Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет информационных технологий и управления

Кафедра информационных технологий автоматизированных систем

Отчёт
по практической работе №5
«Построение продукционной базы знаний экспертной системы»
по дисциплине «Экспертные Системы»

Выполнил:
студент гр. 820601
Шведов А. Р.

Проверила:
Герман Ю. О.

Минск 2022

ЦЕЛЬ РАБОТЫ

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

1 ХОД РАБОТЫ

2.1 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).

2.2 Понятие продукционной экспертной системы.

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 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.

2.3 Структура продукционного правила.

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”.

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=gt(string,string);

lt(string,string);

eq(string,string)

usl=uslovie*

Здесь gt, lt, eq – функторы, с помощью которых будут представляться отдельные условия, составляющие условную часть. Функтором eq будем обозначать условие "равно", gt– "больше", lt – "меньше". 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* gt, lt, eq (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.

2.4 Представление продукционных правил средствами языка Пролог.

Индивидуальное задание:

Составляется набор правил для прогнозирования погоды на основе анализа 4 признаков: скорости ветра, давления, температуры, влажности воздуха. Экспертом приведены следующие примеры:

1) если давление высокое, температура выше 30 С, влажность более 60%, то прогноз – П1;

2) если давление умеренное, температура выше 30 С, влажность более 90%, то прогноз – П2;

3) если давление умеренное, температура выше 30 С, влажность – от 60 до 90%, то прогноз – П1;

4) если скорость ветра высокая, давление умеренное, температура выше 30 С, влажность – менее 60%, то прогноз – П1;

5) если скорость ветра низкая, температура выше 30 С, влажность – менее 60%, то прогноз – П2;

6) если скорость ветра высокая, давление низкое, температура выше 30 С, то прогноз – П1;

7) если скорость ветра низкая, давление низкое, температура выше 30 С, то прогноз – П2;

8) если давление высокое, температура от 10 до 30 С, влажность – от 60 до 90%, то прогноз – П3;

9) если давление низкое, температура от 10 до 30 С, влажность – от 60 до 90%, то прогноз – П4;

10) если скорость ветра низкая, давление умеренное, температура от 10 до 30 С, то прогноз – П3;

11) если скорость ветра высокая, температура от 10 до 30 С, влажность более 90%, то прогноз – П2;

12) если температура от 10 до 30 С, влажность менее 60%, то прогноз – П3;

13) если температура ниже 10 С, то прогноз – П3.

Построить набор правил.

База знаний, содержащая правила:

goal_var("D")

var("P", "Enter pressure (low/high/mid): ")

var("T", "Enter temperature: ")

var("WS","Enter wind speed (low/high): ")

var("H","Enter humidity: ")

var("Forecast","")

rule(1,[eq("P","high"), gt("T", "30"), gt("H", "60")],assign("Forecast","type 1"))

rule(2,[eq("P","mid"), gt("T", "30"), gt("H", "60"), lt("H", "90")],assign("Forecast","type 1"))

rule(3,[eq("P","mid"), gt("T", "30"), gt("H", "90")],assign("Forecast","type

2"))

rule(4,[eq("WS", "high"), eq("P","mid"), gt("T", "30"), lt("H", "60")],assign("Forecast","type 1"))

rule(5,[eq("WS", "low"), gt("T", "30"), lt("H", "60")],assign("Forecast","type 2"))

rule(6,[eq("WS", "high"), eq("P","low"), gt("T", "30")],assign("Forecast","type 1"))

rule(7,[eq("WS", "low"), eq("P","low"), gt("T", "30")],assign("Forecast","type 2"))

rule(8,[eq("P","high"), lt("T", "30"), gt("T", "10"), gt("H", "60"), lt("H", "90")],assign("Forecast","type 3"))

rule(9,[eq("P","low"), gt("T", "10"), lt("H", "30"), gt("H", "60"), lt("H", "90")],assign("Forecast","type 4"))

rule(10,[eq("WS", "low"), eq("P","mid"), gt("T", "10"), lt("T", "30")],assign("Forecast","type 3"))

rule(11,[eq("WS", "high"), gt("H","90"), gt("T", "10"), lt("T", "30")],assign("Forecast","type 2"))

rule(12,[lt("h","60"), gt("T", "10"), lt("T", "30")],assign("Forecast","type 3"))

rule(13,[lt("T", "10")],assign("Forecast","type 3"))

Решение:

include "lab5.inc"

predicates
nondeterm zagruzka
nondeterm prosmotr
nondeterm vyvod
nondeterm start
nondeterm repeat
nondeterm obrab(integer)
goal
start.
clauses
start:–
repeat,
write("1 – loading"), nl,
write("2 – view knowledge base"), nl,

```
write("3 – consultation "), nl,
write("4 – exit"), nl,
write("Enter number: "), readint (N),
obrab(N).
obrab(1):– zagruzka, !,fail.
obrab(2):–prosmotr, !,fail.
obrab(3):– vyvod, !,fail.
obrab(4).

zagruzka:–retractall(_),consult(“/tmp/lab5.txt"),write("Base successfully loaded"),readchar(_),!.
zagruzka.

prosmotr:–var(X,_),
write("Variable: ",X),nl,
readchar(_),fail.
prosmotr:–rule(N,Rule,Zakl),write("Number: ",N," if: ",Rule," then: ",Zakl),nl,readchar(_),fail.
prosmotr.

vyvod.

repeat.
repeat:–repeat.
```