любого другого алгоритмического языка с помощью машин Тьюринга, Поста или любых аналогичных конструкций.

Алгебраическая семантика подчеркивает алгебраическую структуру как синтаксических, так и семантических объектов. Как правило, синтаксические объекты выражаются в виде элементов некоторой начальной алгебры. Тогда отображение синтаксиса на семантику есть однозначно определенный гомоморфизм, соответствующий некоторой интерпретации примитивных символов. Особенностью такого подхода является доскональное изучение свойств программ на чисто синтаксическом уровне или в более общем случае свойств, зависящих только от некоторых точно сформулированных допущений относительно диапазона возможных интерпретаций. Алгебраическая семантика является некоторой детализацией денотационной семантики.

В частности, для раскрытия смысла задачи можно использовать семантические сети. Под семантической сетью [5] подразумевают систему знаний, имеющую определенный смысл в виде целостного образа сети, узлы которой соответствуют понятиям и объектам, а дуги — отношениям между объектами. В базе знаний имеются следующие виды семантических сетей: семантическая сеть задачи первого уровня (СемС31); семантическая сеть определений (СемСО); каузальная семантическая сеть (СемКауС); семантическая классификационная сеть (СемКлС); семантическая сеть задачи второго уровня (СемС32); семантическая функциональная сеть (СемФС).

Прежде чем описать машинное понимание смысла задачи, рассмотрим более подробно понятия: «задача», «смысл задачи», «понимание» — применительно к контексту программы, размещенной в ЭВМ. С одной стороны, любая про-

грамма, составляемая на ЭВМ, всегда предназначена для решения некоторой прикладной задачи. С другой стороны, разработка программы также представляет задачу для программистов, которые должны представлять и прикладную область, и методы разработки программ. В свою очередь, при разработке программ возникает тоже целый ряд задач. Можно сделать предположение, что при решении проблемы создания прикладных программ появляется множество иерархически упорядоченных наборов задач. Некоторые из этих иерархически упорядоченных наборов задач представляют более легкий путь разработки и решения, а другие – более сложный. При решении любой задачи используются такие универсальные подходы, как декомпозиция и абстракция. При декомпозиции задачи [8] она разбивается на ряд подзадач таким образом, чтобы каждая подзадача имела один и тот же уровень рассмотрения; каждая подзадача могла быть решена по возможности независимо от других; полученные решения могли быть объединены вместе для решения первоначальной задачи. Декомпозиция является способом, значительно экономящим время решения задачи. Декомпозиция не является некоей панацеей, при неграмотном использовании можно не понять задачу или не получить ее решение. Большие или плохо понимаемые задачи поддаются декомпозиции со значительными трудностями. При использовании декомпозиции наиболее распространенной является проблема, когда создание отдельных компонентов, которые соответствуют некоторой проведенной ранее декомпозиции, не приводит к тому, чтобы объединение этих компонентов позволяло решать исходную задачу. Решения, полученные для выделенных подзадач, могут не дать результата для всей задачи, если она была разделена неудачно.

Абстракция представляет собой эффективный способ декомпозиции, осуществляемый посредством изменения списка

детализации. При абстрагировании от проблемы игнорируются подробности, чтобы свести задачу к более простой. Задача абстрагирования и последующей декомпозиции типична для процесса создания программы: декомпозиция используется для разбиения программы на компоненты, которые могут быть потом объединены для решения основной программы; абстрагирование предполагает выбор компонентов. Выполнение то одного, то другого процесса до тех пор, пока не получится разбиение исходной задачи на множество подзадач, решение которых известно, — суть применения абстракции и спецификации.

Процесс абстракции может быть рассмотрен как некоторое обобщение. Он позволяет забыть об информации и рассматривать различные предметы так, как если бы они были эквивалентны. На рис. 1 показаны два фрагмента программы, записанных на языке высокого уровня. На уровне абстракции, определенном использованным языком высокого уровня, приведенные фрагменты отличаются друг от друга. Если e присутствует в a, то первый фрагмент отыскивает индекс его первого вхождения, а второй — индекс последнего вхождения. Первый фрагмент устанавливает i в highbound(a) + 1, а второй — в lowbound(a) — 1. Оба фрагмента написаны для выполнения одной и той же функции: установить в f значение false, если e отсутствует в a и установить в значение true, если e присутствует в a, переменная z содержит индекс вхождения e в a. Для выполнения описанного требования оба фрагмента программы не находятся на требуемом уровне абстракции.

