Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

Лабораторная работа №4. Язык искусственного интеллекта PROLOG

Дисциплина: Интеллектуальные системы

Выполнил студент гр. 13541/1 _____А.А. Дроздовский
(подпись)

Руководитель _____А.М. Сазанов
(подпись)
“___” _____2017 г.

Санкт – Петербург
2017

1. Задание

1.1. Получите начальное представление о синтаксисе и семантике базовых конструкций языка PROLOG, ознакомившись с разделами 1-5 методического пособия

- Бураков С. В. «Язык логического программирования PROLOG», СПбГУАП, 2003.
- Середа С.Н. «Методичка по языку Prolog», Муромский университет. 2003г.

1.2. Создайте проект в оболочке Visual Prolog 7.3., как это показано в примере http://wikiru.visual-prolog.com/index.php?title=%D0%9E%D1%81%D0%BD%D0%BE%D0%B2%D1%8B_%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B_Visual_Prolog

1.3. Удалите проект, созданный в п. 2 и запустить демонстрационный проект family1 в оболочке Visual Prolog 7.3.

1.4. Постройте генеалогическое дерево для данного примера на основе результатов выполнения программы и исходного кода программы.

1.5. Построить описание онтологии из данного примера на естественном языке.

1.6. Построить концептуальную карту (семантическую сеть), описывающую данный пример (подсказка: используйте понятие фрейма).

1.7. Создать проекты 1-21 для каждого из примеров в пособии из п.1 и привести листинги результатов работы каждой из программ в ответ на запросы пользователя. При создании проектов указывать тип «консольный». Чтобы протестировать консольную программу, используйте команду *Run in Window*, не *Execute*. Синтаксис языка Пролг, использующийся в версии 7.3 уточнить по ссылке http://wikiru.visual-prolog.com/index.php?title=%D0%9A%D0%B0%D1%82%D0%B5%D0%B3%D0%BE%D1%80%D0%B8%D1%8F:VipLanguage

1.7. Выполнить одно из индивидуальных заданий (см. задание 9-15 на стр. 32-34 из пособия Бураков С. В. «Язык логического программирования PROLOG», СПбГУАП, 2003).

1.8. Изучить 1-2 лабы по методичке Седана С.Н. (доп литература)

Согласно своему варианту решить задачу с помощью PROLOG. Продемонстрировать скриншотами и, при желании, нарисовать дерево решения.

1.9. Написать выводы. В выводах отразить, помимо своих мыслей, возникших в ходе работы, ответы на приведенные ниже вопросы:

1. В чем Плюсы и минусы языка Prolog
2. Какие еще языки используются для разработки ИИ, приведите примеры (НЕ МЕНЕЕ 2-х) проектов, языков и краткое описание проектов. (Альтернативы PROLOG)
3. Решаема ли проблема комбинаторного взрыва, пути решения.
4. Корректно ли по-вашему в принципе разработка языка ИИ? Что он должен из себя представлять?
5. Можно ли разработать ИИ не понимая, как он работает, должны ли мы понимать, как он работает, думает, рассуждает?

2. Ход работы

2.1. Построить генеалогическое дерево для примера на основе результатов выполнения программы и исходного кода программы:

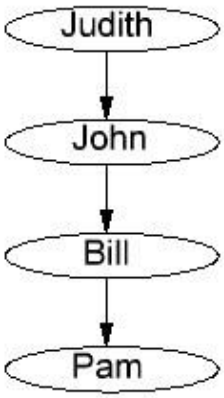

Рис. 2.1. Генеалогическое дерево, соответствующее данным проекта family1.

Согласно данному дереву, Джон является дедушкой Пэм и сыном Джудит, Билл является отцом Пэм. Все они являются родственниками Пэм. Данное описание соответствует решению, предоставленному программой.

2.2. Построить генеалогическое дерево для примера на основе результатов выполнения программы и исходного кода программы.

Онтология данного примера состоит из следующих знаний:

1. Существует мужской и женский пол;
2. Человек имеет имя и пол;
3. Человек может быть родителем другого человека;
4. Если человек – родитель мужского пола, то он отец;
5. Если человек – отец родителя другого человека, то он является дедушкой этого человека;
6. Родитель человека – это его предок;
7. Предок родителя человека – предок этого человека.

2.3. Построить концептуальную карту (семантическую сеть), описывающую данный пример.

