Automated Warehouse Scenario - Project Report

Apoorv Kakade

Abstract

An automated warehouse is used in this project. In the hypothetical situation, robots transport goods and shelves from one place to another. The name and coordinates of each rectangular grid on the warehouse floor allow for easy identification. Some grids are prohibited from having shelves because they are classified as roads. The robots can move between shelves to get to the desired shelf if it's empty or underneath the shelves if it isn't because they are so flat. The robot can lift and lower shelves, distribute goods, move vertically and horizontally but not diagonally, and stand stationary without colliding with other robots. This project's main goal is to have robots transport as many things as they can to the appropriate pickup location in the lowest amount of time.

Problem Statement

The problem description uses a warehouse as a real-world example. There are many components to the warehouse, such as picking stations, robots, shelves, and motorways. In the scenario, robots deliver the shelf carrying the products to the picking station. The desired behavior of the solution to this problem is to automate the robots so that they transport the required commodities in the proper sequence to the appropriate picking station in the fewest number of steps. The problem statement is subject to the following assumptions and limitations. The warehouse is represented by a grid. The robots can only move in two directions: vertically and horizontally. Robots are not capable of diagonal movement. The robot is also limited to taking one step at a time through the grid. Robots can be positioned under a shelf and used to pick items up as needed because they are conceived of as flat structures that don't require additional space. There could be a lot of robots in the warehouse. These robots shouldn't collide while performing the tasks simultaneously in the warehouse. The three different types of grid cells are choosing station, shelf, and highway. The robot should avoid dropping the shelf on the road when transporting it from the picking point. Filenames inst1 to inst5 are used to define the warehouse situation.

Project background

In order to deliver goods quickly, profitably, and optimally, and to complete order transactions as quickly as possible, an automated warehouse is required. As a typical example of how Knowledge representation, Answer set programming, and Clingo are applied, the simplified form of the automated warehouse scenario is used. This project was created using Clingo and the answer set programming language. Clingo uses answer set solvers to get through the NPHard class issues. Solvers for answer sets serve as the foundation of ASP. It builds stable models using these Answer set solvers to address related issues. The search problems are reduced by the answer set solvers into computing stable models. To solve the project problem statement I used answer set programming and used Clingo to compile and run the programs written. The lectures helped me learning answer set programming a lot. I also updated the readings, assignments, and tests. Being familiar with all of the class Clingo programs, such as the k queens problem, Sudoku, and the most recent block puzzles. They contributed to the analysis of the fundamental components of the automated warehouse problem. I applied various concepts of answer set programming for building this project. Some concepts include:

- Commonsense Law of Inertia.
- Action and State Constraints.
- Necessary conditions and impacts of actions.

Problem Solving Approach

To solve the problem, the problem is divided into multiple small tasks as following.

- Executed Highway and Grid rules.
- Executed the movement of the robot in the X and Y axis.
- Put down and Pickup from the shelf functions are implemented.
- Implemented constraints for delivering to the pickup station.
- Applied Common Sense Law of Inertia.
- Edge cases of the system are handled.

Created a time-sensitive plan to meet each requirement by defining the system requirements for the outstanding tasks. Created the system based on the specifications. Utilized answer set programming in Clingo to put the intended functionality into practice. Verified that each condition has been met. Looked for opportunities to improve the system. Here are some of the tasks in more detail.

Understanding Input

First thing was to study the input of the warehouse given to us. It has been given to us as initialization rules. An example of some rules has been given in Figure 1.

```
init(object(node,15),value(at,pair(3,4))).
init(object(node,16),value(at,pair(4,4))).

init(object(highway,4),value(at,pair(4,1))).
```

Figure 1: Input

Inference

It is needed that we find the values of different variables like number of nodes, number of products, number of shelves, etc.

```
% Number of Nodes
num_nodes(ND):- ND=#count{I:init(object(node,I),value(at,pair(X,Y)))}.

% Number of Shelves
num_shelves(ND):- ND=#count{I:init(object(shelf,I),value(at,pair(X,Y)))}.

% Number of Products
num_products(ND):- ND=#count{I:init(object(product,I),value(on,pair(X,Y)))}.
```

