Programming Project 1

ECCS 2681

Wayne A. Campbell, Bryce Grey

4/08/16

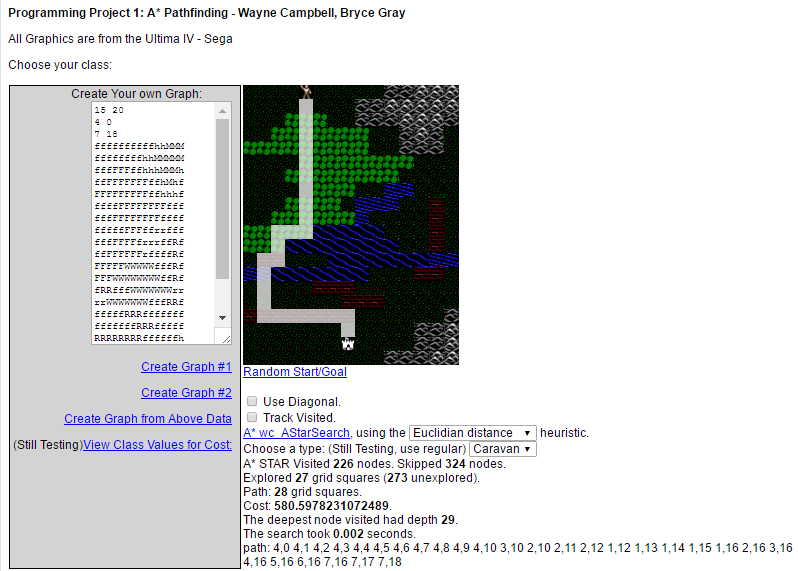
Implementation and Evaluation

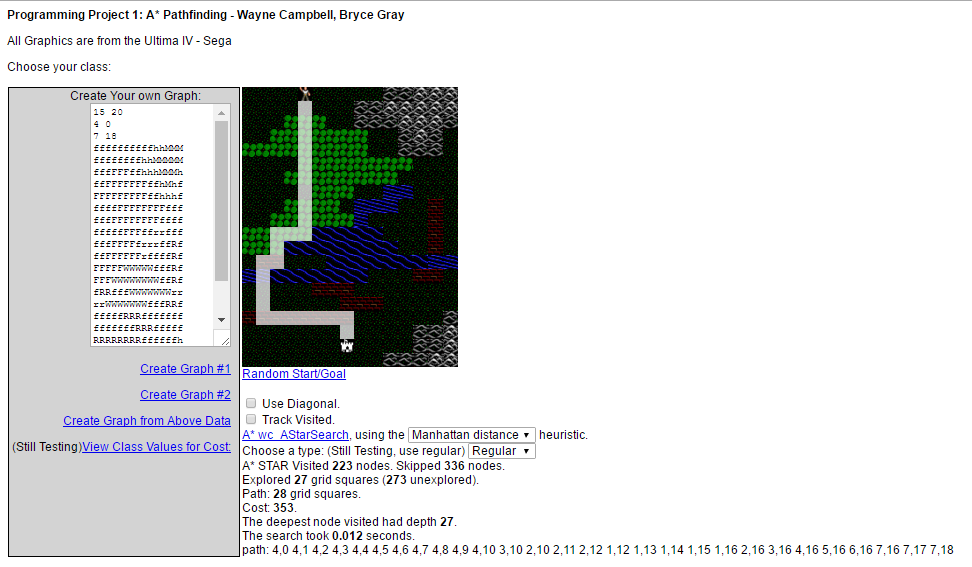
To implement the A\* algorithm with heuristics, we designed a JavaScript webpage that we could make maps, display the paths, and also display all the results for evaluation.

Because of our original proposal being very vague, we don’t see many changes from the actual implementation from the proposal. Instead of implementing the Manhattan, Diagonal, Euclidean, and other heuristics; We only Implemented Manhattan and Euclidean to simplify our original design. The other difference from our proposal, is that we have different and more test results from the algorithm. In the proposal document we defined the analysis exactly as the time it took the algorithm to find the path using the algorithm. This isn’t a good judge of analyzing different heuristics because of the fast computing time of newer processors and the efficiency of the A\* algorithm. Instead we have a graphical representation of the path found, and the nodes of the graph searched. Because of the proposal being very vague, I will describe our problem statement much more in the following paragraph.

The search problem in our implementation our problem is in the RTS domain, what path has the least cost by looking at the heuristic cost. In this graph(map) we have different terrain, each adding to the heuristic cost. The start point will have either 4 actions(movements) or 8 actions depending on the user input. Our cost will be defined by the heuristic cost plus the terrain cost.

Our implementation allows you to create a map, or use a map created already. It displays the map on the webpage, and allows you to select different heuristics. The option box to select what type the user is, is for the next project.





Comparing the Euclidian and Manhattan heuristics, when the start and destination points are in the same location, frequently the resultant paths differ slightly. This is because being different heuristics, they choose paths based of different algorithms and not only the paths are different because of it but also the resultant path costs. Generally, the resultant paths can only be called near-optimal, because the path sometimes ends up being sub-optimal. Also, for the next part of the project to consider a different type of optimization, we’ll compare the resultant paths found to paths picked by a group of users and see how A\* and different heuristics hold up to the user choice. Between heuristics not only do the paths differ, but also the amount and which nodes were searched to find the near-optimal path.

The cost value you see on the application currently is the total cost of the heuristic as a whole and not the cost of just the path found. For the next part of the application the path cost only will be added as well.