Automated Warehouse Scenario

Animesh Chaudhary
School of Computing and Augmented Intelligence
Arizona State University
achaud81@asu.edu (ID: 1229421130)

Problem Statement

The central focus of this project is to increase the efficiency and speed of product transportation within a warehouse to designated pick-up points. This improvement is critical for the timely execution of customer orders. In our automated warehouse setup, robots are utilized to speed up the movement of items. Their primary tasks include fetching products from assigned shelves and navigating without collisions to the pick-up stations.

To effectively carry out their responsibilities, the robots are designed to move both horizontally and vertically across a rectangular grid that mirrors the warehouse layout. This grid-based navigation is crucial for their functionality. The robots are uniquely designed with a low-profile form, allowing them to access items on the lower sections of shelves. A significant aspect of their operation involves the strategic relocation of shelves they are currently transporting. This is necessary to make room for accessing other shelf undersides, ensuring that the robots can perform their tasks smoothly without any physical obstructions.

In our warehouse model, time is divided into distinct increments, a crucial setup that allows each robot to perform a single action per increment. This method enhances the efficiency of task completion. The primary goal is to achieve all tasks in the shortest time possible while maintaining a safe environment where robots avoid collisions. Key safety rules include prohibiting two robots from occupying adjacent cells on the grid simultaneously.

Each robot is designated specific tasks such as picking up goods, navigating through the grid, remaining stationary when needed, or delivering goods, all without the risk of collision. Additionally, robots are restricted from entering cells occupied by others or transitioning through them during different operational phases. This restriction poses a challenge in the dynamic warehouse environment where a robot's location or position may constantly change. Such complexity requires precise coordination and control to ensure smooth and safe operations.

The grid cells in the warehouse act more like pathways for robot movement rather than storage locations, emphasizing the importance of exact product delivery as per the order. In scenarios involving multiple units of the same product, these need to be delivered to different locations, requiring meticulous coordination from the robots to ensure each delivery is accurate and efficient.

This project utilizes advanced knowledge representation tools to streamline decision-making in the warehouse setting. Under this system, robots are limited to performing only one action at each moment, focusing on moving products to specified pick-up points while adhering to operational rules. These critical rules include:

- Ensuring the aggregate of units on the shelves matches the total units specified in the product orders.
- Restricting each robot to a single operation per time increment, with time segmented into discrete steps.
- Prohibiting a robot carrying a shelf from positioning itself under another shelf, unless the shelf it's carrying is first relocated.
- Advising against placing shelves on the main routes used by robots within the grid layout.
- Mandating that robots maneuver through the grid without any instances of collision with one another.
- Requiring robots to transport only those goods intended for the picking stations.

These measures are set to ensure an efficient, precise, and error-free operation within the automated warehouse system.

Project Background

The Automated Warehouse Project exemplifies the ongoing shift towards automation in e-commerce, as demonstrated by major retail players like Amazon and Walmart, who are increasingly relying on automated systems. While not all warehouses are fully automated, this project illustrates how automation can provide a more cost-effective and faster alternative to traditional manual operations. In the sphere of artificial intelligence, a crucial component is knowledge representation and reasoning (KRR). This field concentrates on formulating knowledge using formal languages and enhancing its automatic processing, enabling computers to derive intelligent conclusions efficiently.

For the implementation of this Automated Warehouse project, our primary technological tool will be Clingo, an advanced Answer Set Programming (ASP) tool. We aim to represent the warehouse's functions and the operational needs of its robotic systems through a combination of choice rules and constraints. These have been a focal point of our learning in recent coursework, specifically during weeks three to five. Our approach to automation will leverage strategies centered on "Reasoning about actions and planning." The initiation of this project will employ these specific techniques and methods, providing a structured and theory-driven framework for the automation of warehouse activities.

Approach to Solving the Problem

To tackle the complexities of this project, I initiated the process by methodically mapping out the system operations as outlined in the initialization files. This preliminary step involved more than just creating diagrams; it was about dissecting and understanding the array of actions and constraints integral to the functioning of the automated warehouse. Alongside this, a detailed review of the initialization files was undertaken to not only grasp the inputs but also to devise strategies for simplifying the implementation process.

