Automated Warehouse Scenario

Project Milestone 4

Kavya Chandrika Vempalli
kvempall@asu.edu

Abstract

Creating a system for warehouses where products can be picked up and placed precisely is super important for making sure things get delivered the best way possible, especially in big places like Amazon's warehouses. So, I made a pretend warehouse where robots get told what to do at stations and then they take those orders and give stuff to customers automatically. To do this, I used a special kind of computer programming called Answer Set Programming with CLINGO. I learned a bunch of useful things about this from a course by Dr. Joohyung Lee called CSE 579 - Knowledge Representation and Reasoning. While working on this project, I learned a lot of cool stuff about programming and other ideas that I think will help me in my future projects too.

Problem Statement

In the automatic warehouse, robots have a crucial job: they move items to specific spots called picking stations to finish orders. The warehouse looks like a big rectangle divided into smaller squares. The robots can go from one square to another next to it, either side to side or up and down. Their main job is to take shelves with the things that people ordered to the right spots for them to pick up.

The robots are smart—they can move under shelves and even lift them up. But there's a catch: a robot holding a shelf can't pass under another shelf. This means they might have to move shelves around to make space. The big goal here is to finish all the orders as quickly as possible. We measure time in steps, where each robot can only do one thing in each step, like moving, lifting, delivering, or just waiting.

It's super important that robots don't bump into each other while doing their jobs. They have strict rules—no robot can share a square with another one in one step or move into the square another robot was in right before.

Also, some squares in the warehouse are special roads called highways. Robots can't put shelves on these highways, which adds another challenge to the whole thing. So, the big challenge here is to plan how the robots move and where they put shelves so that they finish all the orders and follow all these tough rules and limits.

This task is like solving a puzzle: you have to think carefully about how each robot moves and where shelves should go to finish all the orders without breaking any rules.

Project Background

Automated Warehouses stand at the forefront of the dynamic world of e-commerce and business, wielding immense significance. These highly sophisticated facilities serve as the lifeblood for numerous platforms, ensuring the smooth processing of orders and timely delivery to a vast array of customers. Their pivotal role lies in streamlining operations, revolutionizing the speed at which products are retrieved and stored, thereby minimizing delays and optimizing the use of resources. The automation of critical tasks such as inventory management, sorting, and packaging not only reduces human errors but also significantly enhances accuracy. The hallmark of these warehouses lies in their efficient order fulfillment, a factor that directly translates to contented customers and recurrent business, eventually bolstering the bottom line of companies.

Operating behind this seamless facade are Answer Set Solvers, robust computational tools that tackle intricate and often categorized as NP-Hard problems. These solvers specialize in generating stable models, allowing us to grapple with challenges inherent in product placement, routing, and load balancing within the warehouse. They possess the unique capability to transform convoluted logistical dilemmas into elegant, efficient solutions.

Our project venture involves the implementation of an automated warehouse scenario utilizing CLINGO, an answer set programming language. The inspiration for this endeavor stems from the Block Words problem, where we draw parallels and insights. Concepts like exogeneity and the commonsense law of inertia serve as guiding principles, steering our design decisions. Leveraging our class lectures and meticulous notes, we harness these invaluable resources to craft an intelligent and efficient warehouse system.

Automated Warehouses, the engine rooms of modern commerce, operate as an intricate network ensuring the smooth flow of goods and services. These state-of-the-art facilities play a pivotal role in the fast-paced world of e-commerce and business operations. They serve as the nerve center for various platforms, allowing for seamless order processing and timely delivery to a diverse customer base. In essence, these warehouses streamline operations, enabling swift retrieval and storage of products while minimizing delays and optimizing resource utilization.

Central to the success of automated warehouses is their ability to automate crucial tasks like inventory management, sorting, and packing. By doing so, these facilities significantly reduce human errors, enhancing accuracy and efficiency in the order fulfillment process. Such efficiency translates into satisfied customers, fostering customer loyalty and ultimately contributing to the profitability of businesses.

Behind the scenes of these efficient operations lie the Answer Set Solvers, robust computational tools capable of tackling complex problems often categorized as NP-Hard. These solvers generate stable models, allowing for the resolution of challenges related to product placement, routing, and load balancing within the warehouse. They possess the unique ability to transform intricate logistical problems into elegant and effective solutions.

