Assignment 1 — Solutions

1. [25 points] Write and test a Prolog program `q1.pl` that deals with family relations. Assume that the predicate `parent(X,Y)`, meaning that `X` is a parent of `Y` has already been defined (and that the definition is loaded separately).

   Your program's job is to define the following predicates:

   - `ancestor(X,Y)`, meaning that `X` is an ancestor of `Y`, i.e. either `X` is a parent of `Y` or `X` is an ancestor of someone who is a parent of `Y`;
   - `common_ancestor(X,Y,Z)`, meaning that `X` is a common ancestor of `Y` and `Z`, i.e. `X` is an ancestor of both `Y` and `Z`;
   - `closest_common_ancestor(X,Y,Z)`, meaning that `X` is a closest common ancestor of `Y` and `Z`, i.e. `X` is a common ancestor of `Y` and `Z` and no child of `X` is a common ancestor of `Y` and `Z`;
   - `ancestorList(X,Y,L)`, which holds iff `X` is an ancestor of `Y` and `L` is a list of the decendants of `X` (i.e., people whose ancestor is `X`) that are ancestors of `Y` ordered from the closest to the farthest from `X`; e.g. if `john` is a parent of `paul` who is a parent of `henry` who is a parent of `helen`, then `ancestorList(john,helen,L)` should succeed with `L = [paul,henry]` being returned;
   - `descendantTree(X,L)`, which holds iff `L` is a list structure representing the tree of all descendants of `X`; each node in the tree of descendants should be represented by a list whose first element is the node label and the remaining elements are the representations of its children if any; e.g. if `john`'s children are `paul` and `mary`, `paul`'s children are `henry` and `june`, `henry`'s only child is `helen`, `mary`'s only child is `adam`, and neither `adam` nor `june` nor `helen` have any children, then `descendantTree(john,L)` should succeed with

```
L =
[john,
  [paul,
    [henry,
      [helen]
    ],
    [june]
  ],
  [mary,
```

```
      [adam]
    ]
  ]
```

being returned (indentations and line breaks have been added here for readability, but your program should not do this).

You may define some auxiliary relations if that helps in defining the ones above. Test your program thoroughly before submitting it. Document your code appropriately. Do not include any `parent(X,Y)` facts in your file.

A solution is the following:

```
ancestor(X,Y) :- parent(X,Y).
ancestor(X,Y) :- parent(Z,Y), ancestor(X,Z).

common_ancestor(X,Y,Z) :- ancestor(X,Y), ancestor(X,Z).

closest_common_ancestor(X,Y,Z) :-
   common_ancestor(X,Y,Z), \+ has_child_common_ancestor(X,Y,Z).

% uses the following auxiliary predicate
has_child_common_ancestor(X,Y,Z) :-
   parent(X, XC), common_ancestor(XC,Y,Z).

ancestorList(X,Y,[]) :- parent(X,Y).
ancestorList(X,Y,[Z | R]) :- parent(X,Z), ancestorList(Z,Y,R).

descendantTree(X,[X]) :- \+ parent(X,_).

descendantTree(X,[X | L]) :- parent(X,_),
   setof(T, Y^(parent(X,Y), descendantTree(Y,T)), L).
```

Each part of the question is worth 5 marks, for a total of 25 marks.

2. [20 points] Write and test a Prolog program `q2.pl` that solves the following puzzle:

   The police are trying to track down the gang of three kids who have been stealing pumpkins. So far, they have established the following facts: the kids' first names are Angela, Mary, and David; one is 5, one is 7, and one

is 8; one has the last name Diamond, and the one with the last name Grant is 3 years older than the one with the last name Leung. You can assume Angela and Mary are female and David is male.

Use the technique shown in the zebra example discussed in class (the code is available on the course web page) to find missing information on the gang: each child's age, gender, first name and last name, consistent with the data above. (Encode the above data as is and do not add additional facts.)

Use your Prolog code to show whether or not the computed information uniquely identifies the culprits; submit these test results in the file `q2tests.txt`, together with the program file. Document your code appropriately.

The code submitted might look like the following:

```
   :- op(100,xfy,on).  % a bit of grammar

find(List) :-
   makeGang(3, List),
   kid('Angela', _, female, _) on List, %fact 1
   kid('David', _ , male,  _) on List,  %fact 2
   kid('Mary', _, female, _) on List,   %fact 3
   kid(_, _, _, 5) on List,             %fact 4
   kid(_, _, _, 7) on List,             %fact 5
   kid(_, _, _, 8) on List,             %fact 6
   kid(_,'Diamond', _, _) on List,      %fact 7
   kid(_,'Grant',_ , Age1) on List,     %fact 8
   kid(_, 'Leung', _, Age2) on List,    %fact 9
   Age1 is Age2 + 3.                    %fact 10



%********** DEFINITIONS ***************************

makeGang(0,[]).

makeGang(N,[kid(First, Last, Gender, Age) | List])
   :- N>0, N1 is N - 1, makeGang(N1,List).

X on List :- member(X, List).
```

(The use of the `on` operator is optional.)

