Kyle Price
U1018878
Homework09
Analysis Document

1. Who is your programming partner? Which of you submitted the source code of your program? My partner is Patrick Ekel. I submitted the sourced code.

2. Evaluate your programming partner. Pat was great as always. Our schedules fit together relatively well and we have a similar skill and experience level. He contributed just as much as I did for this assignment. I would rate him 10/10 on Yelp.

3. Does the straight-line distance (the absolute distance, ignoring any walls) from the start point to the goal point affect the running time of your algorithm? Yes. The more nodes you have to check, the more time it will take. The algorithm stops looking the moment it finds the 'G' node. If the straight-line distance is small, the program will terminate faster, resulting in a better run time.

4. Explain the difference between the straight-line distance and the actual solution path length. Give an example of a situation in which they differ greatly. How do each of them affect the running time of your algorithm? Which one is a more accurate indicator of run-time? They can be very different. A straight line solution doesn't take the walls into account. It's like a birds-eye view in a sense. In bigMaze, there are tons of walls all over the place. This is an example of when straight-line and the actual solution differ immensely. If there are no walls, they should be the same. The closer the actual solution is to the straight-line solution, the better the running time will be. However, that is rarely the case and the actual solution is a much more accurate representation of run time.

5. Assuming that the input maze is square (height and width are the same), consider the problem size, N to be the length of one side of the maze. What is the worst-case performance of your algorithm in Big-Oh notation? Your analysis should take in to account the density of the maze (how many wall segments there are in the field). For example, a completely open field with no walls other than the perimeter is not dense at all, as opposed to the example maze given"bigMaze.txt", which is very dense. There is no one correct answer to this since solutions may vary, but you must provide an analysis that shows you are thinking about the problem in a meaningful way related to your solution. The worst possible case is O(N^2). Depending on the implementation, the program could technically have to look at each node in the graph. If there are N number of items along each edge, the total number of items will simply be N*N => N^2. In my program, the loop still has to briefly look at nodes with 'X' in them (the walls). If the maze it was looking at was set up in such a way that the goal node was the last item it looked at, the program would have looked at each node in the graph and the big-Oh notation would be O(N^2). The more densely populated the maze is, the more likely the program is to look at more nodes. This is because the program's movement is restricted.

6. How many hours did you spend on this assignment? I would say we probably spent around 10 hours on this assignment.