Fundamentals of Artificial Intelligence and Knowledge Representation –

academic year 2022—2023: Module 2 (Chesani)

previous academic years: Module 2 (ex-Gaspari), Module 4 (Chesani)

Prof. Federico Chesani – 12th of June, 2023

Available time: 1h.

1) The candidate is invited to define a Prolog predicate **`filter/3`**, that receives in input two lists **`L1`** and **`L2`** of ground terms, and returns in output a list **`L3`**. The list **`L3`** will contain the terms of **`L1`** that appears strictly more than once in **`L2`**.

   For example, if invoked with:
   ```
   :- filter([a, 3, b], [a, a, p(X), 3], L3).
   ```
   The expected outcome is:
   ```
   Yes, L3 = [ a ]
   ```

2) The candidate is invited to define a meta-interpreter for the Prolog language, where whenever a special predicate **`break/0`** is encountered, the meta-interpreter directly solve it by asking the user if she wants to continue. The evaluation of subgoal break/0 will succeed only if the user types 'yes'.
   To this end, the candidate can use the predicate **`read(Term)`**, that "Read the next Prolog term from the current input stream and unify it with Term."

3) The candidate is invited to briefly introduce the three different approaches (presented in the course), to deal with the reasoning with temporal information.

4) The candidate is invited to briefly introduce the Description Logics, and in particular the principal operators of the ALC fragment (AND operator; ALL operator; [EXISTS 1 r] operator; concept complement (negation)).

Fundamentals of Artificial Intelligence and Knowledge Representation –

academic year 2022—2023: Module 2 (Chesani)

previous academic years: Module 2 (ex-Gaspari), Module 4 (Chesani)

Prof. Federico Chesani – 12th of June, 2023

Available time: 1h.

Solution

1) The candidate is invited to …

```
filter([], _, []).
filter([H|T], L2, [H|Rest]) :-
    count(H, L2, N),
    N > 1,
    !,
    filter(T, L2, Rest).
filter([_H|T], L2, Rest) :-
    filter(T, L2, Rest).

count(_, [], 0).
count(X, [X|T], N) :-
    !,
    count(X, T, N1),
    N is N1+1.
count(X, [_|T], N) :-
    count(X, T, N).
```

2) The candidate is invited …

```
solve(true) :- !.
solve((A,B)) :-
    !, solve(A), solve(B).
solve(break) :-
    !, write('Do you want to continue? '), read(yes).
solve(X) :-
    clause(X, Body),
    solve(Body).
% Example program:
p(X) :- q(X).
q(X) :- break, r(X).
r(1).
```

3) *The candidate is invited …*

See the slides

4) *The candidate is invited …*

See the slides