COMP 343: Programming Project 1

Fall 2016

Your first programming project asks you to implement an informed search-based agent and compare its performance with five different heuristics for the Eights Puzzle.

The Program and Code

You must implement an agent for solving the Eights Puzzle and using informed search. You will then use that agent to complete exercise 3.32 from the text. This exercise calls upon you to compare the performance of these different heuristics. Your agent implementation should be modular enough that you can easily swap in different heuristics without needing to rewrite the agent proper. It should be developed independently of any one particular heuristic. You cannot simply implement five different agents each with the heuristic hard-coded. You *must* implement an agent with a modular heuristic function. All major programming paradigms support the techniques needed to do this in one way or another and you should be practicing working at this level of abstraction by this point in your career. In addition to the agent code itself you'll need code to generate random, solvable instances of the Eights puzzle and code that profiles the performance of your agent.

Your code should be neat, well documented, well organized, and well-tested for the sake of debugging and seeking out help from others. Your grade is not dependent on these things, but if you come seeking help it may not be rendered until tests and documentation are written for the parts of the program with which you're seeking help. Similarly, incomplete projects with poor documentation and specification are going to receive less credit than well written but incomplete projects. You may work in whatever language you choose. You should make use of libraries for core data structures whenever possible and should consider the availability of libraries for core data structures when choosing your algorithm.

Comparing Performance

We'll be measuring the performance of a given heuristic on three different metrics: quality of solution, time until solution, and memory needed to achieve solution. The best solutions minimize the overall cost of the solution as determined by the problem's cost function. For the Eights Puzzle this means solving the puzzle with the least amount of steps. Time until solution should be measured in clock time needed to compute the solution. Typically this is done by taking the time before and after the call to the agent's main puzzle solver method and then computing the difference between those times. Different languages provide different ways of collecting time data and offer that data at different levels of granularity. You should be as accurate as your tools allow. Memory needed to solve the problem should be measured by the maximum size of the search tree used in solving the problem. Measuring this is likely to require some extra code in the agent itself in order to count the total number of nodes added to the frontier of the search, i.e. total search tree nodes expanded.

Performance data should be gathered for at least 20 random instances of the puzzle. You may do more if you'd like. For each random instance, all five of the heuristics should be used to solve the puzzle in order to make accurate comparisons. That is to say, for each randomly generated instance of the problem you should have five data points per performance metric, one for each heuristic.

Presenting The Data

You'll present your findings with a simple write up and some basic data visualizations. Your write up must be no more than one page of text, excluding images, and should clearly lay out what data was collected, how the data was gathered, and the how to read the visualizations. You do not need to comment on the results or present an interpretation of the data, simply lay out the form, structure, and source of the data.

The data itself will be represented graphically. Minimally, you must provide a table for each performance metric that includes basic statistics like min, max, sample mean, and sample standard deviation each heuristic and each metric. Your are strongly encouraged, but not required, to develop point plots and scatter plots as well. To develop the point plots you can sort the performance data data by something like min, max, or mean and then plot them by rank order on the x axis and the metric on the y. This gives the reader some sense of speed across different instances of the problem as well as different heuristics. Alternatively, you could do bars and whiskers graphs instead of points. To enable deeper analysis you should do scatter plots with one performance metric on the x and one one the y. This can potentially highlight relationships between pairs of metrics. You could also do this in three dimensions using all three metrics, but that may or may not transfer well to printed medium. Again, you must at least produce a table and should seriously consider other visualizations. Tools like Mathematica, Matlab, and Octave can quickly produce the kinds of visualizations discussed above. Each visualization, including the table, should be well labeled and include a caption such that it may be examined independently of the text in your writeup.

Logistics

At the completion of the project you should submit your source code along with a brief, write up of your results that includes not more than one page of written text along with your graphical displays of your. The writeup should be printed and handed in directly. The source code should be submitted electronically by uploading it to the server and using the *handin* script. **The code is due by end of day on Tuesday 9/20 and the writeup is due in class the following day**.

Grades will be based on the validity of the data, the quality of its presentation, and the overall completeness of the project. Projects presenting data from a subset of the the heuristics will garner more partial credit than "complete code" without any apparent data or lacking any presentation of the data.