Programming Assignment 2

In this second programming assignment, you will again use Clingo, this time to solve more complex problems, specifically four problems involving actions and effects. You will use ASP to formulate transition systems involving actions and effects, decisions, etc.

NOTE: This graded assignment's problems are again drawn from the week's lectures. In fact, the four questions you will answer are the same four questions covered in the "Blocks World in ASP" video, which includes a walkthrough of the problems' steps in clingo.

Q1. Consider the clingo program blocks.lp below that is introduced in Module 4.

```
%%%%%%%%%%%%%%%%%%%
% File: blocks.lp: Blocks World
%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% sort and object declaration
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% every block is a location
location(B) :- block(B).

% the table is a location
location(table).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% state description
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% two blocks can't be on the same block at the same time
:- 2{on(BB,B,T)}, block(B), T = 0..m.


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% effect and preconditions of action
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% effect of moving a block
on(B,L,T+1) :- move(B,L,T).

% concurrent actions are limited by num of grippers
:- not {move(BB,LL,T)} grippers, T = 0..m-1.
```

```
% a block can be moved only when it is clear
:- move(B,L,T), on(B1,B,T).

% a block can't be moved onto a block that is being moved also
:- move(B,B1,T), move(B1,L,T).


%%%%%%%%%%%%%%%%%%%%%%%%%%%
% domain independent axioms
%%%%%%%%%%%%%%%%%%%%%%%%%%%

% fluents are initially exogenous
1{on(B,LL,0):location(LL)}1 :- block(B).

% uniqueness and existence of value constraints
:- not 1{on(B,LL,T)}1, block(B), T=1..m.

% actions are exogenous
{move(B,L,T)} :- block(B), location(L), T = 0..m-1.

% commonsense law of inertia
{on(B,L,T+1)} :- on(B,L,T), T < m.

#show move/3.
```

Modify the file blocks.lp to reflect the assumption that the table is small, so that the number of blocks that can be placed on the table simultaneously is limited by a given constant. How many steps are required to solve the example problem above if only 4 blocks can be on the table at the same time? What if only 3? You may test your codes with a scenario, which is also represented by a clingo program such as blocks-scenario.lp below.

```
%%%%%%%%%%%%%%%%%%%
% File: blocks-scenario.lp
%%%%%%%%%%%%%%%%%%%
block(1..6).
% initial state
:- not on(1,2,0; 2,table,0; 3,4,0; 4,table,0; 5,6,0; 6,table,0).
% goal
:- not on(3,2,m; 2,1,m; 1,table,m; 6,5,m; 5,4,m; 4,table,m).
```

Q2. The file blocks.lp above specifies that the initial state correctly. Without the specification, there will be stable models that do not correspond to valid states, like the following.
on(1,2,0) on(2,1,0) on(3,3,0) on(4,table,0) on(5,6,0) on(6,table,0)
Modify the file blocks.lp so that the stable models are in a 1-1 correspondence with valid states. How many valid states are there?

Q3. A serializable plan is such that the actions that are scheduled for the same time period can be instead executed consecutively, in any order without affecting the result.
Modify blocks.lp to generate only serializable plans. Find a minimal length plan for the following scenario. (Hint: you need to modify blocks-scenario.lp to reflect this new scenario.)

Initially:
loc(m)=table, loc(l)=m, loc(a)=l, loc(b)=a, loc(c)=b,
loc(o)=table, loc(n)=o, loc(d)=n, loc(e)=d, loc(j)=e,
loc(k)=j, loc(f)=table, loc(g)=f, loc(h)=g, loc(i)=h
In maxstep:
loc(e)=j, loc(a)= e, loc(n)=a, loc(i)=d, loc(h)=i,
loc(m)=h, loc(o)= m, loc(k)=g, loc(c)=k, loc(b)=c,
loc(l)=b.

Q4. A minimal length plan is not necessarily optimal. Modify the program done for Problem 3 to find a plan that has the least number of actions. What is that number when maxstep m is 8, 9, and 10?