CSE 579: Knowledge Representation and Reasoning
Course Project: Milestone-4

AUTOMATED WAREHOUSE SCENARIO[1] *

Sai Likhith Yadav Seera
[1]Computer Science
School of Computing and Augmented Intelligence
Arizona State University
Tempe, United States
sseera@asu.edu
ASU ID: 1224791490

Abstract

Warehousing is an important component of logistics management since it controls inventory transportation, storage, and retrieval. Organizations are working to come up with compelling and effective management to reduce manpower. This project describes the approach taken to address the specific challenge of order fulfillment in a distribution center.

Problem Description

The project focuses on automating the warehouse tasks linked to fulfilling orders. It utilizes robots to retrieve items from storage shelves and transport them to meet customer orders. This system aims to enhance the warehouse process by minimizing human involvement, improving efficiency in managing inventory and order handling. The warehouse is divided into cells, some allocated as picking stations for order placement, and the others as highways with a necessary condition that no shelves are placed on them. The solution includes developing individual plans for robots to transport products from storage to the specific picking station where the order was made.

Constraints:

- No two entities (robots or shelves) can occupy the same place at the same time, nor can one entity exist in multiple places simultaneously.
- Similar items can be stored on different shelves throughout the warehouse.
- Robots are capable of lifting and placing shelves, but they can only transport one shelf at a time.
- While robots can move beneath shelves, they can't travel underneath another shelf if they are already carrying one.
- Robots are constrained to moving in vertical or horizontal directions between adjacent cells; diagonal movement is not permitted.
- Each robot can only execute a single action at a time, whether that involves picking up, putting down, or transporting a shelf.

Project Background

The problem statement was a part of the ASP Challenge 2019, I needed to implement the problem statement in an ASP language. I opted for clingo, an ASP system which is better known for solving combinatorial problems and also using prior knowledge of my coursework, I saw clingo as a suitable choice.

Using clingo, which is known for solving ASP problems, along with my experience from previous ASP programming assignments like the block world and monkey and banana problem, I worked on understanding how to approach the given scenario. I had to formulate all the constraints into ASP rules within clingo by referring Module 4: Practice of Answer Set Programming and the clingo documentation to understand ASP programming concepts for better output of writing the logic of the code.

Approach for Solution

I conceptualized the workings of the issue by studying the flow of the system in the given initialization files to get an idea of the constraints over the problem and various actions performed by the system.

My initial approach was to solve the problem by executing robot movements within the framework towards a specific goal state without any restrictions included. Gradually, I introduced constraints and actions related to all the objects of the system. For instance, I began coding the robots' 'pickup' and 'putdown' actions with shelves, aiming for an end state where a shelf rested on top of a robot. This methodology helped in debugging the code and identifying optimal solutions for each scenario. Initially, I faced a lot of issues while coding the problem with the constraints being included which required a lot of debugging of code. Moreover, I explored different coding approaches during the initial stages of the project, thinking some edge cases and test

*Primarily Mike Hamilton of the Live Oak Press, LLC, with help from the AAAI Publications Committee cases to check the rightness of the code. This required a lot of knowledge from ASP programming documentation.

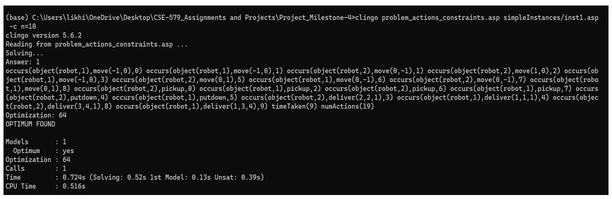
Result and Analysis

I have worked through all the instance files provided in the project for testing the developed code. My primary focus was on minimizing both the time taken and the total number of actions required for the robots to successfully deliver the requested order to the specific picking station.

Results:

The output of the program when tested with all five instances:

1. **clingo problem_actions_constraints.asp simpleInstances/inst1.asp -c n=10**



```
(base) C:\Users\likhi\OneDrive\Desktop\CSE-579_Assignments and Projects\Project_Milestone-4>clingo problem_actions_constraints.asp simpleInstances/inst1.asp
 -c n=10
clingo version 5.6.2
Reading from problem_actions_constraints.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(ob
ject(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(robo
t,1),move(0,1),8) occurs(object(robot,2),pickup,0) occurs(object(robot,1),pickup,2) occurs(object(robot,2),pickup,6) occurs(object(robot,1),pickup,7) occurs
(object(robot,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(objec
t(robot,2),deliver(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         : 0.724s (Solving: 0.52s 1st Model: 0.13s Unsat: 0.39s)
CPU Time     : 0.516s
```