Understanding the intricate details of the problem led me to adopt an iterative approach. This strategy allowed for progressive enhancement of both my understanding and the solutions themselves, by addressing issues incrementally. Each iteration helped in identifying and resolving gaps in my understanding, thereby solidifying my grasp of the system's intricacies.

My initial strategy allowed for free movement of robots within a defined framework, with the aim of reaching a designated node. As the project evolved, I incorporated specific constraints and actions that applied to all elements of the system. Early tasks included programming the robots to perform 'pickup' and 'putdown' actions, essential for manipulating the shelves. This approach was pivotal in debugging the system and refining solutions to various challenges encountered along the way.

To effectively manage the complexity, I broke down the problem into smaller, manageable sub-problems such as decoding input scenarios, and coding movements and actions—move, put down, pick up, and transport. The objective was to complete all orders in the minimum number of time steps. I began with a simple 2x2 grid scenario for initial validation, which was crucial for ensuring accurate interpretation of scenarios and establishing appropriate constraints, sidestepping the complexities a larger 4x4 grid might introduce at an early stage.

While defining the movement and action rules, I identified overlaps that caused repeated values at the initial time step. To rectify this, I used a grounding method **muve(XX,YY)**, as recommended by Dr. Lee, which was crucial for eliminating redundant actions. This grounding method was applied specifically for rules in the brain of our system, as it was not necessary for rules affecting the body.
Throughout this project, the action domain principles guided the design and execution phases, particularly from the fifth week of our course. Key strategies included:

- **Action Constraints:** Ensuring that a robot carrying a shelf cannot pick up another, enforced with Clingo rules:

```
{ p_upshelf(R1, S, T) : shelfOn(S, robot(R2), T) } = 0.
```

- **Location Constraints:** Preventing robots from placing shelves on pathways, implemented with:

```
:- p_dwnshelf(RI, S, T), highway(ND), robotAt(RI, ND, T).
```

The overarching aim was to expedite order fulfillment efficiently using strategic actions and constraints, focusing on:

- **Calculating Total Actions Required:**

```
countAction(N) :- N = #sum { 1,O,A,T : occranc(O, A, T) }.
```

- **Minimizing Steps Needed:**

```
#minimize { 1,O,A,T : occranc(O, A, T) }.
```

By adopting these methodologies, the project steadily moved towards achieving quick and efficient order processing in the automated warehouse environment.

Main Results and Analysis

The operational efficiency of the system was thoroughly evaluated using Clingo.exe in conjunction with various instance files (inst.asp), each labeled with a different parameter 'n'. These tests generated comprehensive reports, detailing every movement, item pickup, and delivery by each robot. These reports are invaluable for assessing the functionality of the system, identifying improvement areas, and documenting all relevant data for future analysis and optimization.

Here are the snapshots of the results obtained from different instance files:

• *Figure 1*. Result obtained for inst1.asp:

```
Answer: 1
occranc(object(robot,1),muve(-1,0),0) occranc(object(robot,1),muve(-1,
0),1) occranc(object(robot,2),muve(0,-1),1) occranc(object(robot,2),mu
ve(1,0),2) occranc(object(robot,1),muve(-1,0),3) occranc(object(robot,
2),muve(0,1),5) occranc(object(robot,1),muve(0,-1),6) occranc(object(r
obot,2),muve(0,-1),7) occranc(object(robot,1),muve(0,1),8) occranc(obj
ect(robot,2),pickup,0) occranc(object(robot,1),pickup,2) occranc(objec
t(robot,2),pickup,6) occranc(object(robot,1),pickup,7) occranc(object(
robot,2),putdown,4) occranc(object(robot,1),putdown,5) occranc(object(
robot,2),deli(2,2,1),3) occranc(object(robot,1),deli(1,1,1),4) occranc
(object(robot,2),deli(3,4,1),8) occranc(object(robot,1),deli(1,3,4),9)
 timeTaken(9) numActions(19)
Optimization: 64
OPTIMUM FOUND
```