Можно пойти по пути создания языка высокого уровня, содержащего абстракции и пополняемого абстракциями. Например, для рассматриваемых фрагментов можно создать следующую абстракцию:

```
f := is_in(a, e)
if f then z := index_of(a, e) end
```

```
f := false;
i := lowbound(a);
while i < highbound(a) + 1
do
 if a[i] = e then z := i
  f:= true
 end
 i := i + 1
end
```

```
f := false;
i := highbound(a);
while i > lowbound(a) - 1 do
 if a[i] = e then z := i
  f:= true
 end
 i := i - 1
end
```

Рис. 1. Уровень абстракции, определяемый применяемым языком

Этот подход таит опасность накопления большого количества абстракций, с которыми работать стало бы невозможно. Альтернативой может быть создание языковых механизмов, позволяющих программисту создавать свои собственные абстракции по мере надобности. Наиболее распространенным механизмом такого рода является использование процедур. Разделяя в программе тело процедуры и обращения к ней, язык высокого уровня реализует два важных метода абстракции: абстракцию через параметризацию и абстракцию через спецификацию.

Второй подход не отрицает первый, который можно сравнить с развитием естественного языка, содержащего значительное количество слов, но каждый пользователь языка применяет только некоторое подмножество всех слов.

Рассмотрение композиции и абстракции преследует цель разработки механизмов машинного понимания смысла задачи, когда данные задачи помещены в виде некоторой формальной структуры в памяти ЭВМ. Это не означает, что только использование композиции и абстракции приведет к созданию таких механизмов, но полное отрицание их применения может привести к тупиковому пути.

Абстракция через параметризацию позволяет представить фактически неограниченный набор различных вычислений одной программой. Интересным на этом пути подходом

является построение многоуровневого механизма абстракции в виде набора некоторых деревьев, когда абстракции более высокого уровня может создавать не только пользователь, но и программы механизма абстракции.

Абстракция через спецификацию позволяет абстрагироваться от процесса вычислений до уровня знания того, что данная процедура должна реализовать. Это достигается путем задания для каждой процедуры спецификации, описывающей эффект ее работы, после чего смысл обращения к данной процедуре становится ясным через анализ спецификации, а не тела процедуры. Абстракция через спецификацию используется каждый раз, когда с процедурой связывается некоторый комментарий, достаточно информативный для того, чтобы иметь возможность работы с ней без анализа тела процедуры. Одним из возможных способов является указание для каждой процедуры пары утверждений, задающих истинность начального условия (requires) и истинность конечного условия (effects). Утверждение requires задает на входе процедуры истинность или ложность некоторого условия. Утверждение effects задает некоторое условие, которое предполагается истинным по завершении данной процедуры в предположении, что для этой процедуры было удовлетворено начальное условие. При анализе спецификации для уяснения смысла обращения к процедуре необходимо придерживаться двух правил: после выполнения процедуры можно считать, что конечное условие выполнено; можно ограничиться только теми свойствами, которые подразумевает конечное условие.

Метод абстракции через параметризацию и метод абстракции через спецификацию позволяет определить три различных вида абстракций: процедурную абстракцию; абстракцию данных, абстракцию через итерацию.

В общем случае каждый вид абстракции использует оба метода абстракции. Процедурная абстракция позволяет расширить заданную некоторым способом программирования

виртуальную машину новой операцией. Абстракция данных состоит из набора объектов и набора операций, характеризующих поведение этих объектов. Абстракция через итерацию дает возможность не рассматривать информацию, не имеющую прямого отношения к управляющему потоку или циклу.

С каждой задачей соотносятся определенного вида знания, которые или явно хранятся в ЭВМ, или их необходимо приобретать. Но даже наличие достаточного количества знаний не позволяет решить задачу, если не активизировать их с помощью специальных процедур, которые явно или неявно заложены в семантических сетях.