Figure 1: Results when the program is tested against inst1.asp

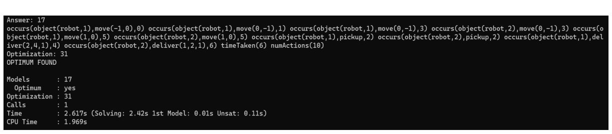
2. **clingo problem_actions_constraints.asp simpleInstances/inst2.asp -c n=12**

```
(base) C:\Users\likhi\OneDrive\Desktop\CSE-579_Assignments and Projects\Project_Milestone-4>clingo problem_actions_constraints.asp simpleInstances/inst2.asp
 -c n=12
clingo version 5.6.2
Reading from problem_actions_constraints.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(ob
ject(robot,1),move(-1,0),3) occurs(object(robot,1),move(1,0),5) occurs(object(robot,2),move(0,1),5) occurs(object(robot,1),move(0,-1),6) occurs(object(robot
,1),move(0,-1),7) occurs(object(robot,2),move(-1,0),7) occurs(object(robot,1),move(-1,0),8) occurs(object(robot,2),move(-1,0),8) occurs(object(robot,2),move
(0,1),9) occurs(object(robot,2),pickup,0) occurs(object(robot,1),pickup,2) occurs(object(robot,2),pickup,6) occurs(object(robot,2),putdown,4) occurs(object(
robot,2),deliver(2,2,1),3) occurs(object(robot,1),deliver(1,1,1),4) occurs(object(robot,2),deliver(1,3,2),10) timeTaken(10) numActions(20)
Optimization: 75
Answer: 2
occurs(object(robot,1),move(-1,0),1) occurs(object(robot,2),move(0,-1),1) occurs(object(robot,1),move(-1,0),2) occurs(object(robot,2),move(1,0),2) occurs(ob
ject(robot,1),move(-1,0),4) occurs(object(robot,2),move(0,1),5) occurs(object(robot,1),move(1,0),6) occurs(object(robot,2),move(-1,0),7) occurs(object(robot
,1),move(0,1),8) occurs(object(robot,2),move(0,1),8) occurs(object(robot,2),move(-1,0),9) occurs(object(robot,2),pickup,0) occurs(object(robot,1),pickup,3)
occurs(object(robot,2),pickup,6) occurs(object(robot,2),putdown,4) occurs(object(robot,2),deliver(2,2,1),3) occurs(object(robot,1),deliver(1,1,1),5) occurs(
object(robot,2),deliver(1,3,2),10) timeTaken(10) numActions(18)
Optimization: 73
Answer: 3
occurs(object(robot,1),move(-1,0),0) occurs(object(robot,1),move(-1,0),1) occurs(object(robot,2),move(1,0),1) occurs(object(robot,2),move(0,-1),2) occurs(ob
ject(robot,1),move(-1,0),3) occurs(object(robot,1),move(1,0),5) occurs(object(robot,2),move(0,1),5) occurs(object(robot,2),move(-1,0),7) occurs(object(robot
,2),move(-1,0),8) occurs(object(robot,2),move(0,1),9) occurs(object(robot,2),pickup,0) occurs(object(robot,1),pickup,2) occurs(object(robot,2),pickup,6) occ
urs(object(robot,2),putdown,4) occurs(object(robot,2),deliver(2,2,1),3) occurs(object(robot,1),deliver(1,1,1),4) occurs(object(robot,2),deliver(1,3,2),10) t
imeTaken(10) numActions(17)
Optimization: 72
OPTIMUM FOUND

Models       : 3
  Optimum    : yes
Optimization : 72
Calls        : 1
Time         : 1.580s (Solving: 1.31s 1st Model: 0.09s Unsat: 0.82s)
CPU Time     : 1.172s
```

Figure 2: Results when the program is tested against inst2.asp

3. **clingo problem_actions_constraints.asp simpleInstances/inst3.asp -c n=14**

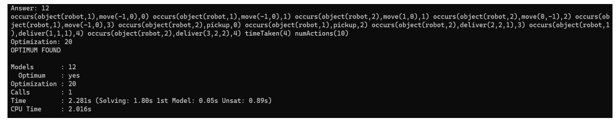


```
Answer: 17
occurs(object(robot,1),move(-1,0),0) occurs(object(robot,1),move(0,-1),1) occurs(object(robot,1),move(0,-1),3) occurs(object(robot,2),move(0,-1),3) occurs(o
bject(robot,1),move(1,0),5) occurs(object(robot,2),move(1,0),5) occurs(object(robot,1),pickup,2) occurs(object(robot,2),pickup,2) occurs(object(robot,1),del
iver(2,4,1),4) occurs(object(robot,2),deliver(1,2,1),6) timeTaken(6) numActions(10)
Optimization: 31
OPTIMUM FOUND

Models       : 17
  Optimum    : yes
Optimization : 31
Calls        : 1
Time         : 2.617s (Solving: 2.42s 1st Model: 0.01s Unsat: 0.11s)
CPU Time     : 1.969s
```

