Logic and Functional Programming

Labwork 9 – Answers

April 26, 2021

1 Warmup exercises

1. Consider the following logic program:

```
% thief(X) expresses the fact that X is thief
thief(bob).
% likes(X,Y) expresses the fact that X likes Y
likes(mary,candies).
likes(mary,wine).
likes(bob,X) :- likes(X,wine).
% may_steal(X,Y) expresses the fact that X may steal Y
may_steal(X,Y) :- thief(X), likes(X,Y).
```

The query

```
?-may_steal(bob,X).
```

asks Prolog to find all `X` that Bob may steal.

2. Assume the following relations have already been defined in a program:

   - `father(X,Y)` to indicate that `X` is the father of `Y`
   - `mother(X,Y)` to indicate that `X` is the mother of `Y`
   - `man(X)` to indicate that `X` is a man
   - `woman(X)` to indicate that `X` is a woman

   Extend this program with definitions of the following relations:

   (a) `parent(X,Y)` to indicate that `X` is a parent of `Y`
   (b) `isFather(X)` to indicate that `X` is a father
   (c) `isMother(X)` to indicate that `X` is a mother
   (d) `sister(X,Y)` to indicate that `Y` is the sister of `X`
   (e) `grandpa(X,Y)` to indicate that `X` is the grandpa of `Y`

```
parent(X,Y):-father(X,Y).
parent(X,Y):-mother(X,Y).
isFather(X):-father(X,_).
isMother(X):-mother(X,_).
sister(X,Y):-woman(X),parent(P,X),parent(P,Y),X\=Y.
grandpa(X,Y):-man(X),parent(X,P),parent(P,Y).
```

3. Consider the problem of finding all elements which appear in two given lists, by defining a predicate `member_both(X,L1,L2)` to hold if `X` is both an element of list `L1` and list `L2`.

```
member_both(X,L1,L2):-member(X,L1),member(X,L2).
```

4. Consider the problem of defining the relation `neighbor(X,Y)` for the fact that `X` is neighbor of `Y`. This relation is assumed to be symmetric: if `X` is neighbor of `Y`, then `Y` is neighbor of `X`.

   (a) How would you encode the following knowledge base: ”Alan is neighbor of Bob. Bob is neighbor of Caleb. Caleb is neighbor of Dan and Dick. Dan is neighbor of Erin.”

   (b) Write a query for the question “Who are the neighbors of Dan?” What answers will you get?

```
neighbor1(alan,bob).
neighbor1(bob,caleb).
neighbor1(caleb,dan).
neighbor1(caleb,dick).
neighbor1(dan,erin).
neighbor(X,Y):-neighbor1(X,Y).
neighbor(X,Y):-neighbor1(Y,X).

?- neighbor(dan,X).
```

5. `app([],L,L).`
   `app([H|T],L,[H|R]):-app(T,L,R).`

   (a) `app(L1,L2,[1,2,3,4])` computes all lists `L1`, `L2` whose concatenation is `[1,2,3,4]`.

   (b) `app(L,_,[1,2,3,4])` computes all prefixes `L` of `[1,2,3,4]`.

   (c) It is easy to observe that `S` is sublist of `L` if and only if it is a suffix of a prefix of `L`.

```
sublist(S,L):-app(P,_,L),app(_,S,P).
```

6. Consider the problem of arranging three 1's, three 2's, ..., three 9's in sequence so that for all $1 \leq i \leq 9$ there are exactly i numbers between successive occurrences of i. Use Prolog to define the relation `niceList(L)` for lists which have this property.

```
niceList(L):-
 L=[_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_],
 sublist([1,_,1,_,1],L),
 sublist([2,_,_,2,_,_,2],L),
 sublist([3,_,_,_,3,_,_,_,3],L),
 sublist([4,_,_,_,_,4,_,_,_,_,4],L),
 sublist([5,_,_,_,_,_,5,_,_,_,_,_,5],L),
 sublist([5,_,_,_,_,_,_,6,_,_,_,_,_,_,6],L),
 sublist([7,_,_,_,_,_,_,_,7,_,_,_,_,_,_,_,7],L),
 sublist([8,_,_,_,_,_,_,_,_,8,_,_,_,_,_,_,_,_,8],L),
 sublist([9,_,_,_,_,_,_,_,_,_,9,_,_,_,_,_,_,_,_,_,9],L).
```

