Group Homework 1: Classical Planning

Robot Intelligence - Planning CS 4649/7649, Fall 2014
Instructor: Sungmoon Joo

Administrative

- **Due Date:** September 29 11:59PM, 2014
- **Deliverables:** Email the instructor a PDF summary that describes your work, results and answers to all the questions posed below. Thoroughness in analysis and answers in the reports are the primary component of your grade. The summary should list your group members' names. Attach a zip file that includes your code in an organized form with a README on how to run the code to re-create the results (DO NOT send movies, if any). Alternatively, you can use a repository on the course organization in the github(the instructor can create a repository upon request from a group), and commit the files (summary, code, etc.) to the repository. In that case, you can just send the instructor an email notice.
- **Participation:** Include a page in your summary describing what each group member did to participate in the project. If someone did not contribute, say so. Please be honest.
- **Printing:** On September 30, bring a printout of your summary to the class.

1 Warming up: Towers of Hanoi

A famous problem in classical planning is the Towers of Hanoi. Apparently, some priests in Vietnam are required to stack enormous discs from one tower to another by command of an ancient prophecy. Lets help them out with modern automation. The discs must always be stacked in order of increasing height. The goal is to move all the discs from the first tower to the third.

Experiment with at least two different classical planners to solve this problem. Links to the domain are provided on the course web page. The page also contains links to some recommended planners. You are welcome and recommended to try other planners as well.

1.1 Questions

a) Explain the method by which each of the two planners finds a solution.

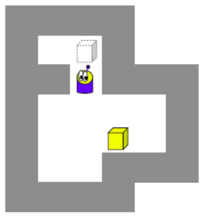
b) Which planner was fastest?

c) Explain why the winning planner might be more effective on this problem.

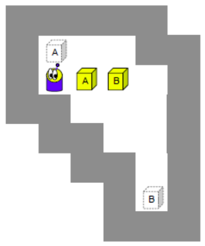
Figure 1: Towers of Hanoi with 3 discs

2 Sokoban PDDL

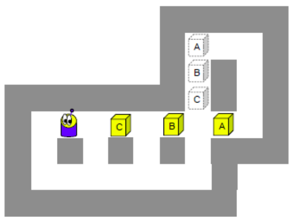
During the times of Pong, Pac-Man and Tetris, Hiroyuki Imabayashi created an complex game that tested the human abilities of planning: Sokoban. Many folks are still addicted to solving Sokoban puzzles and you can join them by playing any of the versions freely distributed on the web. The goal is for the human, or robot, to push all the boxes into the desired locations. The robot can move horizontally and vertically and push one box at a time.



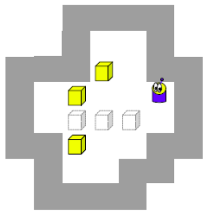
(a) Problem 2.1



(b) Problem 2.2



(c) Problem 2.3



(d) Sokoban Challenge

Figure 2: Sokoban Problemsi

Describe the Sokoban domain in PDDL. For each of the problems in Figure 2, define the problem in PDDL. You can either use the target lettering given in the picture or let the planner move any box to any target square. For the challenge problem, any box in any location is a solution. In the challenge, PDDL should NOT inform the robot which box should go to which location. In addition you may also try other problems you invent or find on the web. How well do your two planners perform on these problems? If no planner seems to be solving it, perhaps you should consider a different method for defining your problems.

2.1 Questions

a) Show successful plans from at least one planner on the three Sokoban problems in Figure 2(1-3). The challenge problem is optional.

b) Compare the performance of two planners on this domain. Which one works better? Does this make sense, why?

c) Clearly PDDL was not intended for this sort of application. Discuss the challenges in expressing geometric constraints in semantic planning.

d) In many cases, geometric and dynamic planning are insufficient to describe a domain. Give an example of a problem that is best suited for sematic (classical) planning. Explain why a semantic representation would be desirable.

3 Sokoban Challenge

You may now think that the difficulty of this problem has to do with PDDL. Your instructor seriously doubts that. Lets try an experiment. Define the Sokoban in any way that you like and write a computer program that solves it. Your instructor recommends you start with some kind of simple state space search and then see if you can make it faster. Test your planner on the sample problems in Figure 2 and any other examples you find interesting. Make sure that your planner addresses the following points:

- The planner must be complete.
- Your state representation should be efficient.
- The planner should make an effort to be fast.

To give you feedback on how well you accomplished these objectives we conduct an informal competition for who can produce the most optimal planner for the 4th problem(Figure 2.(d)). Purportedly this is a difficult problem for humans to solve - have a go at it before you let your planner do the work! If your planner can solve it, include the solution and report the following pieces of information:

- Computation time for Sokoban Challenge (and the other problems)
- Number of steps in your plan
- Number of states explored
- Language used
- Machine vitals

3.1 Questions

a) Give successful plans from your planner on the Sokoban problems in Figure 2 and any others.

b) Compare the performance of your planner to the PDDL planners you used in the previous problem. Which was faster? Why?

c) Prove that your planner was complete. Your instructor has a math background: a proof is a convincing argument. Make sure you address each aspect of completeness and why your planner satisfies it. Pictures are always welcome.

d) What methods did you use to speed up the planning? Give a short description of each method and explain why it did or didnt help on each relevant problem.

4 Towers of Hanoi Revisited

Having struggled to solve Imabayashi-sans puzzles, maybe we have a better feel for what planning is all about. Lets take another trip out to Hanoi. With modern automation, the monks aspire to even greater challenges. They can now solve Hanoi tower problems where there are 10 discs on the first tower. Can your planners from Problem 1 do this? Change the PDDL specification so that you can solve the problem with 10, and even 12 discs.

Figure 3: Towers of Hanoi with N discs

4.1 Questions

a) Give successful plans from at least one planner with 10 and 12 discs.

b) Do you notice anything about the structure of the plans? Can you use this to increase the efficiency of planning for Towers of Hanoi? Explain.

c) In a paragraph or two, explain a general planning strategy that would take advantage of problem structure. Make sure your strategy applies to problems other than Towers of Hanoi. Would such a planner still be complete?

5 Towers of Hanoi with HTN planning

We can encode our domain knowledge in the HTN planning framework. Let's see if we can get some performance improvement by using the domain knowledge actively.

5.1 Questions

a) Formulate a HTN planning problem for the Towers of Hanoi.

b) Describe the domain knowledge you encoded and resulting planning domain(i.e. primitive tasks, compound tasks, and methods) in detail.

c) Solve your HTN problem for the cases with $3 \backsim 12$ discs (use SHOP/SHOP2 or whatever HTN planner available to you).

d) Solve the same cases with a non-HTN planner of your choice, and compare the results with (c).

e) Describe your observations and discuss about them.