Можно отметить три особенности знаний [9, p. 2], которыми они отличаются от данных: интерпретируемость; наличие классифицирующих связей; наличие ситуативных отношений. Когда данные помещены в машинную память, то их содержательная интерпретация становится невозможной. В этом случае интерпретация реализуется через работу программы с этими данными. Выбирая в нужный момент те или иные данные из памяти и совершая над ними те или иные преобразования, программа как бы учитывает содержательный аспект данных. Содержательный аспект данных вносится извне программистом при создании программы. В самой ЭВМ, в хранящихся в ее памяти данных и программе нет информации о содержательной интерпретации. Для возможности интерпретации необходимо кроме данных хранить описательную информацию об этих данных.

При работе с данными различного типа нет возможности установить между ними связь, отражающую зависимость между этими данными. Напротив, при работе со знаниями становятся возможными связи между отдельными единицами знаний, которые отражают характер их взаимоотношений. Можно выделить, например, такие классифицирующие связи: род — вид; элемент — класс; класс — подкласс; тип — подтип; ситуация — подситуация и т. д. Ситуативные связи определя-

ют ситуативную совместимость тех или иных знаний, хранимых в памяти. Эти связи могут определять самые разнообразные отношения: осуществляться одновременно, быть в одной области пространства и т. д.

При работе со знаниями необходимо иметь процедуры пополнения знаний, процедуры обобщения, позволяющие вводить классифицирующие знания и ряд других процедур.

Машинное понимание смысла задачи опирается на представление знаний в базе знаний, на встроенные процедуры в базу знаний и на способ общения пользователя с базой знаний, другими словами, можно говорить о моделях представления знаний, моделях преобразования знаний, моделях общения с базами знаний. Если в этих моделях выделяются языковые аспекты, то говорят о языках представления знаний, языках манипулирования знаниями и языках общения с базами знаний.

Различают два типа знаний: декларативные и процедурные. Декларативные знания соответствуют различным видам типизации, процедурные знания реализуются в наборе операций. В семантических функциональных сетях могут храниться процедурные знания в явном виде. В семантической сети определений могут содержаться описания, которые надо превратить в процедурные знания или ссылки на процедурные знания.

Известно, что структуры знаний человека не подобны множеству предложенных структур знаний в искусственном интеллекте. Знания человека ситуативны, и он хранит информацию, взаимно исключающую друг друга. Значит, модель представления знаний должна отражать эту ситуативность при необходимости. Знания человека принципиально неполны, а существующие у него механизмы позволяют использовать эти неполные знания для принятия решений и организации процедур. Знания у человека играют активную роль.

Исходя из вышеизложенного можно сделать следующие выводы:

— для машинного понимания смысла задачи необходимо представлять ее в виде некоторой модели;

— понимание задачи, представленной в виде некоторой модели, должно опираться на представление знаний в виде некоторых моделей, необходимых для возможности нахождения подходов решения этой задачи;

— для использования хранящихся знаний и возможности приобретения знаний необходимы модели представления механизмов решения задач.

Неформально под семантической сетью понимается сеть с помеченными вершинами и дугами, в основе которой лежит математическая модель алгебраической реляционной системы [10, 11]. Рассмотрим формальное определение семантической сети.

Пусть A = {A_1, A_2,…, Ar} — это множество символов, называемое атрибутами. Схемой или интенсионалом некоторого отношения Rn_i в атрибутивном формате называется набор пар

$INT(Rn_i) = \{(Aj, DOM(Aj)) : Aj \in A, j = 1, 2,\ldots, n_i\}$,

где Rn_i — имя отношения; n_i — местность отношения; Aj — атрибут отношения Rni; DOM(Aj) — множество значений атрибута Aj.

Объединение всех доменов W — базовое множество модели, набор объектов, на которых задаются отношения Rn_i i = 1, 2,…, m — число различных отношений.

Экстенсионалом отношения Rn_i называется множество

$EXT(Rn_i) = \{Fk : k = 1, 2,\ldots, p_i\}$,

где p_i — кардинальность множества $EXT(Rn_i)$, Fk — факты отношения Rn_i, записываемые в виде $Fk = \{\nu_{ijk} : \nu_{ijk}$ есть значение j-го атрибута k-го факта экстенсионала отношения $Rn_i\}$. Последовательность из двух элементов вида «атрибут — значение» называется атрибутивной парой.