Данная семантическая сеть показывает людей и родственные отношения между ними:

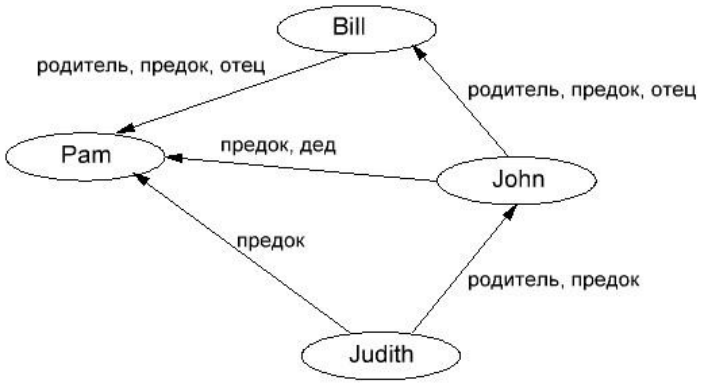

Рис. 2.2. Семантическая сеть примера family1.

2.4. Создать проекты 1-21 для каждого из примеров в пособии [1].

Далее приведены листинги вывода программ, составленных на основе примеров из пособия. Соответствующие проекты содержатся в прикрепленном архиве.

- Пример 1:

```
Berlin in Europe

SPb in Europe

Warszawa in Europe
```

- Пример 2:

```
Петров is owner of this book
```

- Пример 3:

```
Sidor is brother of Igor
```

- Пример 4:

```
Sum = 7

FSum = 6.5

4 is greater of 3 and 4

End
```

- Пример 5:

```
China of territory 9597000

Peru of territory 1285000

End
```

- Пример 6:

```
Hello
```

- Пример 7:

```
Moscow

Minsk

Kiev

Omsk
```

```
end
```

- Пример 8:

```
Петя

Вася

Олег
```

- Пример 9:

```
end
```

- Пример 10:

```
number is 1

number is 2

number is 3

number is 4

number is 5

number is 6

number is 7

end
```

- Пример 11:

```
Enter a number: 777

Summ = 21

end
```

- Пример 12:

```
Enter a number of disks: 3

Disk from left() to right()

Disk from left() to middle()

Disk from right() to middle()

Disk from left() to right()

Disk from middle() to left()

Disk from middle() to right()
```

```
Disk from left() to right()
```

- Пример 13:

```
second dog is: борзая
```

- Пример 14:

```
лайка есть в списке
```

- Пример 15:

```
summ of list 17
```

- Пример 16:

```
A solution is:

 Мужик везет козу с west() берега на east()

 Мужик едет с east() на west()

 Мужик везет капусту с west() берега на east()

 Мужик везет козу с east() берега на west()

 Мужик везет волка с west() берега на east()

 Мужик едет с east() на west()

 Мужик везет козу с west() берега на east()
```

- Пример 17:

```
Alex at Three, Petr at One, Nike at Two
```

- Пример 18:

```
Sergey is from Moskow
```

- Пример 19:

```
Students:

Andrey,Dmitry,no,no,no

Students:

Andrey,no,Boris,no,Grigory

Students:

Andrey,no,Boris,no,no
```

```
Students:
Andrey,Dmitry,no,no,no
Students:
no,no,no,Viktor,Grigory
Students:
Andrey,Dmitry,Boris,no,no
Students:
Andrey,no,no,no,Grigory
Students:
Andrey,no,no,no,no
Students:
no,Dmitry,Boris,Viktor,no
Students:
no,Dmitry,no,Viktor,no
Students:
no,no,Boris,Viktor,Grigory
Students:
no,no,Boris,Viktor,no
Students:
no,no,no,Viktor,no
end
```

- Пример 20:

```
player in DB: Михайлов 180 87
player in DB: Петров 187 93
player in DB: Харламов 177 80
end
```

- Пример 21:

```
вопрос – у рыбы вес >40 кг? (да/нет)
```

```
нет
вопрос – у рыбы вес <40 кг? (да/нет)
да
вопрос – у рыбы есть усы? (да/нет)
нет
вопрос – у рыбы вес <20 кг? (да/нет)
да
вопрос – у рыбы длинное узкое тело? (да/нет)
нет
вопрос – у рыбы широкое тело? (да/нет)
да
вопрос – у рыбы темные полосы? (да/нет)
да
ваша рыба это окунь
end
```

2.5. Выполнить задание (Вариант 12).

