Assignment 1

Bria Whitt (183008412), Kelly Jiang (185009352), Sandhya Veludandi (186008894)

October 22, 2021

1 Understanding the Methods

- A, initially starting at E2, moves to the east instead of north because it assumes all cells are unblocked, and can only check the blockage status of cells directly adjacent to it. Since the cell in the east is closest to T, and A can't see that the path is blocked yet, it goes to the east.
- n = unblocked cells, m = moves in one loop, l = amount of loops
  The worst case move m can make in one loop is n, which means m is going through every unblocked cell in the grid. The worst case of path computing (represented by l) would be n as well. To calculate the total number of moves the agent has made would be to multiply l and m, and in doing so we'd have to multiply the ns of both cases. So, it would be l*m ¡= n*n

2 The Effects of Ties

Repeated Forward A* that takes priority of the smaller g, let's refer to this as SG A*, is more efficient than the Repeated Forward A* that takes priority of the larger g, let's refer to this as LG A*. When we look at the amount of expanded cells that each method takes, SG A* has expanded less cells than in LG A*. With our first 5 test cases, we saw that on average, SG A* expanded 6,217 cells and LG A* expanded 7,226.4 cells. We believe that SG A* is more efficient because when the smaller g-value is prioritized, the agent is more likely to stay within the area that it has already explored. With LG A*, it's the opposite and the agent is more likely to explore new territory, and therefore expand more cells.

3 Forward vs. Backward

Repeated Forward A* is more efficient than Backward A* because it expanded less cells than Repeated Backward A*. We ran specific test cases for this problem, and more often than not, Forward A* would expand less cells than Backward A*. From our test cases, test cases 1 and 2 expanded less cells in Backward

A*, with Backward A* expanding 6,916 cells and Forward A* expanding 9,499 cells for test case 1 and Backward A* expanding 5,765 cells and Forward A* expanding 6,941 cells. However, test cases 3, 4, and 5 show that Forward A* expand less cells than Backward A*. On average, from these 5 test cases, Backward A* expands 7,407.2 cells and Forward A* expands 7,226.4 cells. Backward A* and Forward A* are similar except for the starting and ending points. We believe that Backward A* is less efficient than Forward A* because of how the algorithm approaches backwards, where it encounters cells differently enough to have it expand more.

4 Heuristics in the Adaptive A*

Manhattan distances are consistent in grid worlds in which the agent can move only in the four main compass directions because the only way it would not be consistent in these grid worlds is if the agent was able to move diagonally. Manhattan distances are given by the absolute value of the difference of the start and goal coordinates. This is just like moving the agent up or down and left or right to calculate the distance from the start to the target. The only way this distance would not be consistent is if the agent could move diagonally.

The Manhattan distances are consistent in grid worlds in which the agent can move only in the four main compass directions.

Let's assume that Manhattan distance is inconsistent.

h(n) >= c(n, a, n') + n(m) where c(n, a, n') is the cost of moving in 1 of 4 directions and h(n') is the cost from n' to goal, and h(n) is the cost from n to goal. Next if we move h(n') to the left side, we get

h(n) - h(n') >= c(n, a, n') using the Manhattan Distance, this can be written as:

$|x(n) - x(n')| + |y(n) - y(n')|$ and the only way that

$|x(n) - x(n')| + |y(n) - y(n')| >= c(n, a, n')$ is if we move diagonally, which we cannot do. Thus, Manhattan distances are consistent in grid worlds that only allow movement in the four compass directions.

Adaptive A* leaves initially consistent h-values consistent even if action costs increase because of how we calculate the h-values on the following iterations of A*. Adaptive A* uses the following heuristic function

h(s) = g(target) - g(s) for this to be consistent

h(n) <= cost(n, a, n') + h(n') substituting the heuristic function, we get

g(target) - g(n) <= c(n, a, n') + g(target) - g(n') simplifying this, we get

g(n') - g(n) <= c(n, a, n') If g(n') - g(n) is found using the Manhattan distance, we get the distance between n' and n, which is the same as c(n, a, n'). Therefore the new heuristic, h(s) = g(target) - g(s), is consistent too.

5 Heuristics in the Adaptive A*

Adaptive A* is more efficient than Repeated Forward A*, since it has a smaller number of expanded cells. For the 5 test cases we performed, Adaptive A* on average had a lower number of cells expanded than Forward A*, with Adaptive A* only expanding more cells than Forward A* on test case 4. Adaptive A* has less cells expanded due to it keeping an updated h value. The changing h value helps the heuristic become more accurate in Adaptive A*, which in turn makes the f values more accurate, so when choosing which cell to expand Adaptive A* is more informed and able to choose a more efficient path.
Test cases 1-5 Adaptive A* average: 6,995.4 expanded cells
Test cases 1-5 Forward A* average: 7,226.4 expanded cells

6 Statistical Significance

If we were to compare the performance differences between Repeated Forward A* Smallest G and Adaptive A*, we would measure the average run time. Our null hypothesis would be that there is no difference between the average run time of Repeated Forward A* Smallest G and Adaptive A*: avg(Repeated Forward A* Smallest G) == avg(Adaptive A*). Then, our alternative hypothesis would be that there is a performance difference between the search algorithms. Our method to get the average run times would to generate 100 random mazes, and then find the Repeated Forward A* Smallest G and the Adaptive A* runtime. Then, we would find the average runtime for each algorithm, and find the p-value. If the p-value is less than 0.05 then we will reject the null hypothesis that there is no performance difference between the 2 algorithms and accept the alternative hypothesis that there is a performance difference between Repeated Forward A* Smallest G and Adatpive A*.