 doi:10.20944/preprints202011.0461.v1

Automate Warehouse

Name: Raju
Email: rsingh80@asu.edu

Abstract

The individual project report is intended to illustrate definitions and constraints for "automate warehouse" in the context of knowledge representation and reasoning.

Problem Statement

The automate warehouse scenario inscribe the process in which the robots delivers products to picking stations in order to fulfill the orders. Warehouse is represented in form of a rectangular grid. In this grid, robots can move freely, however the movement is limited to adjacent vertical and horizontal cells. The robots need to carry shelves with the required products to the respective matching picking stations. Robots specification are such that they are flat and can move underneath shelves to pick it up and move. Incase if the robot carrying a shelf does not fit under another shelf, the shelves need to be moved out of the way prior.

Constraints: To fulfill all orders within minimal time. The time is counted in steps, given each robot might perform one action per step. As the robots move around to pick up and to put down shelves, deliver the product or stay idle, there must not be any collisions. Given this, no robot moves to adjacent cells with another one from one step to the next. Also, the robots must not put down the shelves on highway boundary cells.

Project Background

Warehousing, an integral part of modern day supply chain system, provides an infrastructure that enables businesses to extract, cleanse, and store vast amounts of data, which is used by knowledge based algorithm to run the automation. The basic purpose of a data warehouse is to empower the knowledge workers with information that allows them to make decisions based on a solid foundation of fact. However, only a fraction of the needed information exists on computers; the vast majority of a firm's intellectual assets exist as knowledge in the minds of its employees. E-commerce firms such as Amazon, Target, Best Buy, Walmart maintains huge infrastructure for warehousing. These warehouses are fully or semi-automation in order to make it cost effective, time bound order processing and improve quality of the services. I propose a solution to automate the warehouse where all the order fulfilment request is taken care by robots adhering to the constraints to avoid collision.

I opted to use *Answer Set Programming* as it provides a robust and knowledge representation based approach to solve the problem statement. In ASP, we can define set of constrains which reduces the chances of collisions. Such constraints are technically challenging in conventional programming language. The choice of ASP is exemplary over conventional approaches as it reduces the number of iterations, conflict of logic, number of algorithms, time-space-compute challenges etc, which becomes prevalent for such scenarios in conventional approaches.

As a coding platform, I used *CLINGO* for *ASP*. For visualization, I used *ASPRILO*.

Approach to solve the problem.

Before I could, start, I realized that learning ASP and CLINGO holds the key to code the scenario. But this has to be supplemented with closer look at the requirement and understand the core constraints. Going through the project details, I could conclude that I need to create divide the work into two buckets: *collection of material process* and *verification strategy*. I researched and started to create rules. This was one of the most challenging aspect, as it behaves like an un-satisfiability loop. Then I adopted the behavior driven development methodologies (*BDD*), where we first think about the behavior of the robot, break it down into simple understandable rules, code these rules in *ASP*, and execute (troubleshoot) in *CLINGO*. Debugging, triaging are at the heart of the scripting satisfiable constraints. Once, I had separate satisfiable rules, I started on running integration specification on these rules iteratively, so that each rule can run in its entirety, without breaking the expected output. This way, I was able to code a robot, which can move to adjacent cells, performs all necessary actions such as shelves pick up, deliver to matching stations, avoid collisions.

Main results and analysis.

For this project, I created a 4x4 grid with highways along grid edges. The grid hosts product, shelves, pickup stations and order. The goal is of the order 2,2,0,m.

```
init(object(node,'n'),value(at, pair('x','y')))
where, the term 'n' denotes a grid cell at
coordinates positive integers 'x' and 'y'.


init(object(highway,'h'),value(at, pair('x','y')))
where, the term 'h' denotes a highway with coordinates
at 'x' and 'y'.


init(object(pickingStation,'p'),value(at, pair('x','y')))
where, the term 'p' denotes a picking station with
coordinates at 'x' and 'y'.

init(object(robot,'r'),value(at, pair('x','y')))
where, the term 'r' denotes a robot, initially
located at coordinates 'x' and 'y'.

init(object(shelf,'s'),value(at, pair('x','y')))
where, the term 's' denoted a shelf, initially
located at coordinates 'x' and 'y'.
```

Initial location of robot or shelf is unique, and shelves are not being carried out by any robot. With the given information, once we input the location of the shelves, robots, picking station, highways etc. Number of orders, kinds of products each order consists of and location of the shelves are provided as input to the program (as an input file.)

Given the constraints and input information, I coded the program in ASP.

