## TLLP interpreter in Prolog

```
/*
/ *
       TLLP interpreter in Prolog
                                                    TLLP interpreter in Prolog
*/
                                           * /
:- op(1060, xfy, (\&)).
                                           :- op(1060, xfy, (\&)).
:- op( 950, xfy, [-<>, =>]).
                                           :- op( 950, xfy, [-<>, =>]).
:- op(900, fy, [!, @, #]).
                                           :- op(900, fy, [!, @, #]).
prove(G) :-
                                           prove(G):-
       prove(G, 0, [], []).
                                                    prove(G, 0, [], []).
prove(true, T, I, I) :- !.
                                           prove(true, T, I, I) :- !.
prove(top, T, I, 0) :-!,
                                           prove(top, T, I, O) :-!,
       subcontext(T, 0, I).
                                                    subcontext(T, O, I).
prove((G1, G2), T, I, 0) :-!,
                                           prove((G1, G2), T, I, O) :-!,
       prove(G1, T, I, M),
                                                    prove(G1, T, I, M),
       prove(G2, T, M, O).
                                                    prove (G2, T, M, O).
prove((G1 & G2), T, I, 0) :-!,
                                           prove((G1 & G2), T, I, O) :-!,
       prove(G1, T, I, 0),
                                                    prove(G1, T, I, O),
       prove(G2, T, I, 0).
                                                    prove (G2, T, I, O).
prove((G1; G2), T, I, 0):-!,
                                           prove((G1; G2), T, I, O):-!,
       (prove(G1, T, I, 0);
                                                     (prove(G1, T, I, O);
        prove(G2, T, I, 0)).
                                                     prove (G2, T, I, O)).
prove((R -<> G), T, I, 0) :-!,
                                           prove((R -<> G), T, I, O) :- !,
       count_next(R, N, R1),
                                                    count next(R, N, R1),
       T1 is T + N,
                                                    T1 is T + N,
       prove(G, T, [(R1,T1)|I], [1|0]).
                                                    prove (G, T, [(R1,T1)|I], [1|0]).
prove((S => G), T, I, 0) :- !,
                                           prove ((S => G), T, I, O) :- !,
       prove(G, T, [(!S,0)|I], [(!S,0)|O]).
                                                    prove (G, T, [(!S, 0)|I], [(!S, 0)|O]).
prove(!G, T, I, I) :-!,
                                           prove(!G, T, I, I) :-!,
       prove(G, T, I, I).
                                                    prove(G, T, I, I).
prove(@G, T, I, 0) :-!,
                                           prove(@G, T, I, I) :-!,
       T1 is T + 1,
                                                    T1 is T + 1,
       prove(G, T1, I, 0).
                                                    prove(G, T1, I, O).
prove(A, T, I, 0) :-
                                           prove(A, T, I, O) :-
       pick(T, I, O, A).
                                                    pick(T, I, O, A).
prove(A, T, I, 0) :-
                                           prove(A, T, I, O) :-
       pick(T, I, M, (G - <> A)),
                                                    pick(T, I, M, (G -<> A)),
       prove(G, T, M, O).
                                                    prove (G, T, M, O).
```

```
count next (R, N, R1) :- !
count_next(@R, N, R1) :- !,
                                                          count next(R, N1, R1),
       count_next(R, N1, R1),
                                                         N is N1 + 1.
      N is N1 + 1.
                                                 count next (R, 0, R).
count_next(R, 0, R).
                                                 pick(T, I, O, S) :-
pick(T, I, 0, S) :-
                                                          pick1(T, I, O, S).
       pick1(T, I, O, S).
                                                 pick(T, I, I, S) :-
pick(T, I, I, S) :-
                                                          rule(S).
      rule(S).
                                                 pick(T, I, I, (G -<> A)) :-
pick(T, I, I, (G -<> A)) :-
                                                          rule((A :- G)).
      rule((A :- G)).
                                                 pickl(T, [(!S,0)|I], [(!S,0)|I], S).
pick1(T, [(!S,0)|I], [(!S,0)|I], S).
                                                 pick1(T, [(\#R,T0)|I], [1|I], S) :-
pick1(T, [(#R,T0)|I], [1|I], S) :-
                                                          T >= T0,
      T >= T0
                                                          select(R, S).
       select(R, S).
pick1(T, [(R,T)|I], [1|I], S) :-
                                                 pick1(T, [(R, T)|I], [1|I], S) :-
                                                          +(R = (!_{})), +(R = (#_{})),
                                                          select(R, S).
      select(R, S).
pick1(T, [R|I], [R|O], S) :-
                                                 pick1(T, [R|I], [R|O], S) :-
                                                          pick1(T, I, O, S).
       pick1(T, I, 0, S).
select((R1 & R2), R) :-!,
                                                 select((R1 & R2), R) :-!,
       (select(R1, R); select(R2, R)).
                                                          (select(R1, R); select(R2, R)).
select(R, R).
                                                 select(R, R).
subcontext(T, [], []).
                                                 subcontext(T, [], []).
subcontext(T, [(!S,0)|0], [(!S,0)|I]) :-
                                                 subcontext(T, [(!S,0)|O], [(!S,0)|I]) :-
       subcontext(T, 0, I).
                                                          subcontext(T, O, I).
subcontext(T, [R1|0], [(#R,T0)|I]) :-
                                                 subcontext(T, [R1|0], [(\#R,T0)|I]) :-
       (R1 = (\#R, T0) ; R1 = 1),
                                                          (R1 = (\#R, T0) ; R1 = 1),
       subcontext(T, 0, I).
                                                          subcontext(T, O, I).
subcontext(T, [R1|0], [(R,T0)|I]) :-
                                                 subcontext(T, [R1|0], [(R,T0)|I]) :-
      +(R = (!_{})), +(R = (#_{})),
                                                          TO >= T,
                                                          T0 >= T
       (R1 = (R,T0) ; R1 = 1),
                                                          (R1 = (R, T0); R1 = 1),
       subcontext(T, 0, I).
                                                          subcontext(T, O, I).
rule((p(V,V,[V]):-v(V))).
                                                 rule((p(V, V, [V]) :- v(V))).
rule(( p(U,V,[U|P]) :-
                                                 rule((p(U,V,[U|P]):-
       v(U), e(U,W), @p(W,V,P))).
                                                           V(U), e(U,W), @p(W,V,P))).
rule(( e(U,V) )).
                                                 rule(( e(U, V) )).
rule(( goal(P) :- #v(a) -<> @ @v(b) -<>
                                                 rule(( goal(P) :- #v(a) -<> @ @v(b) -<>
        @ #v(c) -<> #v(d) -<> p(a,d,P) )).
                                                           \emptyset #v(c) -<> #v(d) -<> p(a,d,P)).
```