Figure 3: Results when the program is tested against inst3.asp

4. **clingo problem_actions_constraints.asp simpleInstances/inst4.asp -c n=16**

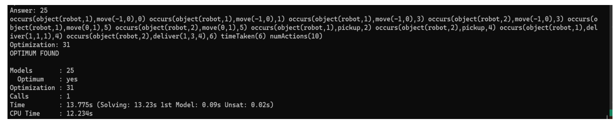


```
Answer: 12
occurs(object(robot,1),move(-1,0),0) occurs(object(robot,1),move(-1,0),1) occurs(object(robot,2),move(1,0),1) occurs(object(robot,2),move(0,-1),2) occurs(ob
ject(robot,1),move(-1,0),3) occurs(object(robot,2),pickup,0) occurs(object(robot,1),pickup,2) occurs(object(robot,2),deliver(2,2,1),3) occurs(object(robot,1
),deliver(1,1,1),4) occurs(object(robot,2),deliver(3,2,2),4) timeTaken(4) numActions(10)
Optimization: 20
OPTIMUM FOUND

Models       : 12
  Optimum    : yes
Optimization : 20
Calls        : 1
Time         : 2.281s (Solving: 1.80s 1st Model: 0.05s Unsat: 0.89s)
CPU Time     : 2.016s
```

Figure 4: Results when the program is tested against inst4.asp

5. **clingo problem_actions_constraints.asp simpleInstances/inst5.asp -c n=18**



```
Answer: 25
occurs(object(robot,1),move(-1,0),0) occurs(object(robot,1),move(-1,0),1) occurs(object(robot,1),move(-1,0),3) occurs(object(robot,2),move(-1,0),3) occurs(o
bject(robot,1),move(0,1),5) occurs(object(robot,2),move(0,1),5) occurs(object(robot,1),pickup,2) occurs(object(robot,2),pickup,4) occurs(object(robot,1),del
iver(1,1,1),4) occurs(object(robot,2),deliver(1,3,4),6) timeTaken(6) numActions(10)
Optimization: 31
OPTIMUM FOUND

Models       : 25
  Optimum    : yes
Optimization : 31
Calls        : 1
Time         : 13.775s (Solving: 13.23s 1st Model: 0.09s Unsat: 0.02s)
CPU Time     : 12.234s
```

Figure 5: Results when the program is tested against inst5.asp

Analysis:

- Certain rule sections demanded multiple updates due to their complex nature at times during the project. For instance, when I increased the requirements for robots and orders, I encountered random movements that needed debugging, taking up a significant portion of my time.
- However, All instances ran successfully within optimal timeframes, ensuring that the robots consistently reached their goal states. The optimized plan I have implemented achieved efficiency by minimizing timestamps and the cumulative actions performed by the robots.
- Upon analysis, I found out that increasing the number of robots led to reduced delivery times because of simultaneous delivery. With multiple robots operating on a grid, I could efficiently navigate them through empty cells, allowing smoother order deliveries without any accidents.
- Also, increase in the number of orders resulted in an increase in the time taken by the robots for delivering. This was mainly due to the constraint allowing each robot to carry only a single shelf at any given time.
- Even after increasing the number of robots or shelves, no collisions were detected. Ultimately, every instance was satisfied and the robot consistently reached the goal state within the fewest steps possible.

Conclusion

The project successfully addresses the challenge of order fulfillment in a distribution center through automation. By utilizing robots to manage inventory retrieval and order assembly, the system significantly reduces human involvement, enhancing efficiency in warehouse operations.

The implementation of ASP programming, particularly using clingo, allowed for the formulation of complex constraints and actions within the warehouse environment which required a lot of propositional logic. The practical application of answer set language in resolving an automated robot scenario provided invaluable hands-on experience, fostering a deeper understanding of ASP programming. Despite initial challenges in debugging and refining the code, the system efficiently navigates robots through the warehouse, ensuring optimal order delivery within minimal timeframes. Also, the robots traversed through the workspace without collisions and successfully completing all orders in minimum time, ensuring an optimal solution and concluding the project's development cycle.

Opportuities for Future Work

The project's applications span across various industries that manage product warehouses and orders. The automated warehouse system, driven by robots, offers speed, efficiency, and reliability far beyond manual labor. Tasks like storage, retrieval, relocation, and delivery of goods become remarkably flexible with robots, capable of executing complex operations and collaborating to expedite orders in specific scenarios. Extending adaptability to dynamic environments ensures the system's responsiveness to unforeseen obstacles. Enhanced multi-agent coordination streamlines robot movements, optimizing order fulfillment processes.

Given these features and the transformative potential of robotics, promoting the widespread use of robots in warehouses and various industries emerges as a compelling avenue for innovation and efficiency enhancement.

References

- ASU CSE-579 Lecture Videos and Lecture Slides by Professor Joohyung Lee:
  - Module 3: Theory of Answer Set Programming
  - Module 4: Practice of Answer Set Programming
  - Module 5: Reasoning about Actions
- Technische Universitat Wien (Austria) and the University of Genoa (Italy), Answer Set Programming Challenge (2019),
  https://sites.google.com/view/aspcomp2019.
- Gebser, M.; Kaminski, R.; Kaufmann, B.; Ostrowski, M.;Schaub, T.; and Thiele, S. A User's Guide to gringo, clasp, clingo, and iclingo, The University of Texas at Austin(2010),
  https://wp.doc.ic.ac.uk/arusso/wp-content/uploads/sites/47/2015/01/clingo_guide.pdf.