RBE550 - Homework 2

Winston Crosby

Spring 2023

Breadth First Search

Figure 1: BFS with 0 Percent Obstacles

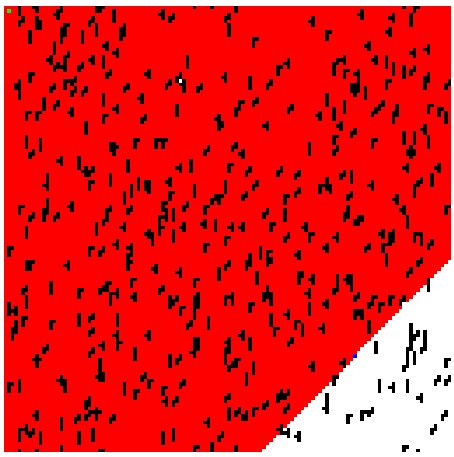

Figure 2: BFS with 10 Percent Obstacles

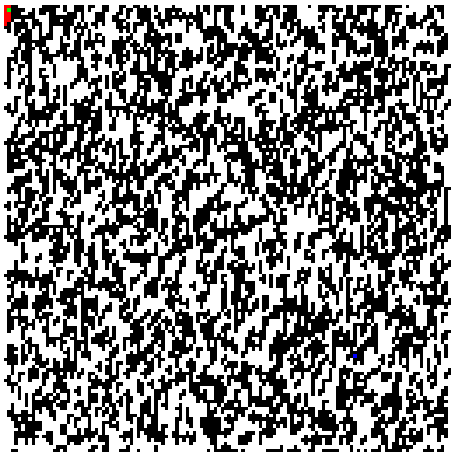

Figure 3: BFS with 50 Percent Obstacles

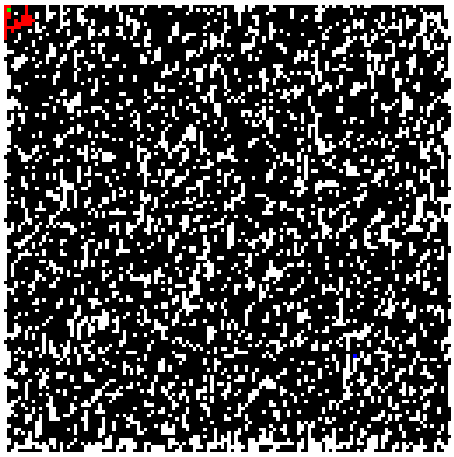

Figure 4: BFS with 75 Percent Obstacles

Depth First Search

Figure 5: DFS with 0 Percent Obstacles

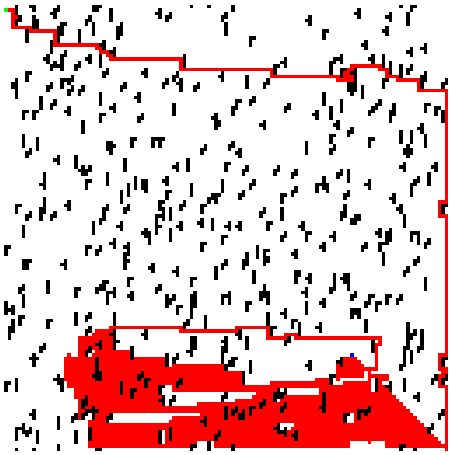

Figure 6: DFS with 10 Percent Obstacles

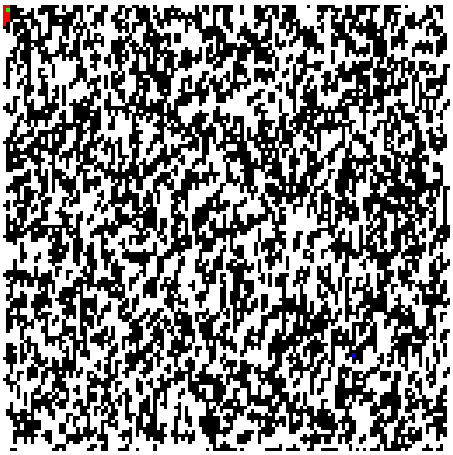

Figure 7: DFS with 50 Percent Obstacles

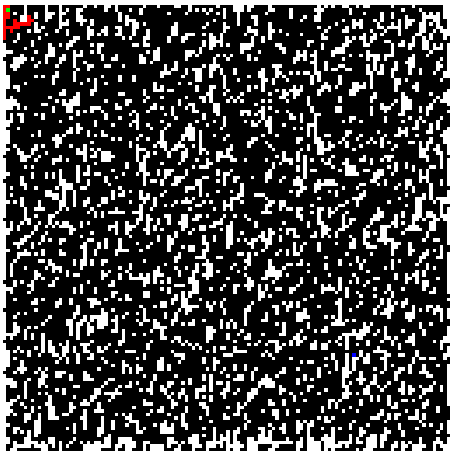

Figure 8: DFS with 75 Percent Obstacles

Dijkstra's Search

Figure 9: Dijkstra's with 0 Percent Obstacles

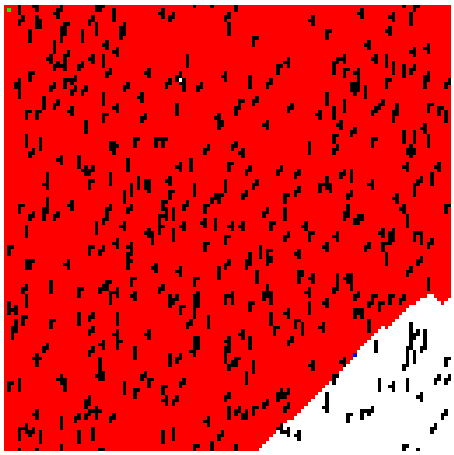

Figure 10: Dijkstra's with 10 Percent Obstacles

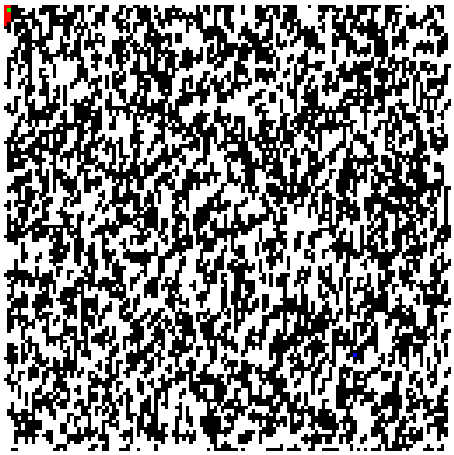

Figure 11: Dijkstra's with 50 Percent Obstacles

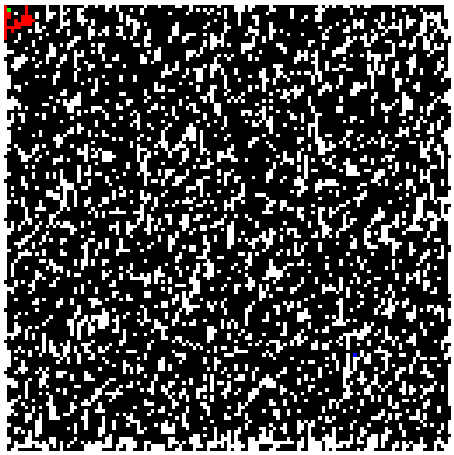

Figure 12: Dijkstra's with 75 Percent Obstacles

Random Search

Figure 13: Random Search with 0 Percent Obstacles

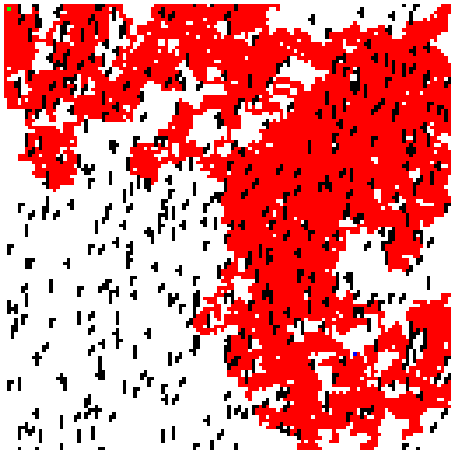