Executing the code gives answers like the following:

```
?- find(List).

List = [kid('Angela', 'Leung', female, 5),
        kid('David', 'Diamond', male, 7),
        kid('Mary', 'Grant', female, 8)] ;

List = [kid('Angela', 'Leung', female, 5),
        kid('David', 'Grant', male, 8),
        kid('Mary', 'Diamond', female, 7)]
```

showing that the information on the gang members computable from the data does *not* uniquely identify the culprits.

3. [30 points] (Adapted from Genesereth and Nilsson (1987))

Victor has been murdered, and Arthur, Bertram, and Carleton are the only suspects (meaning exactly one of them is the murderer). Arthur says that Bertram was the victim's friend, but that Carleton hated the victim. Bertram says that he was out of town the day of the murder and besides, he did not even know the guy. Carleton says that he saw Arthur and Bertram with the victim just before the murder. You may assume that everyone, except possibly the murderer, is telling the truth.

a) (5 points) Write sentences in first-order logic that represent this knowledge. Also provide a glossary where you indicate the intended meaning of your predicate, function, and constant symbols in English.

In my solution I use the following symbols:

$m(X)$ means X has murdered Victor.

$f(X, Y)$ means X is a friend of Y.

$w(X, Y)$ means X was with Y at the time of the murder.

a, b, c, and v denote Arthur, Bertram, Carleton, and Victor, respectively.

(1) $m(a) \vee m(b) \vee m(c)$

(2) $\forall X \forall Y . m(X) \wedge m(Y) \rightarrow X = Y$

or

$(\neg m(a) \vee \neg m(b)) \wedge (\neg m(a) \vee \neg m(c)) \wedge (\neg m(b) \vee \neg m(c))$

(3) $m(a) \rightarrow f(b, v)$

(4) $\neg m(a) \rightarrow \neg f(c, v)$

(5) $\neg m(b) \rightarrow \neg f(b, v)$

(6) $\neg m(b) \rightarrow \neg w(b, v)$

(7) $\neg m(c) \rightarrow w(a, v)$

(8) $\neg m(c) \rightarrow w(b, v)$

b) (5 points) Convert the sentences into clausal form and give the resulting set of clauses.

(1) $(m(a), m(b), m(c))$
(2) $(\neg m(a), \neg m(b))$
(3) $(\neg m(a), \neg m(c))$
(4) $(\neg m(b), \neg m(c))$
(5) $(m(a), f(b, v))$
(6) $(m(a), \neg f(c, v))$
(7) $(m(b), \neg f(b, v))$
(8) $(m(b), \neg w(b, v))$
(9) $(m(c), w(a, v))$
(10) $(m(c), w(b, v))$

c) (12 points) Use resolution with answer extraction to find the murderer. State how you represent the query in first-order logic and what clause (with an answer predicate) is added to the theory. Show the complete resolution derivation (in sequence or tree form), clearly indicating which literals/clauses are resolved and the unifier used.

The query in FOL is $\exists X m(X)$ (1 mark).

The query in clausal form with the answer predicate is

(11) $(\neg m(X), ans(X))$ (1 mark)

The derivation in sequence form is:

(12) $R[5b, 7b]$ $(m(a), m(b)$
(13) $R[8b, 10b]$ $(m(b), m(c))$
(14) $R[14b, 3b]$ $(m(b), \neg(m(a))$
(15) $R[15b, 12a]$ $(m(b))$
(16) $R[15, 11a]\{X = b\}$ $(ans(b)$

d) (8 points) Suppose that we can no longer assume that there was only a single murderer. What sentences must you remove from the theory? Show that the modified theory no longer entails that the answer you obtained in c) is the murderer. Do this by specifying an interpretation where the answer in c) is not the murderer and showing that it satisfies all the axioms of the theory.

One must remove clauses(2), (3), and (4) from the theory in part b) (1 mark).

Consider the interpretation $I = \langle D, \Phi, \Psi, v\rangle$ where

- $D = \{a, b, c, v\}$,
- $\Phi(x) = x$ for all $x \in D$,
- $\Psi(m) = \{a, c\}$,
- $\Psi(f) = \{\}$,
- $\Psi(w) = \{\}$.

Then it is easy to show that I satisfies all the remaining clauses in the theory. I satisfies (1), (5) and (6) because $I \models m(a)$. I satisfies (9) and (10) because $I \models m(c)$. I satisfies (7) and (8) because $I \models \neg f(b, v)$ and $I \models \neg w(b, v)$.