Logic Programming – Laboratory 1
Prolog - Introduction

Isabela Drămnesc

1 Useful Links

- Course material.
- The laboratories.
- The student guide.
- Use: SWI-Prolog.
- SWI-Prolog – Useful material.

2 Concepts

- Logic programming
- Predicates
- Clauses
- Facts
- Rules
- Goals
- Variables
- Conjunctions

3 Prolog

– Logic programming: programs = a set of axioms; execution = a constructive proof for a goal.

– **PRO**gramming in **LOG**ic.

– Declarative programming - we describe what we want to solve, not how we want to solve (imperative programming).

– Deal with problems in terms of objects and relations between those objects.

– Use logic (first order predicate logic)

⇒ We show the relations between the objects using **predicates**.

– Programming în Prolog = a conversation with the interpretor:

- declare the facts (about the objects and the relations between them), about the facts we know that they are true. If we declare green(snow). the interpretor it will consider true either if it not corresponds to the real world.

  Example:

```
    mother(john,mary). /* john's mother is mary */

    father(john,albert).  /* john's father is albert */

 Or we can write:

    mother(mary,john). /* mary is the mather of john */
```

- define the rules (head and body)

  Example:

```
    children(john,mary,doru):-
                                 mother(john,mary),
                                 father(john,doru).

    Or:
    uncle(X,Y):-
                 father(Y,X),
                 brother(Y,Z),
                 mother_in_law(X,Z).
           /* X is the uncle of Y if:
           the father of Y is X and
           the brother of Y is Z and
           the mother_in_law of X is Z. */
    Or:
    son_in_law(X,Y):-
                       father(Z,Y),
                       married(X,Z).

    son_in_law(X,Y):-
                       mother(Z,X),
                       married(Y,Z).
```

- ask the interpretor

```
    mother(X,mary). /* Who's mother is mary? */

    son_in_law(X,X).  /* Who is his own son-in-law? */

    uncle(john,X). /* Who is the uncle of john? */
```

– $\underbrace{\text{Facts} + \text{Rules}}_{\text{Clauses}}$ = **Database**.

By defining some facts and some rules we construct (build) a model.

Use:

- Symbolic computation;
- Artificial intelligence;
- Natural language processing, etc.

3.1 Useful commands:

- Ctrl-D or **halt.** – exit the interpretor.
- **help(name-of-the-command).**
- **apropos(keyword).** – displays all the predicates, functions and sections in which exists the keyword.
- **consult(file-name).** or **[file-name].** – for loading the database from a file. Ex: [problem1]. or ['problem1.pl']. We can read more files in the same time: [problem1, problem2, problem3].
- **listing.** – displays the clauses from the database.
- **listing(predicate).** – displays al the clauses from the database which define the adequate predicate. Similar, but for more predicates in the same time: **listing([pred1, pred2, pred3]).**
- **;** – For obtaining more solutions (if there are more). (repeat the searching in order to obtain other valid solutions).
- **trace.** – Follow interactively each step of the execution.

3.2 Exercises:

```
1)

artist(emerese).
artist(denis).
austrian(emerese).

?-austrian(emerese).
?-artist(denis).
?-greek(emerese).

2)

eats(john, fish).
eats(john, pizza).
eats(mary, pie).
eats(mary, sandwish).
eats(john, sweets).
eats(victor, soup).
```

```
?-eats(mary, pie).
?-eats(john,X).
?-eats(X,Y).
?-eats(victor,Z).
```

```
3) -- introduce the database from ex2.

?-eats(john, fish),eats(fish,john).

?-eats(john,X),eats(mary,X).

?-eats(john,X),eats(mary,Y).
```

```
4)

male(albert).
male(edward).
female(alice).
female(victoria).
parents(edward, victoria, albert).
parents(alice, victoria, albert).

sister(X,Y):-
            female(X),
            parents(X,B,F),
            parents(Y,B,F).

?-sister(alice,edward).
?-sister(alice,X).
?-sister(X,X).
?-sister(X,Y).
```

Modify the database such that if we ask

```
?-sister(X,X).
```

we get 3 solutions: edward, alice, albert.

```
5)

likes(marian, beer).
likes(mariana, sweets).
likes(mariana, champagne).
likes(marian,X) :-
                likes(X,champagne).
likes(mariana,Y):-
                likes(Y,champagne).
```

```
?-likes(marian,mariana).
?-likes(marian,X).
?-likes(mariana,X).
?-likes(X,X).
```

6)
Create a database such that you get solutions for the next questions:
– When has alex the logic-programming course?
– In which day ecaterina has the algorithms course?
– Who has the logic-programming laboratory on monday?
– What courses has alex on wednesday?

Write at least one rule such that you get an answer for:
– Which are the mutual courses of alex and ecaterina?

3.3 Homework: (Deadline: next lab.)

Homework1