Alan Smaill
12th Oct 2015

Logic Programming (Programming)

Assessed Coursework

(updated for typo, 13th Oct)
(updated description of input to exit loop, 16th October)

This coursework is due at **3:00 pm, Monday 26th October**, 2015.
Submit your answers in a single file of Prolog source code, which will be tested using Sictus Prolog. To make your submission, use the following command on DICE:

```
=>  submit lp 1 <yourFile.pl>
```

Use `man submit` in DICE to find out more about the on-line submission process.

Note that questions may make use of material that will only be covered in the lecture of Thursday 15th October.

1. Write a predicate `prefixes(L,M)` that, given a list $\mathtt{L} = [\mathtt{X}_1, \mathtt{X}_2, \ldots, \mathtt{X}_n]$, calculates a list `M` of all of the prefixes of `L`:

$$[[\mathtt{X}_1, \ldots, \mathtt{X}_n], [\mathtt{X}_1, \ldots, \mathtt{X}_{n-1}], \ldots, [\mathtt{X}_1, \mathtt{X}_2], [\mathtt{X}_1], [\,]]$$

   For example:

```
?- prefixes([],M).
M = [[]].
?- prefixes([1,2],M)
M = [[1,2],[1],[]]
?- prefixes([1,1,1],M).
M = [[1,1,1],[1,1],[1],[]]
```

   For full credit, your solution should work whether or not `M` is ground. You may assume that `L` is ground when `prefixes/2` is called.

   [5 marks]

2. Write a Prolog procedure `square/0` that should read in an arithmetic term from standard input, and output a message repeating the input term, and giving the square of the input value, clearly marked as such.

   The program should loop, to allow repeated use. The loop should exit if the input is simply `q.` (followed by newline).

   [5 marks]

3. Define an operation that allows replacement of one term by another in a given term. More precisely, write a a predicate `replace/4` such that

```
replace(Term, Subterm, Subterm1, Term1)
```

holds if `Term1` is the result of replacing every occurrence of `Subterm` in `Term` by `Subterm1`.

Your solution should work when all arguments except `Term1` are ground, and also when all arguments except `Term` are ground.

For example:

```
?- replace( plus( 2, plus(3, 2)), 2, 17, Z ).
Z = plus(17,plus(3,17)) ? ;
no

?- replace( X, 2, 17, plus(17,plus(3,17)) ).
X = plus(2,plus(3,2)) ? ;
X = plus(2,plus(3,17)) ? ;
X = plus(17,plus(3,2)) ? ;
X = plus(17,plus(3,17)) ? ;
no
```

You may find the following built-in predicate `=..` useful:

`+Term =.. ?List  ?Term =.. +List`:
List is a list whose head is the atom corresponding to the principal functor of *Term*, and whose tail is a list of the arguments of *Term*.

[20 marks]

4. Consider a directed graph whose nodes are Prolog atoms and whose directed edges are described using a Prolog relation `r/2`, for example:

```
r(a,b).  r(b,c).  r(c,d).
r(d,a).  r(d,e).  r(d,f).
r(f,g).
```

A *simple path* in a graph is a non-empty list $L = [V_1, \ldots, V_n]$ such that:

(a) Each V_i is a ground atom that is a vertex in the graph, and

(b) No vertices are repeated (that is, $V_i \neq V_j$ whenever $i \neq j$), and

(c) For each i between 1 and $n - 1$, the adjacent pair (V_i, V_{i+1}) is a directed edge.

Write a predicate `paths(X,Paths)` such that, on success, $Paths$ is bound to a list containing all simple paths of the graph starting with X, without duplicates.

For example, given the data above:

```
?- paths(f,Paths).
Paths = [[f],[f,g]]
?- paths(a,Paths).
Paths =
  [[a],[a,b],[a,b,c],[a,b,c,d],
  [a,b,c,d,e],[a,b,c,d,f],[a,b,c,d,f,g]]
```

You may assume that X is a ground atom that is one of the vertices of the graph described by `r/2`. The order of the elements in $Paths$ does not matter. Duplicate elements should be removed.

[20 marks]