Run the code:

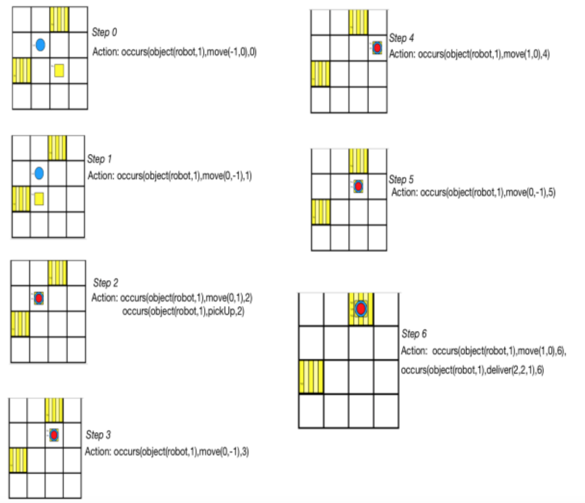
```
$ clingo warehouse.lp instance.asp -c n=4 -c m=7
```

Format the output:

```
occurs(object(robot,1),move(-1,0),0)

occurs(object(robot,1),move(0,-1),1)
occurs(object(robot,1),move(0,1),2)

occurs(object(robot,1),pickUp,2)

 occurs(object(robot,1),move(0,-1),3)

occurs(object(robot,1),move(1,0),4)

occurs(object(robot,1),move(0,-1),5)

occurs(object(robot,1),move(1,0),6),
occurs(object(robot,1),deliver(2,2,1),6)
```

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 November 2020 doi:10.20944/preprints202011.0461.v1

Snap:



Conclusion (Self-assessment)

This project has been an epitome of my learning in KRR course . I have gained an insight into the world of knowledge reasoning and programming. Developing the initial sight into the projects, breaking down the constraints to workable codes, running integration specs on the each constrain component without violating the rules are been core learning for me. Initially I took the approach to brute force logic and coding errors by hit and trail, but later I realized that, if I concentrate on the behaviour of the robot and apply constraints step by step, I could code it more intelligently.
The code is modular with different sections such as

Sort and object declaration
Action: move, pickup, deliver
Fluent
Constrains
Goals
Law of inertia

Constraints:

```
:- not 1{robot(R,X,Y,T)}1,  robot(R,X,Y,T),
T=0..m.
:- not 1{shelf(S,X,Y,T)}1, shelf(S), T=1..m.
:- not 1{product(I,S,U,T)}1, product(I), T=0..m.
:- not 1{order(O,I,U,T)}1, order(O,P), T=0..m.
:- not 1{carries(R,T,B):boolean(B)}1,
robot(R,X,Y,T),  T=0..m.
```

Opportunities for future work

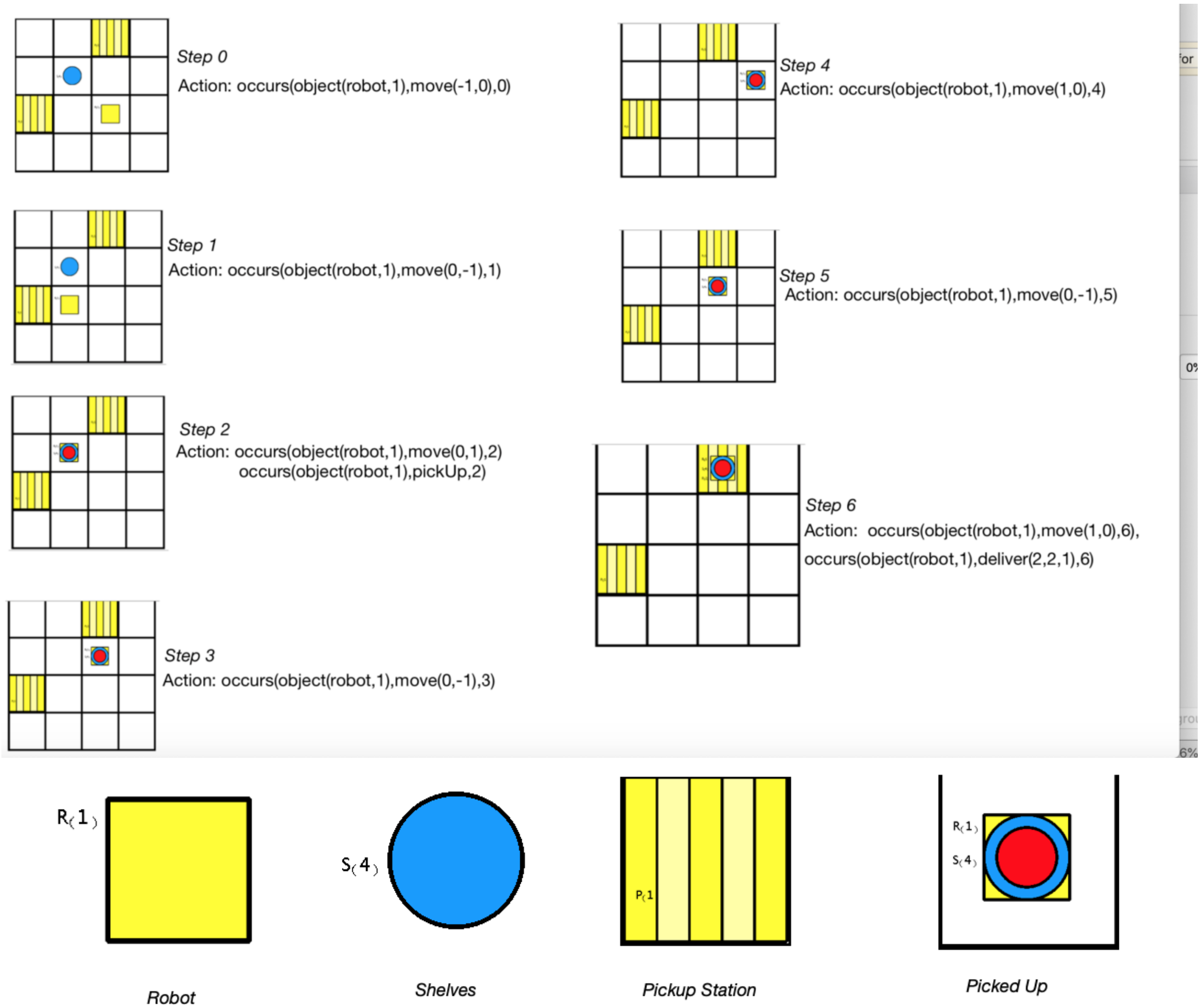
Warehouse is just an example to showcase how ASP (and KRR) plays a powerful role in solving complex real world problem. With increase in the complex nature of the problem, conventional approaches of computer science becomes less and less efficient. It is only when, logic based functional programming leads the way. The whole story telling revolves round the constraints and how KRR handles complex real world constraints with ease and in delicate way.