Remark: By default, SWI-Prolog shows only the first 10 elements of long lists. To see more list elements, e.g.., 28 elements, run the query:

```
?-set_prolog_flag(answer_write_options,
   [quoted(true),portray(false),max_depth(28),spacing(next_argument)]).
```

This is a typical problem to be solved with the method of generate-and-test.

```
?- niceList(L).
L = [1,9,1,2,1,8,2,4,5,2,7,9,4,5,8,6,3,4,7,5,3,9,6,8,3,5,7] ;
L = [1,8,1,9,1,5,2,5,7,2,8,5,2,9,6,4,7,5,3,8,4,6,3,9,7,4,3] ;
L = [1,9,1,5,1,8,2,5,7,2,6,9,2,5,8,4,7,6,3,5,4,9,3,8,7,4,3] ;
L = [3,4,7,8,3,9,4,5,3,5,7,4,8,5,2,9,6,2,7,5,2,8,1,6,1,9,1] ;
L = [3,4,7,9,3,5,4,8,3,5,7,4,6,9,2,5,8,2,7,6,2,5,1,9,1,8,1] ;
L = [7,5,3,8,5,9,3,5,7,4,3,6,8,5,4,9,7,2,6,4,2,8,1,2,1,9,1] ;
false.
```

7. Consider the program defined by

```
part(_,[],[],[]).
part(X,[H|T],[H|L],R) :- H<X,part(X,T,L,R).
part(X,[H|T],L,[H|R]) :- H>=X,part(X,T,L,R).
```

   (a) Use SWI-Prolog to compute the answers to the queries

   ```
   ?-part(4,[1,7,3,5],L,R).
   ?-part(6,[10,1,3,7,5,9,20],L,R).
   ```

   (b) `?- part(X,Lst,L,R).`
   binds `L` to the list of elements in `Lst` smaller than `X`, and `R` to the list of elements in `Lst` greater than or equal to `X`.

2 Unification: exercises

1. f(X, Y, Z) and f(a, Z, h(a))

   ```
   f(X,Y,Z)=f(a,Z,h(a)) ⇒ X=a,Y=Z,Z=h(a) ⇒ X=a,Y=h(a),Z=h(a).
   ```

   We obtained the mgu {X → a, Y → h(a), Z → h(a)}.

2. f(g(X), g(c), Y) and f(g(g(Y)), X, a)

   ```
   f(g(X),g(c),Y)=f(g(g(Y)),X,a) ⇒ g(X)=g(g(Y)),g(c)=X,Y=a ⇒
   g(X)=g(g(a)),g(c)=X,Y=a ⇒ X=g(a),g(c)=X,Y=a ⇒
   X=g(a),g(c)=g(a),Y=a ⇒ X=g(a),c=a,Y=a ⇒ fail.
   ```

   These two terms are not unifiable.

3. f(h(b), X, X, Y) and f(h(b), g(Y), g(g(Z)), g(a))

   ```
   f(h(b),X,X,Y)=f(h(b), g(Y), g(g(Z)), g(a)) ⇒
   h(b)=h(b),X=g(Y),X=g(g(Z)),Y=g(a) ⇒
   b=b,X=g(Y),X=g(g(Z)),Y=g(a) ⇒
   b=b,X=g(Y),g(Y)=g(g(Z)),Y=g(a) ⇒ b=b,X=g(Y),Y=g(Z),Y=g(a) ⇒
   X=g(Y),Y=g(Z),Y=g(a) ⇒ X=g(g(a)),g(a)=g(Z),Y=g(a) ⇒
   X=g(g(a)),a=Z,Y=g(a) ⇒ X=g(g(a)),Z=a,Y=g(a)
   ```

   We obtained the mgu {X → g(g(a)), Y → g(a), Z → a}.