```
%********* Best First Search Strategy ******************

best_first(State,[State]) :- goal(State),!.
best_first(State,[State|Path]) :-
        findall(Dist-Next_State,(cost(State,Next_State,_),
h(Next_State,Dist)),States),
        keysort(States,[_-BestState|_]),
        best_first(BestState,Path).


%********** A* search algorithm *************************
solve(Start,Soln) :- f_function(Start,0,F),
              search([F-Start],Soln).

f_function(State,D,F) :- h(State,H),
                  F is D + H.

search([_-State| _], [State]) :- goal(State),!.
search([F-State|R],[State|Path]) :- expand(F-State, Children),
             insert_all(Children, R, NewOpen),
             search(NewOpen,Path).

expand(_-State, All_My_Children) :-
      findall(F-Child,
          (cost(State,Child,D),
           f_function(Child,D,F) ) ,
          All_My_Children).

insert_all([],Open,Open).
insert_all([F|R],Open1,Open3) :- insert(F,Open1,Open2),
                      insert_all(R,Open2,Open3).

insert(B,[],[B]).
insert(_-Node,[_-Node|R],[_-Node|R]) :- ! .
insert(B,[C|R],[B,C|R]) :- cheaper(B,C), ! .
insert(B,[B1|R],[B1|S]) :- insert(B,R,S), !.

cheaper( H1-_ , H2-_) :- H1 <  H2.
```