github.com/sagedurivage/eight-puzzle-solver Sage DuRivage
CS170 sduri001

Project 1

The Eight Puzzle

1. Challenges in This Project

The primary challenge for me in this project was implementing the Euclidean Distance Heuristic and A* Search functionality to produce an accurate resulting trace.

2. Design (Objects & Methods)

I organized the program around four main classes: Node, Puzzle, Heuristic, and Solver.

- The Node class represents each state in the puzzle, with links to parent nodes to trace the solution path.
- The Puzzle class maintains the initial and goal states, as well as a getLineage method to identify possible successive states.
- The Heuristic class provides methods for the Euclidean Distance and Misplaced Tile heuristics used in A*, as well as a zero cost heuristic to turn A* into Uniform Cost Search.

3. Code Optimization

To optimize performance, I used a priority_queue for the frontier so that nodes with the lowest total cost values were expanded first.

4. Graph Search

I implemented a graph search by keeping a record of explored states in a set to avoid redundant expansions and looping. I did not implement a tree search method, though I would expect that graph search requires more memory for tracking explored states.

5. Comparing Heuristic Functions *

The Misplaced Tile heuristic was simple to compute but less accurate in estimating the actual cost, leading to a higher number of node expansions on average. The Euclidean Distance heuristic expanded fewer nodes, indicating a trade-off between accuracy and computational expense. Uniform Cost Search, expanded significantly more nodes due to the lack of heuristic guidance.

6. Findings

This project revealed that for shallower puzzle states, the choice of heuristic has little impact on search efficiency. As puzzles become more complex, heuristics are more involved in reducing the search space and improving performance.
Euclidean Distance was generally more effective in minimizing node expansions, while the Misplaced Tile heuristic balanced the computational cost. Uniform Cost Search proved inefficient for complex puzzles due to the lack of heuristic information.

* Tables and diagrams on following page

Tables and Diagrams

Number of Nodes Expanded

	Uniform Cost	Misplaced Tile	Euclidean Distance
Trivial (0)	1	1	1
Very Easy (1)	2	2	2
Easy (2)	3	3	3
Doable (3)	5	5	5
Oh Boy (4)	23	23	23
Impossible (5)	32	32	33

Maximum Queue Size

	Uniform Cost	Misplaced Tile	Euclidean Distance
Trivial (0)	1	1	1
Very Easy (1)	3	3	3
Easy (2)	6	6	3
Doable (3)	16	16	4
Oh Boy (4)	59818	59818	957
Impossible (5)	71274	71274	37868