ЛАБОРАТОРНАЯ РАБОТА №5

Построение продукционной базы знаний экспертной системы

Цель работы. Изучение возможностей представления знаний в виде продукционных правил. Реализация продукционной базы знаний средствами языка Пролог.

Структура экспертной системы

Экспертная система (ЭС) - это компьютерная система, содержащая знания специалистов-экспертов в некоторой предметной области и способная принимать решения, близкие по качеству к решениям эксперта.
В структуре экспертной системы обычно выделяют следующие компоненты:

- база знаний и данных;
- механизм вывода;
- интерфейс пользователя;
- подсистема объяснения;
- подсистема извлечения знаний.

База знаний (БЗ) содержит знания, относящиеся к конкретной предметной области, в том числе факты, правила (отношения между фактами), оценки достоверности фактов и правил (например, коэффициенты уверенности). Знания экспертов в БЗ представляются в некоторой стандартной форме, например, в виде продукционных правил.

Механизм вывода - это программные средства, обеспечивающие поиск решений на основе базы знаний. Действия механизма вывода аналогичны рассуждениям человека-эксперта. Механизм вывода выполняет поиск решений в базе знаний, а также оценивает достоверность предлагаемых решений.

Интерфейс пользователя обеспечивает обмен информацией между пользователем и ЭС. Обычно интерфейс пользователя строится на основе системы меню. Кроме того, существуют интерфейсы на основе командного языка ЭС и элементов естественного языка.

Подсистема объяснения - это программные средства, объясняющие пользователю процесс вывода решения. Эта подсистема должна иметь возможность объяснять следующее: как ЭС пришла к какому-либо выводу (ответ на вопрос "как"); для чего у пользователя запрашивается та или иная информация (ответ на вопрос "почему").

Подсистема извлечения знаний предназначена для пополнения и корректировки базы знаний. В этой подсистеме могут быть реализованы методы приобретения знаний у экспертов. Такие методы могут быть прямыми (у эксперта непосредственно запрашиваются его знания) или косвенными (например, обучение ЭС по набору примеров рассуждений, приведенных экспертом). Могут применяться также методы обучения ЭС в процессе работы (например, методы обучения на ошибках).

Представление знаний в продукционных ЭС

Продукционная ЭС - это экспертная система, в которой знания экспертов представлены в виде правил "если - то", т.е. продукционных правил. Каждое правило представляет собой некоторое условное утверждение (например, ЕСЛИ условие, ТО заключение; ЕСЛИ ситуация, ТО действие). Представление знаний в виде набора правил и имеет следующие основные достоинства:

• естественность: человек-эксперт во многих случаях выражает свои знания именно в форме правил;
• модульность: каждое правило представляет собой один относительно независимый фрагмент знаний;
• удобство модификации базы знаний (как следствие модульности); можно добавлять новые и изменять существующие правила, не изменяя при этом других правил;
• прозрачность: удобство объяснения процесса вывода решения.

Основными элементами правила являются посылка (условие применимости правила) и заключение (действие, выполняемое в случае истинности посылки). Кроме того, в структуру правила могут входить также метка (номер правила или некоторое поясняющее обозначение), элементы для объяснения (комментарии), оценка достоверности правила и некоторые другие элементы.

Представление базы знаний средствами языка Пролог

Рассматриваемый в данной работе вариант представления базы знаний близок к форме представления знаний, используемой в оболочке для построения продукционных экспертных систем GURU.
Основными элементами продукционного правила являются посылка (условная часть) и заключение (действие). Условная часть правила - это набор условий ("больше", "меньше", "равно" и т.д.). Будем считать, что все условия, составляющие условную часть правила, объединены логической операцией “и”. Если требуется задать условие с использованием операции “или”, то следует ввести в базу знаний несколько правил (по одному для каждого из условий, связанных операцией “или”). Заключение правила - это действие, выполняемое при истинности его условной части. Для простоты будем считать, что в правилах возможно только одно действие: присваивание значения переменной. Будем также считать, что в каждом правиле может быть только одно действие.

