Lab5.pl

/\*

1. what fluents/predicates

2. what are the things that can happen?

3. write down a precondition axiom for each action, in this case, there is just one

poss([move(X, Y)| S]):-

clear(X, S),

clear(Y, S),

block(X),

block(Y),

position(Y),...

4. foreach fluent write down successor state axiom.

successor state axiom:

on(X, Y, [A|S]):-

poss([A|S]),

(

%becomming true with A

A = move(X, \_, Y);

% was true and not becomming false

on(X, Y, S),

A \= move(X, \_, Y).

)

clear(X, S):-\+on(\_, X, S).

5. initial state

on(a, 1, []).

on(b, a []).

on(c, 3, []).

\*/

block(a).

block(b).

block(c).

position(p1).

position(p2).

position(p3).

position(p4).

% posOrBlock(X) is true if X is a position or a block

posOrBlock(X) :- block(X);position(X).

% Poss(move(B, X, Y),S) is true if Block B could be moved from Position X to % Position Y based % on the situation S

poss([move(B, X, Y)|S]):- block(B), posOrBlock(X), posOrBlock(Y),

dif(B,X),

dif(B,Y),

dif(X,Y),

on(B, X, S),

\+on(\_, B, S),

\+on(\_, Y, S).

% on(B, X, S) means that Block B is on Position or Block X in Situation S

on(a, p1, []).

on(b, p3, []).

on(c, a, []).

%fluent is a predict comparing to the situation

%\_\_\_\_fluent\_\_\_\_

%| |

on(B, X, [A|S]):-

poss([A|S]),

(A = move(B, \_, X);

on(B,X,S), % already was true before doing A

A \= move(B, X, \_)).

%plan(g(S),S) is true when g(S) is a formula (the goal state depending on S) that is true, and S is an action history (the plan)

plan(Goal,Plan):-bposs(Plan),Goal.

%bposs(S) is true when S is a sequence of possible actions considering shortest sequences first

bposs(S) :- tryposs([],S).

% tryposs(S,S) is true when S is a sequence of possible actions considering shortest sequences first (breadth first search for possible action sequences)

tryposs(S,S) :- poss(S).

tryposs(X,S) :- tryposs([\_|X],S).

Testing promp

?- plan(on(a, p1, S), S).

**S = [move(b, p3, a)] ;**

**S = [move(b, p3, c)] ;**

**S = [move(b, p3, p2)] ;**

**S = [move(c, p4, a)] ;**

**S = [move(c, p4, b)] ;**

**S = [move(c, p4, p2)] ;**

**S = [move(a, b, p1), move(a, p1, b)] ;**

**S = [move(a, c, p1), move(a, p1, c)] .**

?- plan(on(a, p1, [move(b, p3, c)]),[move(b, p3, c)]).

**true ;**