Practical "Introduction to Artificial Intelligence"

Prof. Dr. Gunter Grieser

Block 1: Prolog

Sheet 9: Database Manipulation and Collecting Solutions

Hints:

- *In Block 1 (Prolog) you do not have to submit your solutions to me. Just solve the excercies and discuss your problems and solutions.The aim of Block 1 is that you become familiar with the prolog programming.*
- *If you do not succed with a task, just delay it and try it again later. Some constructs need time to settle in the brain and will become easier as you get more experienced.*

Preparation (at home):

Read Chapter 11 of LearnPrologNow!.

Excercise 9.1

Read the SWI-prolog documentation for the relevant predicates, at least

- `assert, asserta, assertz`
- `retract, retractall`
- `listing`
- `findall, bagof, setof`

Excercise 9.2

Pseudo random numbers are computed by the formula

$$S = 9747 * S \bmod 131072$$

Write a predicate that computes the first 10 pseudo random numbers using `assert` and `retract`.

Excercise 9.3

You are the lucky to owner of a very expensive restaurant which has 3 tables. Because of the huge demand you have to launch a reservation system. It accepts reservations for the next day between 10:00 and 23:00 only. The fact, that table number `N` is reserved from `F` to `T` (both are integers meaning full hours) for a name `A` is modelled by the fact `isreserved(N,F,T,A)` in the prolog knowledge base.

Write predicates as follows:

- `reserve(Tablenr, From, To, Name)` reserves the table, but only if there is no other reservation at this time.
- `printreservations()` and `printreservations(TableNr)` prints all reservations resp. all reservations for a table.
- `storno(TableNr, From, To, Name)` deletes the reservation, but only if there is such a reservation.
- `vipreservation(TableNr, From, To, Name)` reserves a table in that it first deletes all existing reservations for this table within this time frame.
- `Clearreservations()` and `clearreservations(TableNr)` deletes all reservations resp. all reservations for a table.

Excercise 9.4

Reconsider Excercise 6.2 (resp. 8.4.) Change your program such that the computation of `successor_nodes` is done by `findall/bagof/setof`.

Excercise 9.5

Reconsider Excercise 8.2.
Write a predicate that reads in the rules from `family1.pl` and manually inserts it into the prolog knowledge base.
Verify the correctness by listing.
Execute the predicate twice and ensure, that the rules still exists only once in the knowledge base.

Excercise 9.6

Consider the following rules, save it as `family.rules`

```
fact f1 father anton john
fact f2 father paul mary
fact f3 father john peter
fact f4 father john elisabeth
fact f5 father peter agneta
fact f6 mother mary peter
fact f7 mother mary elisabeth
fact f8 mother elisabeth agneta
fact f9 mother elisabeth sarah

rule r1 if father(X,Y) then parent(X,Y)
rule r2 if mother(X,Y) then parent(X,Y)
```

a) Write a predicate that reads in the rules from `family.rules` and inserts it as prolog facts/rules into the prolog knowledge base, i.e. as:

```
father(anton,john).
father(paul,mary).
father(john,peter).
```

```
father(john,elisabeth).
father(peter,agneta).
mother(mary,peter).
mother(mary,elisabeth).
mother(elisabeth,agneta).
mother(elisabeth,sarah).

parent(X,Y):-father(X,Y).
parent(X,Y):-mother(X,Y).
```

Verify the correctness by `listing` and queries about `parent`.

b) Write a predicate that reads in the rules from `family.rules` and inserts it as prolog facts into the prolog knowledge base:

```
rule(f1, father(anton,john), []).
rule(f2, father(paul,mary), []).
rule(f3, father(john,peter), []).
rule(f4, father(john,elisabeth), []).
rule(f5, father(peter,agneta), []).
rule(f6, mother(mary,peter), []).
rule(f7, mother(mary,elisabeth), []).
rule(f8, mother(elisabeth,agneta), []).
rule(f9, mother(elisabeth,sarah), []).

rule(r1, parent(X,Y), [father(X,Y)]).
rule(r2, parent(X,Y), [mother(X,Y)]).
```

Apply `interprete` resp. `interprete2` from Excercise 8.2e. Does it work correctly?