Семантическая сеть — это совокупность $СемС = \{(INT(Rn_i), EXT(Rn_i)) : i = 1,2,\ldots, m\}$, записываемая в виде ассоциативной

структуры данных. В семантической сети используются самые разнообразные типы структур, но требование ассоциативности, то есть группирования информации вокруг фактов, атрибутов или объектов, является обычно характерным.

Понятие семантической сети естественным образом распадается на понятие экстенсиональной семантической сети, или базы данных ЭСемС = {EXT(Rn_i)}, и интенсиональной семантической сети ИСемС ={INT(Rn_i)}, или базы знаний. Их объединение также назовем базой знаний.

Помимо введенных понятий объектов W и отношений R включают в рассмотрение категорию свойства объекта. При этом обычно имеется в виду одна из двух интерпретаций этого понятия. В соответствии с первой интерпретацией свойство эквивалентно одноместному предикату. Вторая интерпретация этого понятия опирается на понятие «ситуация». Ситуации могут задаваться на уровне своих интенсионалов («классы ситуаций») или экстенсионалов. Экстенсиональной ситуацией или просто ситуацией называется выделение множества фактов, принадлежащих экстенсионалам, представленных в модели отношений. В частности, ситуация может включать один факт и всю экстенсиональную семантическую сеть, что соответствует глобальной ситуации. В этом случае ситуация является обобщением понятия факта. Любое собственное подмножество глобальной ситуации называется локальной ситуацией. Базовым множеством ситуации Sj называется множество объектов из W, фигурирующих в Sj. Если базовое множество ситуации Sj обозначить через B(Sj), то можно записать:

$B(Sj) = \{w : w \in W, w \leftrightarrow Sj\}$.

Понятие глобального свойства объекта опирается на ситуацию, заданную компонентом связности графа экстенсиональной семантической сети, включающей этот объект. Локальные свойства задаются подситуациями ситуации, соответствующей компоненте связности. Глобальное свойство объекта w обозначается как $PR_G(w)$, а локальное — $PR_L(w)$.

Базу знаний, построенную на основе семантической сети, можно представить состоящей из трех основных частей (рис. 2).

Экстенсиональная семантическая сеть фиксирует состояние конкретных объектов модели: это «фотографии» действительности или серия «снимков» для разных моментов времени или временных интервалов.

При разработке баз данных решаются следующие вопросы: выбор конкретного способа представления фактической информации; представление времени в системе; принципы фрагментации базы данных и выбор изобразительных средств организации различных видов логических структур базы данных; формирование набора и семантики системных объектов, отношений и атрибутов, которые имеют специальный смысл и обрабатываются особым образом в отличие от проблемных объектов, отражающих семантику предметной области; установление правил работы с десигнатами объектов; фиксация способов задания нестрогой, недоопределенной или неопределенной информации.

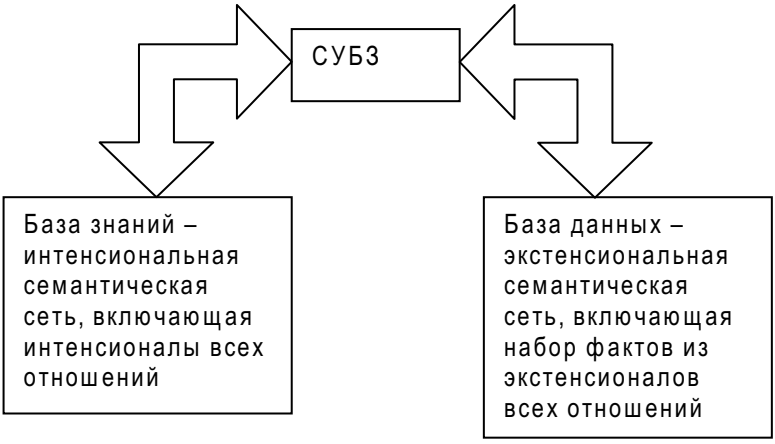

Рис. 2. Основные части базы знаний

В базах знаний концентрируется, в первую очередь, информация, описывающая общую структуру соответствующей предметной области. Ее основу составляют интенсиональные семантические сети, задающие схему отношений реализуемой модели. Для представления знаний в базах знаний нужно: выбрать способы представления обобщенной информации; решить вопросы об ориентации на абстрактных объектах или отношениях; уточнить перечень базовых процессов, выполнение которых может поддерживать СУБЗ; разработать структуры представления соответствующих знаний; контроль ввода, исключения и модификации данных; реализовать причинно-следственные трансформации модели; разработать логический вывод; реализовать процедуры планирования целенаправленной активности и выполнения соответствующих планов; создать поддержку режима имитационного моделирования, выполняемого над базой данных; разработать поддержку процессов интерпретации и синтеза текстов языка общения с внешней средой; обеспечить выполнение процедур обучения, дообучения и обобщения путем синтеза и корректировки соответствующей структуры базы знаний и базы данных.

