

CSE-505 Computing with Logic

Fall '05

Mid-Term Exam

Duration: 1h 20m

Oct 25, 2005

Max: 35 points

1. [8 points] Consider the following definite logic program:

$p(X,Y) :- a(X,Y).$
 $p(X,Y) :- a(X,Z), q(Z,Y).$

$q(X,Y) :- a(X,Z), p(Z,Y).$

$a(1,2).$
 $a(2,1).$
 $a(2,3).$
 $a(3,4).$
 $a(3,5).$
 $a(4,5).$

- (a) [2 points] What is the Herbrand Universe of the above program?
- (b) [2 points] What is the Herbrand Base of the above program?
- (c) [4 points] Using the immediate consequence (T_P) operator, find the least Herbrand model for the above program. Show each step of the iteration sequence used to compute the least model.
2. [8 points] Consider the program given in Question (1) above.
- (a) [2 points] Show the sequence of resolution steps in *some* successful SLD derivation for the query $p(3,A)$.
- (b) [2 points] Show the sequence of resolution steps in *some* successful SLD derivation for the query $q(2,B)$.
- (c) [2 points] Show the SLD derivation tree for the query $p(3,A)$ showing all derivations (successful, failed, or infinite). If the tree has infinite paths, indicate these paths and why they are infinite.
- (d) [2 points] Show the SLD derivation tree for the query $q(2,B)$ showing all derivations (successful, failed, or infinite). If the tree has infinite paths, indicate these paths and why they are infinite.
3. [4 points] Write a Prolog predicate `oneChange(L1, L2)` such that, given two lists L1 and L2, the predicate succeeds if and only if L2 can be obtained from L1 by *replacing* a single element. For instance:

- `oneChange([d,o,o,r], [d,o,o,m]),` `oneChange([d,o,o,m], [r,o,o,m]),`
 `oneChange([r,o,o,m], [r,o,a,m]),` and `oneChange([r,o,a,m], [r,o,a,d])` succeed
- `oneChange([r,o,a,m], [r,o,m,a]),` `oneChange([r,o,m,a], [r,o,m,a,n]),` and
 `oneChange([r,o,m,a,n], [o,m,a,n])` fail.

4. [10 points] Consider an alphabet with two function symbols: f with arity 2, and g with arity 1; and two constant symbols: a and b . Let \mathcal{T} be the set of all terms that can be constructed using these symbols *such that the argument of g is always a constant symbol*. (e.g. $f(g(a),g(b))$ is in \mathcal{T} but $f(g(g(a)),g(b))$ is not in \mathcal{T}).
- (a) [2 points] Write the set of all terms in \mathcal{T} that contain exactly 2 occurrences of symbol f .
- (b) [4 points] Write a Prolog predicate `constants(T,L)` such that, given a term T in \mathcal{T} , it returns in L the sequence of all constants in T in the order in which they appear. For instance:
- `constants(f(g(a), g(b)), L)` should return $L=[a,b]$ and
 - `constants(f(g(a), f(b,a)), L)` should return $L=[a,b,a]$.
- (c) [4 points] Given an input term with N symbols, how long does it take to compute the sequence of constants in the term using your predicate `constants`? If it is not $O(N)$, state precisely how you will change your definition above to make the computation linear in N .
5. [5 points] *Lev* is a simple-minded, one-dimensional robot which works in an infinitely tall building. *Lev* obeys two commands:
- **up**: *Lev* goes to the next higher floor. Since the building is infinitely tall, there is always a next higher floor.
 - **down**: *Lev* goes to the next lower floor. If *Lev* is already on the ground floor, it goes crazy and chases you shouting “Illegal! Illegal!”.

At the beginning, *Lev* is on the ground floor.

Let L be a sequence of commands (represented, of course, as a Prolog list) that you plan to give to *Lev*. Write a Prolog predicate `legal(L)` *without using any arithmetic* that succeeds if and only if:

- *Lev* will *not* go crazy when executing L , and
- after executing all the commands in L , *Lev* will be back on the ground floor.

For instance:

- `legal([])`, `legal([up, down])`, `legal([up,down,up,down])`, `legal([up, up, down, up, down,down])` should all succeed.
- `legal([down])`, `legal([up, down, down, up, up])`, `legal([up, up, down])` should all fail.

You may define and use any helper predicates you need. But neither the definition of `legal` nor those of your helper predicates can use any arithmetic operations.

END OF EXAM