Master MLDM - First year - December 14, 2023
Introduction to Artificial Intelligence - Exam on Prolog

Maximum time allocated: 3h00 - No documents allowed. TAKE CARE: any cheating will be severely punished and will lead to a formal complaint to the disciplinary council of the university.

1 Vocabulary (2 points)

Answer the following questions, related to the Prolog language, in one or two sentences.

1. What is a Horn clause?
2. What is a fact?
3. What is a rule?
4. What is a recursive clause?
5. What is a compound term?
6. What is the Closed-World Assumption?
7. What is the Occurs Check?
8. What does DCG mean?

2 Proof tree (5 points)

Consider the Prolog program below:

```
p1(A, B, C) :- q(A, B), r(B, C).
p1(A, B, C) :- r(A, B), q(B, C).

q(A, B) :- s(A), t1(B).

r(A, B) :- v(A), t2(B).
r(A, B) :- t2(A), v(B).

s(a).  s(b).  s(d).
v(d).  v(e).  v(f).

t1(42). t1(21).
t2(d).  t2(c).
```

1. Draw the proof tree of the resolution of the goal: `?- p1(X,d,Y).` and give all the solutions for this goal.
2. Suppose we put a cut between `q(A,B)` and `r(B,C)` in the first clause of the program. Show, on the tree you built at the previous question, which branches are pruned during the resolution of the goal: `?- p1(X,d,Y).` and give again all the solutions for this goal.
3. Suppose we also put a cut between `v(A)` and `t2(B)` in the fourth clause of the program. Show, on the tree you built at the previous question, which branches are pruned during the resolution of the goal: `?- p1(X,d,Y).` and give again all the solutions for this goal.

3 Writing functions in Prolog (2 points)

The Ackermann function is defined by: $ack(m,n) = \begin{cases} n+1 & if\ m = 0 \\ ack(m-1,1) & if\ m > 0\ and\ n = 0 \\ ack(m-1, ack(m,n-1)) & if\ m > 0\ and\ n > 0 \end{cases}$

The Mldm function is defined by: $mldm(m) = \begin{cases} 1 & if\ m = 0 \\ 2 & if\ m = 1 \\ mldm(m-2) * mldm(m-1) & if\ m > 1 \end{cases}$

1. Write the Prolog program that defines a predicate `ack/3` where `ack(M,N,Res)` is true if `Res` is the result of the Ackermann function of `M` and `N`.
2. Write the Prolog program that defines a predicate `mldm/2` where `mldm(M,Res)` is true if `Res` is the result of the Mldm function of `M`.

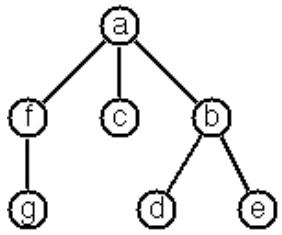
4 Unification (2 points)

For each of the goals below, say whether it is true or false. In cases where it makes sense, give the value(s) of the variable(s) that make(s) a goal true.

1. `?- X = 3, X = 4.`
2. `?- p(A,B,A) = p(a,b,b).`
3. `?- p(A,[B,A|C],A) = p(a,[b,a,c,d,e],a).`
4. `?- p(A,[B,A|C],f(D,g(k))) = p(a,[k,a,c,d,e],f(E,g(B))).`
5. `?- X = 2 + 1.`
6. `?- s(a,s(b,s(c,s(d,s(e,s(n))))))) = s(X,s(b,s(Y,Z))).`
7. `?- [a|[b|[c|[d|[e|[[]]]]]]] = [X, Y, Z, T, U |W].`
8. `?- [a,[b,[c,[d,e],f],g],h,i,[j],[],k]=[X,Y|T], T=[A,B|C], C = [D,E,F].`

5 Trees in Prolog (5 points)

In this exercice we consider n-ary trees where a tree is represented by a compound term `t(Value,List)` where Value is the root value of the tree `t` and `List` is the list of all the subtrees of t. For example, the compound term `t(a,[t(f,[t(g,[])]),t(c,[]),t(b,[t(d,[]),t(e,[])])])` is a Prolog representation of the following tree:



Write the Prolog program specifying the following predicates:

1. `is_a_tree/1`. `is_a_tree(T)` is true if `T` is a tree of the form `t(V,L)` where `V` is a value and `L` is a list of trees, possibly empty.
   For example:
   ```
   ?- is_a_tree(t(a,[t(f,[t(g,[])]),t(c,[]),t(b,[t(d,[]),t(e,[])])])).
   true.
   ?- is_a_tree(t(a,[t(f,[t(g,h)]),t(c,[]),t(b,[t(d,[]),t(e,[])])])).
   false.
   ```

2. `count_nodes/2`. `count_nodes(T,N)` is true if `T` is a tree that has `N` nodes.
   For example:
   ```
   ?- count_nodes(t(a,[t(f,[t(g,[])]),t(c,[]),t(b,[t(d,[]),t(e,[])])]),N).
   N = 7
   ?- count_nodes(t(a,[t(f,[t(g,[])]),t(c,[]),t(b,[t(d,[]),t(e,[])])]),7).
   true.
   ?- count_nodes(t(a,[t(f,[t(g,[])]),t(c,[]),t(b,[t(d,[]),t(e,[])])]),8).
   false.
   ```

3. `length_internal_path/2`. `length_internal_path(T,N)` is true if in the tree `T` the sum of the lengths of the paths to each of the nodes is equal to `N`.
   For example:
   ```
   ?- length_internal_path(t(a,[t(f,[t(g,[])]),t(c,[]),t(b,[t(d,[]),t(e,[])])]),L).
   L = 9
   ```

4. `depth_first/2`. `depth_first(T,L)` is true if `L` is the list of the nodes of the tree `T` traversed depth first.
   For example:
   ```
   ?- depth_first(t(a,[t(f,[t(g,[])]),t(c,[]),t(b,[t(d,[]),t(e,[])])]),L).
   L = [g, f, c, d, e, b, a].
   ```

6 Knowledge base modeling and querying (4 points)

Convert the following information into a Prolog program:
1) For any person, if that person is young and plays the guitar then that person is happy.
2) For any person, if that person is old and plays the violin then that person is happy.
3) For any person, if that person plays the drums then that person is happy.
4) Mary is young, owns a guitar, has learned the guitar and loves Paul.
5) John is old and he knows how to dance.
6) For any person and musical instrument, if that person owns that musical instrument and has learned to play it then that person plays that instrument.
7) For any person and musical instrument, if that person is a genius and that musical instrument is a string instrument, then that person plays that instrument.
8) For any person and musical instrument, if that person person has built that musical instrument then that person plays that instrument.
9) Lindsey has built a violin.
10) The guitar and violin are string instruments.
11) George is a genius.
12) For any person p1 and p2,
    a) If p1 is happy and loves p2, then p1 is a happy lover of p2.
    b) If p1 knows how to dance and p2 is happy then p1 is a happy dancer with p2.
    c) If p1 is a happy dancer with p2 then p1 dances with p2.
    d) If p1 is a happy lover of p2 then p1 dances with p2.
    e) If p1 has built an instrument, then p1 dances with this instrument.

Considering the Prolog program you just wrote, convert the following english queries into Prolog goals (do not try to solve those goals, this is not the question):

1. Who dances with whom/what?
2. Who is a happy lover of Mary?
3. Who plays violin or drums?
4. Who is happy?