WITS UNIVERSITY

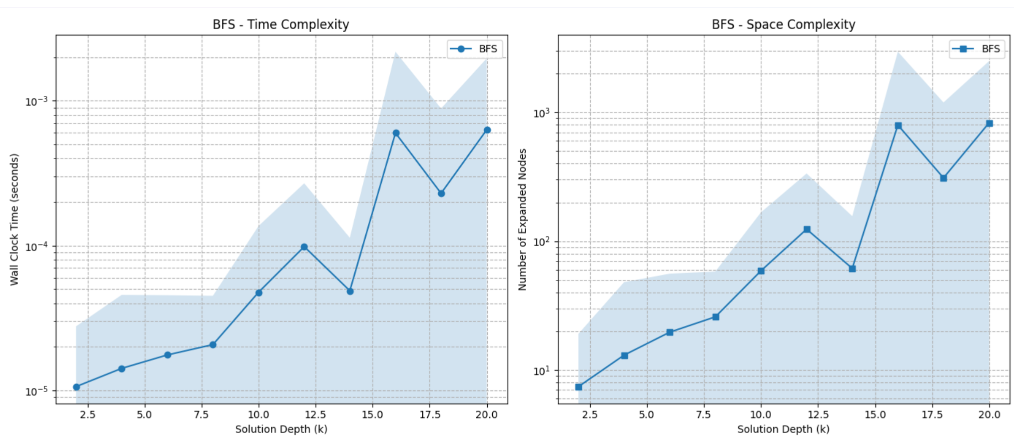
School of Computer Science and Applied Mathematics
UNIVERSITY OF THE WITWATERSRAND, JOHANNESBURG

COMS3005A lab2 – 2.4 Report

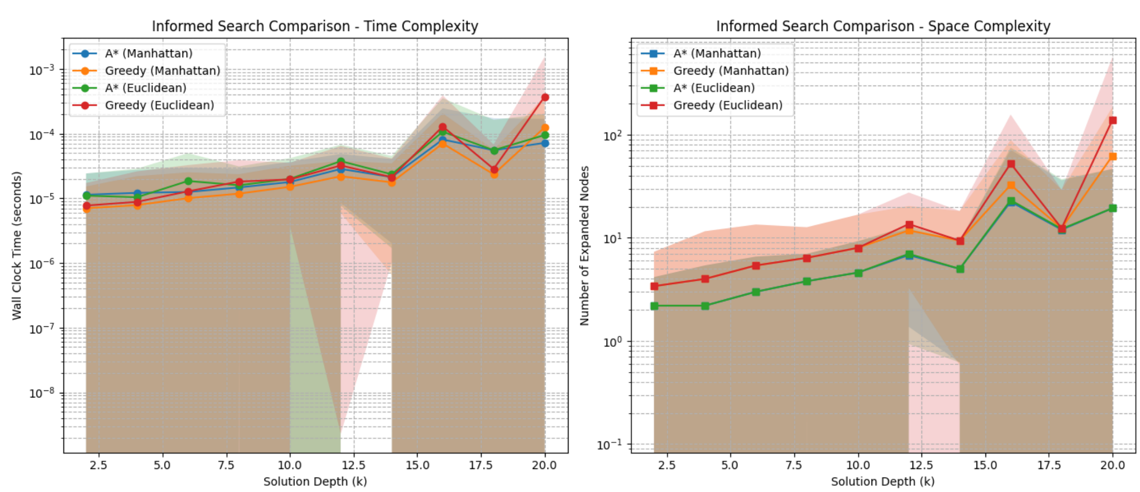
May Badenhorst, 2652142

1 Question 1: Time and Space complexity plots

Plots for BFS time and space complexity:



Plots for Showing the comparison of time and space complexity of Informed search algorithma:



2 Question 2: Short Paragraphs

2.1 The optimality of the greedy-BFS and A algorithms

A* is an optimal because it is guaranteed to find the shortest path solution, as long as its heuristic never overesitmates the true cost. It achieves this by balancing the cost of the path so far (g(n)) with the estimated cost to the goal (h(n)). This prevents it from committing to a path that looks promising initially but is actually very long.The above results confirm this, in every test case, the "Cost" found by A* (using either heuristic) was the same as the true shortest path found by BFS.

Greedy Best-First Search, on the other hand, is not optimal. It is a "greedy" algorithm that only considers the heuristic value (h(n)), completely ignoring the cost to reach the current state. This can lead it down a suboptimal path because a node that appears very close to the goal might have taken a very long and inefficient route to get there.

2.2 Time complexity using Manhattan distance as heuristic

Greedy Best-First Search (GBFS) expands nodes based only on the heuristic value h(n), such as the Manhattan distance, which makes it fast but not guaranteed to find the shortest path. Its time complexity in the worst case is still exponential, since it may get stuck exploring misleading paths. A* Search, on the other hand, uses both the actual cost so far g(n) and the heuristic h(n), ensuring optimality when Manhattan distance is used. Although A* also has a worst-case exponential time complexity, it usually explores far fewer nodes than GBFS and is more efficient and reliable at finding the optimal solution.

2.3 The effect of changing the heuristic function on the optimality of the algorithms

Changing the heuristic does not really affect the optimality of either algorithm,as long as the new heuristic is also admissible.A* stays optimal when switching from Manhattan to Euclidean distance. This is because the Euclidean distance is also an admissible heuristic for the puzzle and the straight-line distance is always less than or equal to the actual number of grid moves needed. Greedy-BFS was never optimal to begin with and this does not change with the change in heuristic. The algorithm's problem is its logic (ignoring path cost), not the heuristic it uses. A different heuristic will change the specific suboptimal path it finds, but it cant fix the underlying greedy nature that stops it from guaranteeing the best solution.

2.4 The effect of changing the heuristic function on the time complexity of the algorithms

Switching the heuristic from Manhattan to Euclidean doesn't change the theoretical time complexity of either algorithm.Both A* and Greedy BFS still have exponential worst-case behavior. What does change is how efficient they are in practice. For problems like the 8-puzzle or a grid where you can only move up, down, left, or right, Manhattan is usually the stronger heuristic because it matches the movement rules. Euclidean often gives smaller estimates, which makes it less informative, so both A* and GBFS will normally end up expanding more states and running slower. If diagonal moves were allowed, though, the opposite could happen: Euclidean would then be the better fit, guiding the search more effectively.This is affirmed by the above results