• *Figure 2*. Result obtained for inst2.asp

```
Answer: 3
occranc(object(robot,1),muve(-1,0),0) occranc(object
(robot,1),muve(-1,0),1) occranc(object(robot,2),muve
(1,0),1) occranc(object(robot,2),muve(0,-1),2) occra
nc(object(robot,1),muve(-1,0),3) occranc(object(robo
t,1),muve(1,0),5) occranc(object(robot,2),muve(0,1),
5) occranc(object(robot,2),muve(-1,0),7) occranc(obj
ect(robot,2),muve(-1,0),8) occranc(object(robot,2),m
uve(0,1),9) occranc(object(robot,2),pickup,0) occran
c(object(robot,1),pickup,2) occranc(object(robot,2),
pickup,6) occranc(object(robot,2),putdown,4) occranc
(object(robot,2),deli(2,2,1),3) occranc(object(robot
,1),deli(1,1,1),4) occranc(object(robot,2),deli(1,3,
2),10) timeTaken(10) numActions(17)
Optimization: 72
OPTIMUM FOUND

Models       : 3
  Optimum    : yes
Optimization : 72
```

• *Figure 3*. Result obtained for inst3.asp

```
Answer: 21
occranc(object(robot,1),muve(0,-1),0) occranc(object
(robot,1),muve(-1,0),1) occranc(object(robot,1),muve
(0,-1),3) occranc(object(robot,2),muve(0,-1),3) occr
anc(object(robot,1),muve(0,1),5) occranc(object(robo
t,2),muve(1,0),5) occranc(object(robot,2),pickup,1)
occranc(object(robot,1),pickup,2) occranc(object(rob
ot,1),deli(2,4,1),4) occranc(object(robot,2),deli(1,
2,1),6) timeTaken(6) numActions(10)
Optimization: 31
OPTIMUM FOUND

Models       : 21
  Optimum    : yes
Optimization : 31
```

• *Figure 4*. Result obtained for inst4.asp

```
Answer: 6
occranc(object(robot,1),muve(-1,0),0) occranc(object
(robot,1),muve(-1,0),1) occranc(object(robot,2),muve
(1,0),1) occranc(object(robot,2),muve(0,-1),2) occra
nc(object(robot,1),muve(-1,0),3) occranc(object(robo
t,2),pickup,0) occranc(object(robot,1),pickup,2) occ
ranc(object(robot,2),deli(2,2,1),3) occranc(object(r
obot,1),deli(1,1,1),4) occranc(object(robot,2),deli(
3,2,2),4) timeTaken(4) numActions(10)
Optimization: 20
OPTIMUM FOUND

Models       : 6
  Optimum    : yes
Optimization : 20
```

• *Figure 5*. Result obtained for inst5.asp

```
Answer: 17
occranc(object(robot,1),muve(-1,0),0) occranc(object
(robot,1),muve(-1,0),1) occranc(object(robot,1),muve
(-1,0),3) occranc(object(robot,2),muve(-1,0),3) occr
anc(object(robot,1),muve(1,0),5) occranc(object(robo
t,2),muve(0,1),5) occranc(object(robot,1),pickup,2)
occranc(object(robot,2),pickup,4) occranc(object(rob
ot,1),deli(1,1,1),4) occranc(object(robot,2),deli(1,
3,4),6) timeTaken(6) numActions(10)
Optimization: 31
OPTIMUM FOUND

Models       : 17
  Optimum    : yes
Optimization : 31
```

Upon closely reviewing the outcomes across these five scenarios, it is evident that the system delivers robust performance and effectively minimizes the number of steps required to complete the tasks. Here is a breakdown of the time steps required for each scenario:

- **inst1.asp** required **9 time steps,**
- **inst2.asp** took **10 time steps,**
- **inst3.asp** was completed in **6 time steps,**
- **inst4.asp** finished in **4 time steps,**
- **inst5.asp** also needed **6 time steps.**

These metrics are crucial for gauging the system's efficiency and provide a clear metric to guide enhancements in future versions. By analyzing the results, including detailed observations of each robot's path, the specific items they handled, and their delivery points, we gain comprehensive insights. This detailed analysis enables us to make data-driven decisions to enhance system performance further, streamlining operations and increasing the overall efficiency of the warehouse environment.

