Name: Swarali Chine

ASU id: 1222583687

# Problem 1

**CSE 579**

# Module 5 Graded Assignment Template for clingo Work

|  |  |
| --- | --- |
| Input | Hint: you only need one program with a new term, whose value will be |
| Program | assigned to 3 or 4 in the command line. |
|  | %%%%%%%%%%%%%%%%%%% % File: q1\_blocks.txt: 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 inde-pendent |
| 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. |
| % space on table is limited |
| :- not {on(B,table,T)}s, T=0..m. |
| #show move/3. |
| %%%%%%%%%%%%%%%%%%% |
| % File: blocks-scenario.txt |
| %%%%%%%%%%%%%%%%%%% |
| 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). |
| Command Lines | You should write multiple command lines below. |
|  | clingo q1\_blocks.txt blocks-scenario.txt -c grippers=2 -c s=3 -c m=4 0 |
|  | clingo q1\_blocks.txt blocks-scenario.txt -c grippers=2 -c s=3 -c m=5 0 |
|  | clingo q1\_blocks.txt blocks-scenario.txt -c grippers=2 -c s=4 -c m=3 0 |
|  | clingo q1\_blocks.txt blocks-scenario.txt -c grippers=2 -c s=5 -c m=4 0 |

|  |  |
| --- | --- |
| Outputs of clingo | You should write multiple outputs, one for each command. These outputs serve as the evidences of your answer to the following question.  Hint 1: Let n be the maximal number of blocks that can be placed directly on the table. There should be 2 command lines and outputs for n=3, where   * the 1st command line and output show k steps are not enough and * the 2nd command line and output show k+1 steps are enough. Similarly, there should be another 2 command lines and outputs for n=4.   Hint 2: We do not give any limitation on the number of grippers.      Out of which the minimum number of steps is 7.  i.e. move(1,5,0) move(1,table,1) move(2,3,1) move(2,1,2) move(3,2,3) move(5,4,3) move(6,5,4)      Out of which the minimum number of steps is 5.  i.e. move(1,table,0) move(2,1,1) move(3,2,2) move(5,4,2) move(6,5,3) |

|  |  |
| --- | --- |
|  |  |
| Answer to Questions | Fill in the following table that lists the minimum number of steps to solve the modified block world problem for different values of n, where n is the maximal number of blocks that can be placed directly on the table. |

**Problem 2**

|  |  |
| --- | --- |
| n | Number of steps |
| 3 | 7 |
| 4 | 5 |

|  |  |
| --- | --- |
| Input Program | Hint 1: You don’t need to represent any scenario since you want to find out all possible valid states with 6 blocks. Think about the value of m.  Hint 2: You don’t need to consider the limitation of the maximum number of blocks on the table. That’s only required in Problem 1.  %%%%%%%%%%%%%%%%%%% % File: q2\_blocks.txt: 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.  % two same blocks cannot be top of one another below(X,Y,T) :- on(Y,X,T).  below(X,Z,T) :- below(X,Y,T), below(Y,Z,T).  :- on(X,X,T). |

|  |  |
| --- | --- |
|  | :- on(X,Y,T), below(X,Y,T).  block(1..6). #show move/3. |
| Command Line | clingo q2\_blocks.txt -c grippers=100 -c m=0 0 |
| Output of clingo | … |
| Answer to Questions | How many valid states are there when there are 6 blocks? (Note that the limitation of blocks introduced in question 1 is not considered here.)  Answer : **4051** valid stated |

# Problem 3

Reading: A plan may allow multiple actions happening at the same time, e.g., when we have multiple robots working together to increase efficiency. However, if there is a little bit delay on one action, then we may get unexpected results. For example, when 2 robots are moving 2 adjacent blocks to the left at the same time, if there is a delay for the robot on the left-hand side, then these 2 robots may hit with each other. To make sure that our plan will get the expected result, we introduce the restriction "serializable" on the actions happening at the same time. This restriction simply says that, even if some actions in the same time stamp happen in serial with arbitrary ordering, the final result would be the same.

|  |  |
| --- | --- |
| Input | Hint: the number of grippers is unlimited, meaning that you can have as many |
| Program | movements as you want as far as the movements are serializable. |
|  | %%%%%%%%%%%%%%%%%%% % File: q3\_blocks.txt: 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 inde-pendent 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.  % serializability constraint  :- move(BB,LL,T), on(X,LL,T), move(X,B,T), block(X), T=1..m-1.  #show move/3.  %%%%%%%%%%%%%%%%%%%%%%%%% File: blocks-scenario.txt  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  block(a;b;c;d;e;f;g;h;i;j;k;l;mm;n;o).  :- not on(mm,table,0;l,mm,0;a,l,0;b,a,0;c,b,0;o,table,0;n,o,0;d,n,0;e,d,0;j,e,0;k,j,0;f,t able,0;g,f,0;h,g,0;i,h,0).  :- not on(e,j,m; a,e,m; n,a,m; i,d,m; h,i,m; mm,h,m; o,mm,m; k,g,m; c,k,m; b,c,m; l,b,m). |
| Command Line | Please only show the command line that outputs the minimal length plan.  clingo q3\_blocks.txt blocks-scenario.txt -c grippers=4 -c m=8 |

|  |  |
| --- | --- |
| Output of clingo |  |

# Problem 4

|  |  |
| --- | --- |
| Input |  |
| Program | %%%%%%%%%%%%%%%%%%% % File: q4\_blocks.txt: 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.  % serializability constraint  :- move(BB,LL,T), on(X,LL,T), move(X,B,T), block(X), T=1..m-1.  #minimize{1,B,L,T:move(B,L,T)}. #show move/3.  %%%%%%%%%%%%%%%%%%%%%%%%% File: blocks-instances.txt  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  block(a;b;c;d;e;f;g;h;i;j;k;l;mm;n;o).  :- not on(mm,table,0;l,mm,0;a,l,0;b,a,0;c,b,0;o,table,0;n,o,0;d,n,0;e,d,0;j,e,0;k,j,0; f,table,0;g,f,0;h,g,0;i,h,0).  :- not on(e,j,m; a,e,m; n,a,m; i,d,m; h,i,m; mm,h,m; o,mm,m; k,g,m;  c,k,m; b,c,m; l,b,m). |
| Command Line | You should write multiple command lines below.  clingo q4\_blocks.txt blocks-instances.txt -c grippers=4 -c m=8 0 |

|  |  |
| --- | --- |
|  | clingo q4\_blocks.txt blocks-instances.txt -c grippers=2 -c m=9 -t4 clingo q4\_blocks.txt blocks-instances.txt -c grippers=4 -c m=10 -t4 |
| Output of clingo | You should write multiple outputs, one for each command. These outputs serve as the evidences of your answer to the question below. |
| Answer to Questions | What is the least number of actions when maxstep m is 8, 9, and 10? |

|  |  |
| --- | --- |
| m | least number of actions |
| 8 | 18 |
| 9 | 16 |
| 10 | 15 |