There are ample opportunities to use ASP (knowledge representation) to solve complex logic based problem in the field of *AI*, *Robotics*, *Computational Biology and Bio-informatics*, *Workforce management*, *Intelligent Call routing protocols (contact centres)*, *Decision Support Centres* etc.

ASP has been instrumental in solving cross domain complex problems as it allows recursive definitions, aggregates, weight constraints, optimization statements, default negations, and external atoms. This kind of inbuild expressiveness allows ASP to be used declaratively and solve combinatorial search problems (such as planning, diagnosis etc.) as well as knowledge intensive problems (such as query answering, explanation generation). In the recent time, ASP has been applied successfully to a wide range of areas in AI and other domain.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 November 2020 doi:10.20944/preprints202011.0461.v1

Figures



References

Custodio, L., Machado, R. Flexible automated warehouse: a literature review and an innovative framework. Int J Adv Manuf Technol 106, 533–558 (2020). https://doi.org/10.1007/s00170-019-04588-z

M. Schwarz *et al*., "NimbRo picking: Versatile part handling for warehouse automation," *2017 IEEE International Conference on Robotics and Automation (ICRA)*, Singapore, 2017, pp. 3032-3039, doi: 10.1109/ICRA.2017.7989348.

Hamberg, R. & Verriet, J. eds., 2012. Automation in Warehouse Development. Available at: http://dx.doi.org/10.1007/978-0-85729-968-0.

Guo, W.L. & Huang, Z.J., 2014. Analysis and Research of the Automated Warehouse Management and Control System. Applied Mechanics and Materials, 511-512, pp.1095–1098. Available at: http://dx.doi.org/10.4028/www.scientific.net/amm.511-512.1095.

Maciol, A., 2016. Knowledge-based methods for cost estimation of metal casts. The International Journal of Advanced Manufacturing Technology, 91(1-4), pp.641–656. Available at: http://dx.doi.org/10.1007/s00170-016-9704-z.

[1]F. Negrello, H. S. Stuart, and M. G. Catalano, "Hands in the Real World," Frontiers in Robotics and AI, vol. 6, Jan. 2020.

 doi:10.20944/preprints202011.0461.v1

[1]T. Beyer, P. Gohner, R. Yousefifar, and K.-H. Wehking, "Agent-based dimensioning to support the planning of Intra-Logistics systems," 2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA), Sep. 2016.

[1]S. Andova, M. G. J. van den Brand, L. J. P. Engelen, and T. Verhoeff, "MDE Basics with a DSL Focus," Lecture Notes in Computer Science, pp. 21–57, 2012.