CS 4200 Project 1 Report

Ayomide Hakeem

July 6, 2025

1 Approach

The Implementation the A* algorithm for the 8-puzzle problem using two heuristics:

- **h1:** Number of misplaced tiles
- **h2:** Sum of Manhattan distances

The program handles both a single-puzzle and bulk testing modes (performed on 100+ random puzzle) then outputs the solution steps, search cost as well as the runtime. The program supports both single-puzzle and batch testing modes, and outputs solution steps, search cost, and runtime.

2 Comparison of Heuristics

I ran 100+ random puzzles for each solution depth (13–20) and recorded the average search cost for both heuristics.

Table 1: Average search cost for $A^*(h_1)$ vs. $A^*(h_2)$.

Depth (d)	$A^*(h_1)$	$A^*(h_2)$
13	137	58
14	225	53
15	394	81
16	537	119
17	909	189
18	1369	247
19	2256	311
20	3219	428

Observation:

- A* with Manhattan distance (h_2) is consistently more efficient than A* with misplaced tiles (h_1), especially as the solution depth increases. this can be seen at depth 20 where $A^*(h_2)$ expands only 428 nodes on average, while $A^*(h_1)$ expands 3219 nodes.
- The difference in efficiency grows with depth, and we can therefore conclude that a more informed heuristic (h_2) leads to a much smaller search space.
- These results are pretty much in sync with the benchmark table provided in the assignment documentation, though absolute numbers may vary due to random puzzle generation and implementation details.

3 Conclusion

- Via this project, I was able to deepen my understanding of heuristic search, particularly learning how to generate and verify solvable 8-puzzle instances.
- The batch testing harness made it easy to collect and analyze results.

4 Output Files

All output files (CSV, sample runs) are included and formatted as required. I also have a function that performs a summary on the results of the batch testing