Figure 2: Input

Functions of the Robot

The Robot can perform various actions like move, put down, pickup and deliver. The movement of the robot happens in the vertical, horizontal, left and right directions. The following figure describes the functions in clingo.

```
{robotMove(R,move(DX,DY),T):move(DX,DY)}1:- R=1..NR, numRobots(NR), T=0..TN,TN=n-1.
{pickUpShelf(R,SI,T):shelf(SI)}1:- R=1..NR, numRobots(NR), T=0..TN,TN=n-1.
{putDownShelf(R,SI,T):shelf(SI)}1:- R=1..NR, numRobots(NR), T=0..TN,TN=n-1.
{deliver(R,OI,with(SI,PR,DQ),T):orderAt(OI,object(node,ND),contains(PR,OQ),T), productO
```

Figure 3: Input

Constraints

The actions defined in the previous section gives complete freedom to the robot in terms of the movement but since it operates in a controlled environment, we need to restrict some of its movements so that it does not do anything that is harmful to the environment or is inefficient to the problem statement. The following figure describes some of the movement constraints in the program. It means that the robot cannot go outside of the grid which means outside of the warehouse.

```
:- robotAt(RI,object(node,ND),T), robotMove(R,move(DX,DY),T), nodeAt(ND,pair(X,Y)), X+DX<1.
:- robotAt(RI,object(node,ND),T), robotMove(R,move(DX,DY),T), nodeAt(ND,pair(X,Y)), Y+DY<1.
:- robotAt(RI,object(node,ND),T), robotMove(R,move(DX,DY),T), nodeAt(ND,pair(X,Y)), X+DX>NC, numColumns(NC).
:- robotAt(RI,object(node,ND),T), robotMove(R,move(DX,DY),T), nodeAt(ND,pair(X,Y)), Y+DY>NR, numRows(NR).
```

Figure 4: Input

Effects

Next we need to define effects of the robot's actions. The actions that have been defined in the actions state has effects on the state of the system. Some of the effects are related to robot movement, shelf management, and product delivery.

```
robotAt(R,object(node,NEW_ND),T+1):- robotAt(R,object(node,ND),T), nodeAt(ND,pair(X,Y)), nodeAt(NEW_ND, pair(X+DX,Y+DY))

shelfOn(S,object(robot,RI),T+1):- pickUpShelf(RI,S,T), shelfOn(S,object(node,ND),T), robotAt(RI,object(node,ND),T).

shelfOn(S,object(node,ND),T+1):- putDownShelf(RI,S,T), shelfOn(S,object(robot,RI),T), robotAt(RI,object(node,ND),T).

orderAt(OI,object(node,ND),contains(PR,OU-DQ),T+1):- deliver(R,OI,with(SI,PR,DQ),T), orderAt(OI,object(node,ND),contains
productOn(PR,object(shelf,SI),with(quantity,PQ-DQ),T+1):- deliver(R,OI,with(SI,PR,DQ),T), productOn(PR,object(shelf,SI),
```

Figure 5: Input

Common Sense Law of Inertia

The common sense law of inertia is used to restrict illegal and unintended state transitions of the system. In the following example we can see that the position of robot is tied to only its movement. Change in its position by any other action is not allowed.

```
robotAt(R,object(node,ND),T+1):- robotAt(R,object(node,ND),T), not robotMove(R,move(_,_),T), T<n.
```

Figure 6: Input

Main Results and Analysis

The project's sample instances in their entirety were all run. Additionally, each and every test case was covered and run. Clingo was utilized to carry out the project, and the warehouse scenario was developed with all the actions, instances, and restrictions precisely in accordance with the project's requirements. I was able to see all of the robotic actions and all of the actions connected to the fulfillment of the order(s) because of the stable model solutions from clingo execution. In terms of time steps, these acts are tracked as atomic operations. Every change in time has an impact on the atom. As a result, every action carried out by the program is regarded as an atom linked to a time step. The minimum steps needed for each of the five simple examples is shown in Table 1. These are the results for the simple instances solved.