Our project delves into the implementation of an automated warehouse scenario through the utilization of CLINGO, an answer set programming language. Taking cues from problems like the Block Words problem, we draw inspiration and guidance, employing concepts like exogeneity and the commonsense law of inertia in our design. Leveraging our class lectures and detailed notes, we navigate the complexities to create an intelligent and efficient warehouse system.

Approach to Solve the Problem

The project consists of several stages as described:

- Initially, rules governing the Grid and Highway are established.
- Robot movement was developed along both the X and Y axes.
- Procedures for picking up and putting down shelves were created.
- Completion of delivering items to the picking station and addressing associated limitations finalized the tasks.

The images below display examples of the grid and limitations related to products. For example, 'node(C, D)' signifies a node X positioned at coordinates (X, Y). 'highway(C, D)' indicates the presence of a highway H at coordinates (C, D). Additionally, 'pickingStation(P, C, D)' represents the existence of a picking station at location (C, D).

```
%%%%%%%%%%%%%%%%%%%%%%
% Grid Constraints
%%%%%%%%%%%%%%%%%%%%%%
node(C, D):- init(object (node, X) , value (at,
pair(C, D))).

pickingStation(P, C, D):- init(object(
pickingStation, P), value(at, pair(C ,D))).

highway(C, D):- init(object(highway, H), value(at,
pair(C ,D))).
```

Figure 1. Constraints of Grid

The tasks mentioned earlier were performed on provided sample scenarios. Test cases covered various actions such as moving robots along different axes, handling shelf pickup and delivery to specific stations. Rules were formulated to prevent robot collision and implement the Commonsense Law of Inertia. This law focused on aspects like item and robot locations, carrying action status, and order status. An accompanying image illustrates the logic used. The 'at()' rule signifies an item I at location (C, D) at time T remaining there at time T+1. 'on()' indicates an object at location (S, Y) and shelf K at time T remaining at the same place at time T+1. The 'carry()' rule states that a robot carrying shelf S at time T will still carry it at time T+1. Finally, the 'order()' rule implies an order with E items of type I at time T will retain the same amount at time T+1.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Constraints for the Commonsense law of inertia
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
{at(object (X ,I), C, D, T + 1)} :- at(object (X, I), C, D, T), T = 0..m-1.

{on(X, K, S, Y, T+ 1)} :- on(X, K, S, Y, T), T = 0..m-1.

{carry(R, S, T + 1)} :- carry(R, S, T), T = 0..m-1.