В базах знаний принято хранить информацию об общих структурах, зависимостях, стратегиях, процедурах, описывающих конкретную предметную область. При рассмотрении экстенсиональных семантических сетей используют абстрактные понятия «объект» и «отношение». Таксономическая структура в семантических сетях строится на соотнесении понятий по категории «род — вид». Таксономия на отношениях позволяет сократить количество информации, фиксируемой в базе данных, и облегчает логический вывод, связанный с переходом от более общих отношений к частным и наоборот.

В основе функционирования баз знаний лежит информационно-поисковый режим. Поиск по базе знаний выполняется в целях поиска релевантных правил, коррекции базы знаний или для оценок характеристик определенных правил. При по-

иске ответа требуется решить задачу изоморфного вложения графа запроса в граф базы знаний или базы данных. При этом попутно происходит конкретизация переменных в графе запроса. Если запрос вложим в базовую структуру, переменные получают в качестве значений соответствующие термы из базы.

Базовым отношением называется такое, экстенсионал которого полностью хранится в базе данных. Виртуальное отношение не имеет эксплицитно хранимого экстенсионала, но в его декларации содержатся правила, структуры или программы, позволяющие конструировать необходимые факты по мере надобности. Реализация виртуальных отношений обеспечивает информационно-логический режим функционирования системы поддержки данных и знаний.

Существуют три основных способа реализации виртуальных отношений: использование вычислимых отношений; использование свойств симметричности, рефлексивности, транзитивности; использование Хорновских моделей. Вычислимые отношения реализуются программным путем, вычисления носят функциональный или предикатный характер: по n заданным значениям атрибутов вычисляется значение недостающего атрибута; если задаются все значения атрибутов и если в результате вычислений выяснится, что этот набор принадлежит экстенсионалу данного отношения, то выдается значение «истина», в противном случае — «ложь». Задание отношения на Хорновских моделях имеет характер импликации:

$R_2(a_1, a_2,.., a_n)\&R_3(a_1, a_2,\ldots, a_m)\& \ldots \&R_k(a_1, a_2,\ldots., a_t) \rightarrow R_1(a_1, a_2,\ldots, a_s)$.

Записанное правило означает:

Если удается установить истинность отношений $R_2, R_3,\ldots, R_k$, то можно утверждать, что истинно и отношение R_1.

Под механизмом решения и составления планов решения задач понимается совокупность программных средств и управляющей и уточняющей информации во вспомогательных семантических сетях для работы с основными семантическими сетями.

Список литературы

1. *Пойа Д.* Математическое открытие. М.: Наука, 1976.

2. *Микиша А.М., Орлов В.Б.* Толковый математический словарь. М.: Русский язык, 1988.

3. *Ожегов С.И.* Словарь русского языка. М.: Советская энциклопедия, 1972.

4. Философский словарь / Под ред. Фролова И.Т. Изд. Четвертое. М.: ИПЛ, 1980.

5. Представление и использование знаний: Пер с япон./ Под ред. Уэно Х., Исидзука М. М.: Мир, 1989.

6. *Петров Ю.А.* Методологические вопросы анализа научного знания. М.: ВШ, 1977.

7. *Орлов С.А.* Технология разработки программного обеспечения: разработка сложных программных систем. М.: ПИТЕР, 2002.

8. *Лисков Б., Гатэг Дж.* Использование абстракций и спецификаций при разработке программ. М.: Мир, 1989.

9. *Moore J., Bailin S.* 1988. Position Paper on Domain Analysis. Laurel; MD:CTA.

10. *Мальцев А.И.* Алгебраические системы. М.: Наука, 1970.

11. *Дейт К.* Введение в системы базы данных. М.: Наука, 1980.