Четыре человека играют в домино.

Их фамилии Кузнецов, Токарев, Слесарев и Резчиков.

Профессия каждого игрока соответствует фамилии одного из других игроков.

Напротив Кузнецова сидит слесарь.

Напротив Резчикова сидит резчик.

Справа от Слесарева сидит токарь.

Кто сидит слева от кузнеца?

Решение:

Идея решения основана на введении предиката atLeft, который выбирает имя и профессию двух людей, которые, с учетом ограничений

задачи, могут сидеть рядом. Далее предикат confirm обходит стол «вокруг», последовательно применяя предикат atLeft. Если найдено противоречие в рассадке игроков, то исходный ответ неверен.

```
implement main
    open core
constants
    className = "main".
    classVersion = "".
clauses
    classInfo(className, classVersion).

domains

class predicates
    name : (string) nondeterm anyflow.
    profession : (string) nondeterm anyflow.
    person : (string Name, string Prof) nondeterm anyflow.
    atLeft : (string Name, string Prof, string NameL, string ProfL) nondeterm
anyflow.
    confirm : (string Name, string Prof, string NameL, string ProfL)
nondeterm anyflow.

    ahead : (string Name, string AProf) nondeterm anyflow.
    right : (string Name, string RProf) nondeterm anyflow.

clauses
    name("Kuznetsov").          name("Tokarev").          name("Slesarev").
name("Rezchikov").
    profession("kuznets").    profession("tokar").    profession("slesar").
profession("rezchik").
    ahead("Kuznetsov","slesar"). ahead("Rezchikov","rezchik").
    right("Slesarev","tokar").

    person(X,Y):-
        name(X),profession(Y),X="Kuznetsov",Y<>"kuznets",Y<>"slesar";
        name(X),profession(Y),X="Tokarev",Y<>"tokar";
```

```
        name(X),profession(Y),X="Slesarev",Y<>"slesar",Y<>"tokar";
        name(X),profession(Y),X="Rezchikov",Y<>"rezchik".

    atLeft(X,Y,X1,Y1):-
        person(X1,Y1),
        person(X,Y),
        X<>X1,Y<>Y1,
        not(ahead(X1,Y)),
        not(ahead(X,Y1)),
        not(right(X,Y1)).

    confirm(X,Y,X1,Y1):-
        atLeft(X1,Y1,X2,Y2),X2<>X,Y2<>Y,
        atLeft(X2,Y2,X3,Y3),X3<>X,Y3<>Y,X3<>X1,Y3<>Y1,
        atLeft(X3,Y3,X,Y).

    run():-
        console::init(),
        atLeft(X,"kuznets",X1,Y1),
        confirm(X,"kuznets",X1,Y1),
        stdIO::writef("% is kuznets, at left % is %\n",X,X1,Y1),
        fail.
    run():-
        stdIO::write("End of test\n").

end implement main

goal
    mainExe::run(main::run).
```

Листинг 1. Решение задачи об игроках в домино.

Вывод программы:

Rezchikov is kuznets, at left Kuznetsov is tokar

End of test

Полученное решение соответствует исходным ограничениям.

2.6. Решить индивидуальное задание.

Три друга заняли первое, второе и третье места в соревнованиях универсиады. Друзья — разной национальности, зовут их по-разному и любят они разные виды спорта.

Майкл предпочитает баскетбол и играет лучше, чем американец. Израильтянин Саймон играет лучше теннисиста. Игрок в крикет занял первое место.