{order(O, I, E, T + 1)}:- order(O, I, E, T), T = 0..m-1.
```

Figure 2. Commonsense law of inertia constraints

Main Results and Analysis

All the provided sample scenarios for the project were run successfully. Additionally, all corresponding test cases were executed comprehensively. Clingo was utilized to execute the project, implementing the warehouse scenario precisely as described in the project specifications, encompassing all actions, instances, and constraints.

The solutions derived from Clingo's stable models enabled the visualization of both robotic activities and actions necessary for fulfilling orders. These actions are observed as singular operations within defined time increments. Each time step alteration corresponds to a change in the atomic representation, reflecting the sequence of actions performed in the program.

The following table gives the minimum steps required for all the instances given.

Instance	Timesteps	Number of Actions
1	9	19
2	10	17
3	6	10
4	4	10
5	6	10

Table 1: Minimum Time-steps and number of actions required for solving sample instances

As the time steps increased, the time taken by the solver increased. The first and second instances took more time than the remaining to complete. The screenshots of instances are placed below.

Figure 3. Instance 1 Output

Figure 4. Instance 2 Output

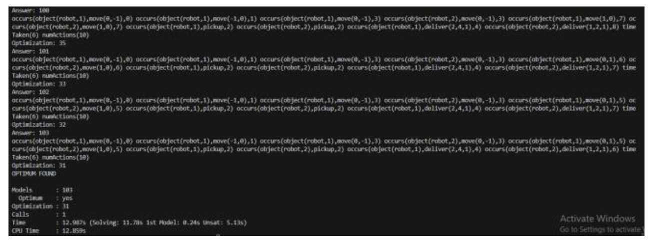

Figure 5. Instance 3 Output

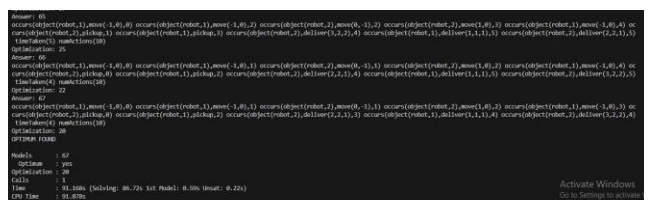

Figure 6. Instance 4 Output

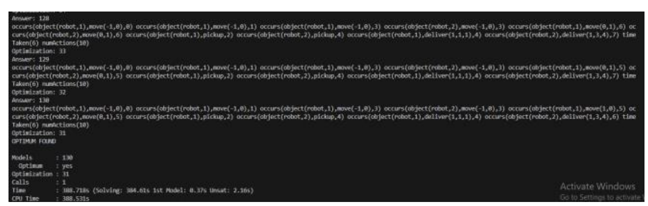

Figure 7. Instance 5 Output

The following are the issues identified while I was going through the project description.

- While going through the input format of the problem statement, I faced a lack of clarity in understanding some instances code.
- Although grasping the intricate details of ASP presented its own challenge, my main struggle revolved around troubleshooting the ASP code. This difficulty arose from the absence of adequate debugging tools and a complete comprehension of the framework.

Conclusion

For this academic project, a simplified rendition of an Automated Warehouse task was assigned. This encompasses creating a solution to automate the handling of product(shelves) pick up, placement and delivery to designated stations within warehouses, aiming for efficiency and cost effectiveness. Drawing on insights from the CSE 579- Knowledge Representation and Reasoning course instructed by Dr. Joohyung Lee, we utilized Clingo, an Answer Set Programming tool, to tackle this challenge.

The knowledge acquired from the course significantly contributed to developing the solution, applying traditional development methods, testing, integration and standardized practices, enriching the overall learning experience. The solution addressed the problem by defining objects and constraints related to nodes, pickup stations, robots, shelves, products, and highways, along with specifying robot actions: movement, picking up, putting down and delivering. The primary objective was to fulfill all orders within the scenario.

Clingo facilitated answering warehouse scenarios and generating optimal-time solutions. Looking ahead, our focus lies in enhancing the simplified automated warehouse model by incorporating additional real-world constraints and features for further development.

Future Work

The project at hand presents a simplified version of the intricate operations found within real-world automated warehouses, akin to those utilized by major companies like Amazon. However, this project is just the beginning—a stepping stone toward creating a more comprehensive and true-to-life representation of these automated warehouse environments. There exists significant potential for future development and enhancements, aiming to bring this simulation closer to the complexities and functionalities seen in genuine automated warehouses.

To achieve this evolution, several enhancements and modifications are proposed. Firstly, there's the need for a more intricate structural setup within the project itself. This involves adding greater precision and detailed constraints, aiming to replicate the intricate layout and design considerations present in real warehouses.

Expanding the scope of the project is crucial. This expansion involves incorporating the concept of warehouse locations, accounting for the distinct settings and layouts of various warehouses. Moreover, the project can be extended to encompass scenarios involving multiple warehouses, reflecting the multifaceted operations seen in large-scale warehouse networks.

One significant aspect is the involvement of warehouse employees. In reality, these facilities operate with the input and assistance of human workers. Including considerations for human intervention or engagement within the automated warehouse model would add a layer of authenticity to the simulation.

Furthermore, an intriguing aspect would be to develop an automated warehouse model that allows for manual interventions. This feature mimics real-world scenarios where automated processes might need human intervention or oversight at certain stages.

Restocking shelves is a crucial aspect of warehouse management. The project can offer options for both static and dynamic restocking of shelves—reflecting scenarios where restocking processes occur at fixed intervals as well as dynamically, depending on demand or supply variations.

Additionally, considering strategies for restocking based on the arrival of supply trucks at the warehouse could add another level of realism. This would involve synchronizing restocking activities with the incoming supply, reflecting real-world logistics and supply chain management.

In essence, by incorporating various elements like restocking stations, the integration of multiple warehouses, the allocation of specialized robots for specific tasks, and implementing diverse restocking mechanisms, this project can evolve to more accurately replicate real-world scenarios within automated warehouses.

In summary, enhancing this project with additional functionalities and intricacies aligns it closer to the realities of automated warehouses, paving the way for a more authentic simulation of these complex operational environments. The potential for this project's development is substantial, with numerous avenues available for further exploration and refinement.

References

1. Monkey and banana problem explained by Professor Lee.
2. Block words puzzle explained by Professor Lee
3. Stable models and constraints lectures from module 3 and module 4.