При реализации базы знаний для продукционной ЭС средствами языка Пролог каждое правило будем представлять в виде предиката базы данных. Этот предикат будет содержать основные элементы правила (посылку и заключение), а также его номер. Для предикатов, описывающих правила, будем использовать имя rule (правило). Эти предикаты можно объявить следующим
образом:

```
global facts
rule (integer, usl, zakl)
```

В предикате rule первый аргумент (с типом integer) - номер правила; usl - условная часть правила (посылка); zakl - заключение (действие). Очевидно, что типы usl и zakl - нестандартные (в Прологе таких типов нет). Поэтому их
необходимо объявить в разделе global domains.
Условную часть правила объявим в разделе global domains следующим образом:

uslovie = eq (string, string);
ne (string, string);
gt (string, string);
lt (string, string);
le (string, string);
ge (string, string)
usl=uslovie*

Здесь eq, ne, gt, lt, le, ge - функторы (см. подраздел 3.1), с помощью которых будут представляться отдельные условия, составляющие условную часть. Функтором eq будем обозначать условие "равно", ne - "не равно", gt- "больше", lt - "меньше", le - "меньше или равно", ge - "больше или равно". Здесь использованы обозначения операций сравнения, образованные от соответствующих английских слов. Они не являются какими-либо зарезервированными словами, поэтому можно использовать любые другие обозначения (например, вместо eq – rav, вместо ne – nerav, и т.д.). Будем считать, что во всех этих функторах первым аргументом является имя переменной, а вторым - некоторая константа, с которой сравнивается значение переменной. Для простоты будем считать, что все величины, обрабатываемые в ЭС - строковые.
Тип uslovie - альтернативный домен. Объект этого типа может представлять собой *любой из функторов* eq, ne, gt, lt, le, ge (но только *один*).
Тип usl - *список* объектов типа uslovie. Таким образом, объект с типом uslovie может представлять собой *список из любого количества функторов*. Такое объявление позволяет описывать условные части правил, состоящие из произвольного количества различных условий.
Заключение правила объявим в разделе global domains следующим образом:

zakl = assign (string, string)

Здесь assign - функтор, которым будет обозначаться присваивание переменной некоторого значения (вместо assign можно использовать любое другое имя). Имя переменной будем указывать первым аргументом, а присваиваемое значение - вторым. Кроме правил, в базе знаний будем хранить описания переменных, используемых в ЭС. Описание переменной будет состоять из двух частей: имя переменной, а также запрос, выводимый на экран при вводе значения данной переменной. Важно отметить, что запрос требуется только для переменных,
значения которых не определяются самой ЭС в ходе обработки правил, а запрашиваются у пользователя. Описание каждой переменной будем представлять в виде предиката базы данных с именем var (можно использовать
любое другое имя). Этот предикат необходимо объявить в разделе global facts следующим образом:

var (string, string)

Первый аргумент этого предиката будет представлять собой имя переменной, второй - текст запроса (если он требуется).
Необходимо также указать целевую переменную, т.е. переменную, значение которой необходимо определить в результате работы ЭС. Для ее указания объявим в разделе global facts следующий предикат базы данных: goal_var (string)
Аргументом этого предиката базы данных будет имя целевой переменной.

Примечание. Следует еще раз обратить внимание, что все обозначения, использованные при описании базы знаний (rule, eq, lt, goal_var и т.д.) не являются какими-либо

стандартными обозначениями языка Пролог. Вместо них можно использовать любые другие обозначения (например, вместо rule - pravilo, вместо var - perem, вместо assign - prisv, и т.д.).

Примечание. Операции, обозначенные с помощью функторов (сравнение, присваивание и т.д.), будут реализованы при построении механизма вывода.
Примечание. Не следует путать переменные, используемые в правилах (т.е. переменные ЭС), с переменными программы на языке Пролог. Как будет показано ниже, переменные ЭС для программы на языке Пролог - это просто данные, обрабатываемые программой в ходе ее выполнения.

Пример базы знаний

Пусть в ЭС, используемой для диагностики неисправностей некоторого устройства, требуется указать следующие правила:

• если температура выше 200ºС, то имеется неисправность типа 1;
• если температура - от 100 до 200ºС, и при этом есть вибрация, то имеется неисправность типа 2, а при той же температуре и отсутствии вибрации - неисправность типа 3;
• при неисправности типа 1 или 2 необходимо прекратить работу устройства, при неисправности типа 3 - сменить режим работы.