Figure 14: Random Search with 10 Percent Obstacles

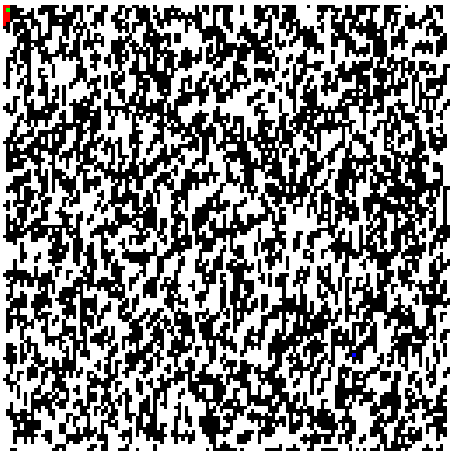

Figure 15: Random Search with 50 Percent Obstacles

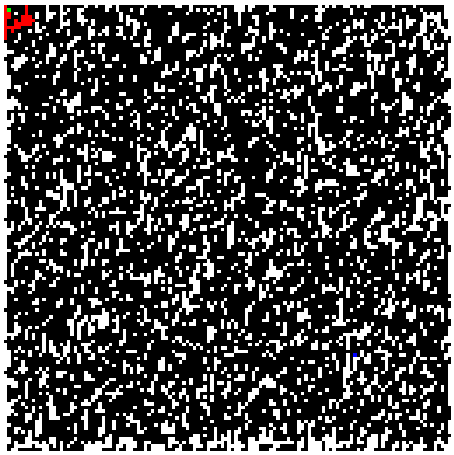

Figure 16: Random Search with 75 Percent Obstacles

Comparison of Algorithms

For all runs start position was at location (1, 1) and end position was at location (100, 100). Counts were measured but not typically consistent. Breadth first search and Dijkstra's search algorithms typically found the target location between 8,000 and 15,000 iterations. These searches showed the most improvement with increased obstacles, likely due to having less options to search. Depth first search was typically able to locate the end position between 500 and 4,000 iterations. Depth first search tended to vary the largest based on obstacles but was overall the most efficient. This may have been due to the placement of the target location relative to the start location, though, as the DFS algorithm was designed to search right, then down, then left, then up, depending on available tile. If the start and end positions were swapped it would likely