CompSci 367/761 ASSIGNMENT 3: Heuristic Search
Due 24 September 11:59PM worth 5%.

1 Introduction

In this assignment, we start moving into search algorithms proper. In assignment 2, you were asked to write a travelling salesman problem (TSP) solver, what is called a "domain-dependent" problem-solver. "Domain-independent" problem-solvers are much more interesting. One goal of AI is to develop general purpose problem solvers. Ones where you describe almost any type of problem to the system, and it automatically solves that problem.

In this assignment, you will be given the framework of a uninformed optimal search algorithm, called uniform cost search. Your assignment is to extend this framework to use heuristics to reduce the amount of searching needed to find an optimal solution. You will basically end up with A*.

2 Design

In order to make this system more domain independent, the domain dependent code has been separated out from the search code. There are the following files:

- solution.pl modify
- datastructures.pl modify
- tsp.pl
- h.pl modify
- script.pl
- counter.pl

solution.pl contains the search code, datastructures.pl contains all the datastructure creation, access, and modification code, tsp.pl contains the domain specific code, h.pl contains all the heuristic specific code, and script.pl contains the code for loading the files necessary to run problems. counter.pl contains code to implement the expanded node counter.

Currently, solution.pl, datastructures.pl, and tsp.pl are the base blind search version of this code. The current h.pl is simply the zero heuristic, you will need to add your code to make it the heuristic described in this document. The current script.pl just loads the files needed to run the blind search version of the code.

Currently, in solution/3 in solution.pl lines 21-22 are there in case you need to know the start and end states of the tour, which will found in the goal state. Those lines remove any records of old goal states and record the new goal state.

The goal state is usually useful in computing the heuristic. But in this heuristic, it might not be necessary.

The base code counts the number of nodes being expanded. This is printed at the end of solution/3.

3 What you need to do

All the code you have been given is designed for blind search, i.e., without a heuristic. Your assignment is to extend this code to handle using a heuristic to search for solutions. Specifically you will need to:

- write a specific heuristic (specified later in this document)
- modify solution.pl to take in and use a heuristic
- modify the datastructures.pl so that the data structures contain the heuristic h and f valued information for nodes (both open and closed)
- modify solution.pl and datastructures.pl so that the nodes in the open list are accessed in f-value order.

4 The heuristic h(+State, +RoadNetwork, ?HValue)

Given a state and a road network, the state's h value is the sum of the unvisited cities' min in-edge costs. This is the same as described in James' tutorial on 28 July.

5 Submission Information

1. What to submit

   You need to submit a zip archive, yourUpi.zip (e.g., mbar098.zip) containing ONLY the following files:

   - solution.pl
   - h.pl
   - datastructures.pl
   - results.pdf

   solution.pl, h.pl, and datastructures.pl are the files you needed to modify to extend the blind search algorithm to the A* search algorithm, and results.pdf is a table showing for each problem, the number of nodes expanded using the base blind version and those expanded with your extended heuristic version. Note that tsp.pl, script.pl, and counter.pl should not be in your submission.

You should expect that for complex problems that using the heuristics will significantly reduce the number of expanded nodes. E.g., the table in results.pl could look like the following:

problem	blind	heuristic
prob1	7345	1001
prob2	10789	1982
prob3	19888	2463
prob4	35347	3973

The problems will be given soon on Canvas. The data in the table above is just made-up and will not be what you get on the supplied problems. You should run both the base blind versions and your extended heuristic versions of the code on the given problems and report the result in the table above in results.pdf. The markers will compare your table against the results they get running your code on the supplied problems. Of course, the markers will run also your code on different problems than the ones you will be given. The other submitted files are extended versions of the base versions.

2. When and where to submit

   You need to submit this to Canvas by 24 September 23:59.

6 Marking Rubric

YOUR CODE WILL BE TESTED UNDER SWI PROLOG. IF IT DOES NOT RUN UNDER SWI PROLOG AS IS, IT WILL GET A ZERO!

1. The expanded nodes table (1 mark)

   The markers will give your solution/3 predicate (using your heuristic h/3) the given problems from canvas and check that they get the same results as in your table. If your table in results.pdf don't match what the markers get, then you get a zero for the assignment. Your code needs to solve at least trivial problems to get .5 mark and solve all the problems to get the full mark.

2. Getting the solution path in the correct direction (1 mark)

   The solution path returned needs to be optimal and needs to be in the correct sequence (from start city around all the other cities and back to the start city). The sum of the edge costs in the direction of the solution path need to equal the solution cost returned.

3. Correct heuristic value (1 mark)

   The heuristic h/3 needs to return the correct values for various states. We will put up some sample states and values on Canvas in a couple of days.

4. Solving somewhat trivial problems (1 mark)

   Need to be able to solve road networks with a single city and with a couple of cities. If can only solve with single city then only get .5 marks. Your code needs to be able to solve road networks with a couple of cities to get full mark.

5. Solving non-trivial problems (1 mark)

   Need to be able to solve complex road network problems to get full mark here.