Кто является австралийцем? Каким видом спорта занимается Ричард?

```
implement main
    open core

domains
    human = friend(symbol, symbol, symbol).
    listOfFriends = human*.

class predicates
    newName : (symbol) multi (o).       % add new NAME in requirements
    newNational : (symbol) multi (o).    % --//--    (also)
    newSport : (symbol) multi (o).        % --//--
    better : (human, human, listOfFriends) nondeterm (o, o, i).  % someone
 is better than someone
    first : (human, listOfFriends) determ (o,  i).    % it`s FACT: who pla
y in cricet - first place
    q_national : (human, symbol) determ.           % set ratio "human-
nationality"
    q_name : (human, symbol) determ (i, i).  % set ratio "human-name"
    q_sport : (human, symbol) determ (i, i).   % set ratio "human-sport"
    solution : (listOfFriends) nondeterm (o).         % solving this task
    printList : (listOfFriends).   % print all list (uses printFriendInf)
    printFriendInf : (human).   % print information abut 1 human

clauses
% someone is better than someone
    better(A, B, [A, B, _]). better(A, C, [A, _, C]). better(B, C, [_, B,
C]).
% it`s FACT: who play in cricet - first place
    first(X, [X|_]).
% init arguments
    newName("Michael"). newName("Saymon"). newName("Richard").
    newNational("america"). newNational("australia"). newNational("israel"
).
```

```
    newSport("cricket"). newSport("basketball"). newSport("tenis").
% set ratios
    q_name(friend(E, _, _), E).
    q_national(friend(_, H, _), H).
    q_sport(friend(_, _, Z), Z).
% start solving
    solution([friend(E1, H1, C1), friend(E2, H2, C2), friend(E3, H3, C3)])
:-
        newName(E1), newName(E2), newName(E3),
        newNational(H1),  newNational(H2), newNational(H3),
        newSport(C1),  newSport(C2), newSport(C3),
        not(H1=H2), not(E1=E2), not(C1=C2),
        not(H1=H3), not(E1=E3), not(C1=C3),
        not(H3=H2), not(E3=E2), not(C3=C2),
        Friends=[friend(E1, H1, C1), friend(E2, H2, C2), friend(E3, H3, C3
)], %каждый friend имеет свое и, н, с
        better(F11, F12, Friends), q_name(F11, "Michael"), q_sport( F11, "
basketball"), q_national(F12, "america"),
        better(F21, F22, Friends), q_name(F21, "Saymon"), q_national(F21,
"israel"), q_sport(F22, "tenis"),
        first(F, Friends), q_sport(F, "cricket").
% recursive function for printing all list
    printList([]):- stdIO::nl().
    printList([X|Y]):- printFriendInf(X),  printList(Y).
    printFriendInf(friend(X, Y, Z)):-
stdIO::write(X,  " is ", Y,  ", he like to play in ", Z, "."), stdIO::nl()
.
 % start
    run() :- solution(Friends), printList(Friends), fail.
    run().
end implement main

goal
    console::runUtf8(main::run).
```

Результат:

```
C:\Users\michael\Documents\Visual Prolog
Projects\prog23_var2_version2\Exe>"C:\U
sers\michael\Documents\Visual Prolog
Projects\prog23_var2_version2\Exe\prog23_va
r2_version2.exe"
Saymon is israel, he like to play in cricket.
Michael is australia, he like to play in basketball.
Richard is america, he like to play in tenis.
```

3. Вывод

Пролог выглядит полезным языком, больше всего в данном языке привлекает факт того, что с помощью данного языка можно создать не просто базу данных, хранящую определенную информацию, а базу знаний. При обращении к такой базе могут быть произведены данные, которые в нее не вкладывались, на основе уже имеющихся фактов и установленных соотношений.

Для создания ИИ можно использовать такие языки как Lisp или Planner.

Мне кажется, что задача комбинаторного взрыва не решаема. Единственным способом (помимо локальных оптимизаций, в зависимости от задачи) является полный перебор возможных решений.

Разработка ИИ корректна, почему нет? Если мы можем сделать что-то полезное, создать что положительное – надо это создать. Вот если создавать ядерную бомбу – это очевидно пользы принести (в действии) не может (только запугать, что не начинать других войн), а создание искусственного интеллекта вещь творческая и опасная, но голосую ЗА.

Специально повторю этот вопрос отдельно: «Можно ли разработать ИИ не понимая, как он работает, должны ли мы понимать, как он работает, думает, рассуждает?»

Если на Земле появится второй вид существ, о которых мы должны будем понимать, что они думают и как устроен их ход мыслей – это будет перебор. Так да, определенно можно разработать (как нейронную сеть), а потом сидеть и ломать голову почему у ИИ все «шиворот на выворот»

Список литературы:

1. Бураков М.В. Язык логического программирования Prolog. СПб.: СПбГУАП, 2003.

2. Коста Э. Visual Prolog 7.1 для начинающих под ред. Е. А. Ефимовой. СПб., 2008.