Conclusion

The journey through the Automated Warehouse project has been both enlightening and rewarding. Here are the key milestones achieved during this endeavor:

- **Thorough Problem Review:** The project commenced with a comprehensive review of the problem statement and all accompanying materials, including instructional resources provided. This initial step was crucial in gaining a clear understanding of the project objectives and requirements.
- **Identification of Critical Constraints:** I meticulously identified the critical functional terms and constraints essential for the successful implementation of the project. This process involved careful analysis and interpretation of the problem's intricacies.
- **Implementation of Constraints:** Leveraging the concepts learned from relevant lecture modules, I adeptly implemented the identified constraints. Regular reference to instructional materials helped reinforce my understanding and ensure accurate implementation.
- **Testing and Validation:** Rigorous testing was conducted using custom test cases to verify the effectiveness of the implemented constraints. This testing phase provided valuable insights into the subtleties of Answer Set Programming, further enhancing my proficiency in this domain.

The Automated Warehouse system emerged as a groundbreaking innovation, offering a flexible solution adaptable to various warehouse setups. Its implementation promises enhanced efficiency in executing physical tasks, particularly during night shifts, thereby optimizing processes and reducing operational costs.

In conclusion, this project has been an invaluable learning experience, significantly deepening my understanding of Knowledge Representation and Reasoning. It has sharpened my practical application skills and equipped me with valuable insights that will undoubtedly prove beneficial in future endeavors.

Opportunities for Future Work

While the Automated Warehouse project has yielded promising results, there are several avenues for future research and development:

- **Expansion to Other Industries:** The principles and methodologies employed in the Automated Warehouse project can be adapted to various industries beyond logistics. For example, in airport operations, implementing automated baggage systems could enhance efficiency and reliability, reducing the chances of misplaced luggage. Similarly, in cargo shipping, automated warehouses could optimize loading and unloading processes, leading to time and labor savings.
- **Integration of Advanced Technologies:** Incorporating machine learning algorithms could further optimize task assignments based on historical data and real-time analytics. This integration would enhance the system's adaptability and efficiency in handling dynamic warehouse environments.
- **User Interface Enhancement:** Developing a user-friendly interface for warehouse managers to interact with the system would improve usability and accessibility. An intuitive interface would allow for easier monitoring and management of warehouse operations, facilitating better decision-making.
- **Energy Efficiency Improvements:** Exploring methods to improve the energy efficiency of robot operations would contribute to the sustainability of the system. By reducing energy consumption, the system can minimize its environmental footprint while maintaining high levels of performance.

In addition to these future directions, scalability testing is essential for evaluating the system's performance in handling larger and more complex warehouse environments. Conducting comprehensive scalability tests will provide insights into the system's robustness and scalability potential, paving the way for its successful deployment in real-world scenarios.

References

[1] Atieh, Anas & Kaylani, Hazem & Al Abdallat, Yousef & Qaderi, Abeer & Ghoul, Luma & Jaradat, Lina & Hdairis, Iman. (2016). *Performance Improvement of Inventory Management System Processes by an Automated Warehouse Management System*. Procedia CIRP. 41. 568-572. 10.1016/j.procir.2015.12.122.

[2] Cabalar, P., Fandinno, J., & Muñiz, B. (2020). *A system for explainable answer set programming*. *arXiv preprint arXiv:2009.10242*.

[3] Xiaoqiao Wang and Yuxiang Hong, "The ASP system design of cluster supply chain management oriented to SMEs," *2010 International Conference On Computer Design and Applications*, Qinhuangdao, China, 2010, pp. V2-305-V2-310, doi: 10.1109/ICCDA.2010.5541474.

[4] Eiter, Thomas & Ianni, Giovambattista & Krennwallner, Thomas. (2009). *Answer Set Programming: A Primer. Lecture Notes in Computer Science*. 5689. 40-110. 10.1007/978-3-642-03754-2_2.

[5] Lierler, Yuliya. "Constraint answer set programming: Integrational and translational (or SMT-based) approaches." *Theory and Practice of Logic Programming* 23, no. 1 (2023): 195-225.