show similar results to the BFS algorithm. Random search was most invariant to change but also showed the largest amount of variance. Random search could find the target location in as few as 300 iterations but could also fail to find the target location within the allotted time and was the only finished algorithm to occasionally fail below 50 percent obstacle fill. Because random search did not include a check on whether a location had been visited or not, a cap was placed on the number of iterations to prevent infinite loops. The cap was placed at 2,000,000 iterations which allowed for failure to be statistically unlikely, but possible.

At higher obstacle infill amounts, 40 percent and up, failure increased for all algorithms due to a lack of a clear path from the start location to the end location. All algorithms were still successfully able to exit after searching all available tiles within their start area, though.

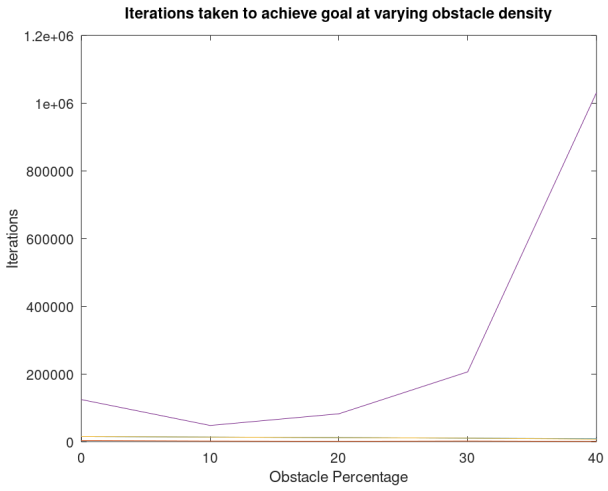

Figure 17: Iterations Comparison

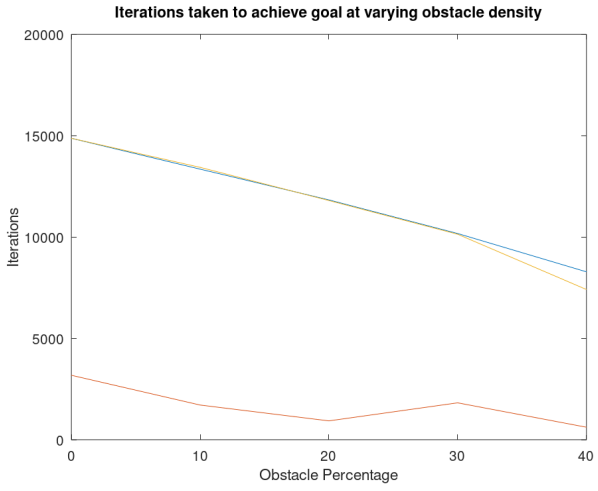

Figure 18: Iterations Comparison without Random Search

References