База знаний, содержащая эти правила, может выглядеть так:

```
goal_var("D")
var("T", "Введите температуру: ")
var("Vibr", "Есть ли вибрация (да/нет): ")
var("Neispr","")
var("D","")
rule(1,[gt("T","200")],assign("Neispr","тип 1"))
rule(2,[gt("T","100"),le("T","200"),eq("Vibr","да")],assign("Neispr","тип 2"))
rule(3,[gt("T","100"),le("T","200"),eq("Vibr","нет")],assign("Neispr","тип 3"))
rule(4,[eq("Neispr", "тип 1")],assign("D", "прекратить работу"))
rule(5,[eq("Neispr", "тип 2")],assign("D", "прекратить работу"))
rule(6,[eq("Neispr", "тип 3")],assign("D", "сменить режим работы"))
```

Примечание. Каждое правило (как и любой факт, хранящийся в базе данных на Прологе) должно быть набрано в одну строку.

Здесь переменная D объявлена как целевая.
Для переменных Neispr и D не указан текст запроса, так как эти переменные не запрашиваются у пользователя, а определяются самой ЭС на основе правил и данных, предоставленных пользователем (температуры и вибрации). Однако указание пустой строки (“”) вместо текста запроса для этих переменных *обязательно*, так как в объявлении предиката var указано, что он
имеет *два* аргумента.

Построение ЭС на языке Пролог

Приведем пример программы на языке Пролог, представляющей собой демонстрационный вариант продукционной ЭС. Программа реализует следующие действия: загрузку базы знаний ЭС, ее просмотр, сеанс работы с ЭС (консультацию). Таким образом, из основных элементов ЭС.
в этой программе предусмотрена реализация базы знаний и механизма вывода.
Структура и принцип работы предлагаемой программы аналогичны программе для работы с базой данных, рассмотренной в лабораторной работе 5.

```
goal
start
clauses
start:-clearwindow,
makewindow (1,7,7,"Экспертная система",0,0,8,60),
repeat, clearwindow,
write ("1 - загрузка"), nl,
write ("2 - просмотр базы знаний"), nl,
write ("3 - консультация "), nl,
write ("99 - выход"), nl,
write ("Введите номер: "), readint (N),
obrab (N), removewindow.
obrab(1):- makewindow(2,7,7,"",8,0,3,60), zagruzka,
removewindow, shiftwindow(1), !,fail.
obrab(2):- makewindow(3,7,7,"База знаний",12,0,10,80), prosmotr_bz,
removewindow, shiftwindow(1), !,fail.
obrab(3):- makewindow(4,7,7,"Консультация",12,0,10,80), vyvod,
removewindow, shiftwindow(1), !,fail.
obrab(99).
```

Предикат zagruzka реализует загрузку базы знаний, предикат prosmotr_bz - просмотр базы знаний на экране, предикат vyvod – работу
механизма вывода. Реализация предиката vyvod будет рассмотрена в следующей лабораторной работе. Предикаты zagruzka и prosmotr_bz предлагается реализовать самостоятельно. Они реализуются аналогично предикатам загрузки и просмотра базы данных, рассмотренным в лабораторной работе 5.
В предлагаемой программе необходимо указать также объявление предикатов базы данных и нестандартных типов данных в разделах global facts и global domains (см. подраздел 5.3), а также объявление всех используемых
предикатов в разделе predicates.

Порядок выполнения работы

По заданию, выданному преподавателем, представить набор заданных утверждений в виде продукционных правил. Реализовать полученную продукционную базу знаний в виде базы данных на Прологе. Разработать и отладить программу на языке Пролог для работы с продукционной базой знаний. Реализовать операции загрузки, поиска и просмотра базы знаний.

Контрольные вопросы

1. Структура экспертной системы.
2. Понятие продукционной экспертной системы.
3. Структура продукционного правила.
4. Представление продукционных правил средствами языка Пролог.
5. Как передать условия в правилах.