Conclusion

A condensed version of the Automated Warehouse scenario was given as the assignment for this course project. This required coming up with an efficient way to automatically pick

Simple Instance	Timesteps	n
1	9	10
2	10	11
3	6	11
4	4	11
5	6	11

Table 1: Simple Instance vs number of time steps required by the robot

```
clingo version 5.3.0
Reading from inst1.asp ...
Solving...
Answer: 1
occurs(object(robot,1),move(-1,0),0) occurs(object(robot,1),move(-1,0),1) occurs
(object(robot,2),move(0,-1),1) occurs(object(robot,2),move(1,0),2) occurs(object
(robot,1),move(-1,0),3) occurs(object(robot,2),move(0,1),5) occurs(object(robot,
1),move(0,-1),6) occurs(object(robot,2),move(0,-1),7) occurs(object(robot,1),mov
e(0,1),8) occurs(object(robot,2),pickup,0) occurs(object(robot,1),pickup,2) occu
rs(object(robot,2),pickup,6) occurs(object(robot,1),pickup,7) occurs(object(robo
t,2),putdown,4) occurs(object(robot,1),putdown,5) occurs(object(robot,2),deliver
(2,2,1),3) occurs(object(robot,1),deliver(1,1,1),4) occurs(object(robot,2),deliv
er(3,4,1),8) occurs(object(robot,1),deliver(1,3,4),9) timeTaken(9) numActions(19
)
Optimization: 64
OPTIMUM FOUND

Models       : 1
  Optimum    : yes
Optimization : 64
Calls        : 1
Time         : 0.366s (Solving: 0.27s 1st Model: 0.04s Unsat: 0.22s)
CPU Time     : 0.353s
```

Figure 7: Simple Instance 1

```
clingo version 5.3.0
Reading from inst2.asp ...
Solving...
Answer: 1
occurs(object(robot,1),move(0,-1),0) occurs(object(robot,2),move(0,1),0) occurs(object
(robot,1),move(-1,0),1) occurs(object(robot,2),move(-1,0),2) occurs(object(robot,1),mo
ve(0,-1),4) occurs(object(robot,1),move(0,1),5) occurs(object(robot,2),move(1,0),5) oc
curs(object(robot,1),move(0,1),6) occurs(object(robot,2),move(0,-1),6) occurs(object(r
obot,1),move(-1,0),7) occurs(object(robot,1),move(-1,0),8) occurs(object(robot,2),move
(0,-1),8) occurs(object(robot,2),move(1,0),9) occurs(object(robot,2),pickup,1) occurs(
object(robot,1),pickup,3) occurs(object(robot,2),pickup,7) occurs(object(robot,2),putd
own,4) occurs(object(robot,2),deliver(1,1,1),3) occurs(object(robot,1),deliver(1,3,1),
10) occurs(object(robot,2),deliver(2,2,1),10) occurs(object(robot,1),deliver(1,3,1),9)
 timeTaken(10) numActions(21)
Optimization: 76
```

Figure 8: Simple Instance 2

```
Optimization: 32
Answer: 21
occurs(object(robot,1),move(0,-1),0) occurs(object(robot,1),move(-1,0),1) occurs(objec
t(robot,1),move(0,-1),3) occurs(object(robot,2),move(0,-1),3) occurs(object(robot,1),m
ove(0,1),5) occurs(object(robot,2),move(1,0),5) occurs(object(robot,2),pickup,0) occur
s(object(robot,1),pickup,2) occurs(object(robot,1),deliver(2,4,1),4) occurs(object(rob
ot,2),deliver(1,2,1),6) timeTaken(6) numActions(10)
Optimization: 31
OPTIMUM FOUND

Models       : 21
  Optimum    : yes
Optimization : 31
Calls        : 1
Time         : 0.352s (Solving: 0.29s 1st Model: 0.00s Unsat: 0.20s)
CPU Time     : 0.340s
```

