Declarative problem solving methods

Lecture 5

Overview

Communication with files
Databases
Built-in predicates

References

Bratko (2012) Chapter 3, 6
Bratko (2001) Chapter 3, 6, 7

Communication with files

• Static predicates.

• Dynamic predicates.

```
:- dynamic father/2, mother/2, son/1.
```

Note! Communicating with files can work differently in different Prolog systems.

Work with files

tell – told

tell(File)
- opens a file for writing,
  creates one if the file does not exist
- changes the output stream to File
- if the file exists the old information is overwritten

told
- closes the file

see – seen

see(File)
- opens a file for reading
- changes the input stream

seen
- closes the file

Read and to use the content in a file.

`consult(File)`

- reads a file a loads it into the memory.

`reconsult(File)`

- updates the memory with the content in the file

Reconsult can be compared to [ ]

Ex:

```
? - ['Program.pl'].
```

Databases

Databases can be altered through:

`assert(Clause)`
- adds Clause to the program

`retract(Clause)`
- deletes Clause from the program

Note that *dynamic* must be specified to be able to change the clauses during the execution.

```
programming_with_database:-
   assert_database,
   open_listing_file,
   retract_from_database.

assert_database:-
   assert(mother(elisabeth)),
   assert(mother(eva)),
   asserta(mother(charlotte)),
      % asserta adds
      % the clause to be the first one
   assertz(mother(anna)),
      % assertz adds the
      % clause to be the last one
   assert(father(ebrahim)),
   assert(father(anders)),
   assert(father(erik)).
```

```
open_listing_file:-
   tell('InfoFile.pl'),
   write(':-dynamic mother/1, father/1.'),
   nl,listing(mother/1),
      % lists all clauses
      % mother(X) on the file
   listing(father/1),
   told.

retract_from_database:-
   retractall(mother(X)),
      % deletes all clauses mother(X)
   retract(father(Y)).
      % deletes the first clause father(Y)
```

```
?- assert_database.
yes

?- listing(mother).
mother(charlotte).
mother(elisabeth).
mother(eva).
mother(anna).
yes

?- listing(father).
father(ebrahim).
father(anders).
father(erik).
yes

?- retract_from_database.
yes

?- listing(mother).
yes

?- listing(father).
father(anders).
father(erik).
Yes
```

DON'T FORGET to empty the database sometimes. Otherwise the program may crash.

Storing all answers

Built-in predicates facilitating the information gathering from databases
`setof, bagof, findall`.
The information is stored in lists.

```
% born(Name,Year,Month,Day)
born(elisabeth, 1973, 7, 21).
born(eva, 1938, 6, 22).
born(charlotte, 1950, 12, 12).
born(pernilla, 1978, 3, 3).
born(anna, 1994, 3, 3).
born(marcus, 1997, 1, 2).
born(ebrahim, 1968, 3, 28).
born(anders, 1948, 5, 25).
born(erik, 1934, 5, 18).
```

setof(X, pred(X), L)

Stores all the instantiations of X that satisfies the predicate pred in list L. The instantiations are ordered and duplicate items are excluded. If a variable needs to be existentially quantified this is done with ^.

```
?- setof(X, Year^Day^born(X, Year, 03, Day),
Outlist).
```

```
?- setof(X, Year^born (X, Year, 03, Day),
Outlist).
```

```
?- setof(X, born(X, Year, 03, Day),
Outlist).
```

bagof(X, pred(X), L)

Stores all the instantiations of X that satisfies the predicate pred in a list L. The list is not ordered and may contain duplicate items.

```
?- bagof(X, Year^Day^born(X, Year, 03, Day),
Outlist)
```

```
?- bagof(X, born(X, Year, 11, Day),
Outlist).
```

```
?- bagof(Day,Name^Year^born(Name,
Year,3,Day),Outlist).
```

```
?- bagof(info(X, Year), Day^born(X, Year,
03, Day), Outlist)
Outlist = [info(pernilla,1978),
info(anna,1994), info(ebrahim,1968)]
```

findall(X, pred(X), L)

Collect all the instantiations of X in a list. Duplicate items are included. The variables need not to be existentially quantified.

If there are no elements that fulfils the question findall gives an empty list in return. Both setof and bagof give the answer no in this case.

```
?- findall(X, born(X, Year, 11, Day),
Outlist).
```

```
?- Month = 03, findall(X, born(X, Year,
Month, Day), Outlist).
```

```
?- findall(birthday(Name, Day, /, Month),
born(Name, Year, Month, Day), Outlist).

Outlist = [birthday(elisabeth,21,/,7),
birthday(eva,22,/,6),
birthday(charlotte,12,/,12),
birthday(pernilla,3,/,3),
birthday(anna,3,/,3),
birthday(marcus,2,/,1),
birthday(ebrahim,28,/,3),
birthday(anders,25,/,5),
birthday(erik,18,/,5)] ?
```

More built-in predicates

`write, read`

Predicates that can be used to read from the input stream and write to the output stream.

`var, nonvar, atom , atomic, integer`

The predicates check the type of variable or constant.

`name(Term,List)`

The predicate facilitates the handling of terms. A term's corresponding ASCII-code is stored in a list.

```
?- name('Prolog',L).
L = [80,114,111,108,111,103] ?
yes

?- name(Term,[97,98,99]).
Term = abc ?
yes
```