Figure 9: Simple Instance 3

up, set down, and transport the goods (shelves) at pickup stations in the warehouses. I solved this issue using all of the information and skills I had learned in Dr. Joohyung Lee's CSE 579: Knowledge Representation and Reasoning course. Clingo concepts were used to create a response utilizing the answer set programming that Clingo has developed. Testing, integration, and standardization contributed to enhance

```
occurs(object(robot,1),move(-1,0),0) occurs(object(robot,1),move(-1,0),1) occurs(objec
t(robot,2),move(1,0),1) occurs(object(robot,2),move(0,-1),2) occurs(object(robot,1),mo
ve(-1,0),3) occurs(object(robot,2),pickup,0) occurs(object(robot,1),pickup,2) occurs(o
bject(robot,2),deliver(2,2,1),3) occurs(object(robot,1),deliver(1,1,1),4) occurs(objec
t(robot,2),deliver(3,2,2),4) timeTaken(4) numActions(10)
Optimization: 20
OPTIMUM FOUND

Models       : 13
  Optimum    : yes
Optimization : 20
Calls        : 1
Time         : 0.397s (Solving: 0.28s 1st Model: 0.01s Unsat: 0.11s)
CPU Time     : 0.384s
```

Figure 10: Simple Instance 4

```
occurs(object(robot,1),move(-1,0),0) occurs(object(robot,1),move(-1,0),1) occurs(objec
t(robot,1),move(-1,0),3) occurs(object(robot,2),move(-1,0),3) occurs(object(robot,1),m
ove(1,0),5) occurs(object(robot,2),move(0,1),5) occurs(object(robot,1),pickup,2) occur
s(object(robot,2),pickup,4) occurs(object(robot,1),deliver(1,1,1),4) occurs(object(rob
ot,2),deliver(1,3,4),6) timeTaken(6) numActions(10)
Optimization: 31
OPTIMUM FOUND

Models       : 19
  Optimum    : yes
Optimization : 31
Calls        : 1
Time         : 0.503s (Solving: 0.39s 1st Model: 0.01s Unsat: 0.13s)
CPU Time     : 0.491s
```

Figure 11: Simple Instance 5

the overall learning of the course while information gathered from this course assisted in constructing the solution using conventional development methods. Definitions and restrictions for the project's many items, including nodes, picking stations, robots, shelves, goods, and roads, serve as the solution to the issue. Move, pick up, set down, and deliver were all robot activities. The scenario's objective result was to complete all of the orders. Clingo responded to the warehouse scenarios and came up with quick fixes.

Future Work

Even while this challenge takes into account a wide range of guidelines, limitations, and players in a warehouse scenario, it may still be seen as rather straightforward in comparison to a much more sophisticated and actual warehouse. As a result, I think the following can be tested or included as an expansion to this effort.

- Employees who work in warehouses might also be taken into account.
- We can consider a network of warehouses and model the interconnection of their requirements.
- Different kinds of robots with varied capabilities can be considered.
- Extra machinery can be added to the warehouse to increase the efficiency with which the problem can be solved.
- Other warehouse variables like time bound operations can be incorporated.
- Options for supply restocking can be taken into account as soon as some item goes out of stock. In conclusion, even if the simple automated warehouse takes a number of limitations into account, other possibilities might be investigated to expand this solution and make it more realistic.

References

1. M. Gebser, P. Obermeier, T. Otto, T. Schaub, O. Sabuncu, V. Nguyen, and T. Son. Experimenting with robotic intralogistics domains. Theory and Practice of Logic Programming, http://arxiv.org/abs/1804.10247.
2. Knowledge Representation and Reasoning course material by Prof. Ali Altunkaya and Prof. Joohung Lee.
3. Martin Gebser and Philipp Obermeier. Automated Warehouse Scenario. http://www.mat. unical.it/ dodaro/aspchallenge2019/automated-w arehousescenario.package.zip