David Vo
7/16/2023
CS 4200

Project 1 Report

I. Findings:

In this report, we explore the classic 8-puzzle problem and its solution using the A* search algorithm with two different heuristic functions: the Hamming distance and the Manhattan distance. The purpose of this program is to implement and compare the performance of these heuristics in solving the 8-puzzle problem. The way I approached writing the program was that I created 2 methods: solveSinglePuzzle and solveMultiplePuzzle. These 2 methods are the heart of the program since they are in charge of all the prompts, calling other methods to take in the puzzle(s) from the user or to generate random puzzle(s) and solve it with method calculateHeuristic or calculateHeuristic2 based on what the user asks for. They also handle the calculation of search costs and the time taken to solve each puzzle. Besides solveSinglePuzzle and solveMultiplePuzzle methods, other methods have different responsibilities. One of the most important ones is the runAStarSearchWithDepth method which checks if the puzzle is solvable. If the puzzle is solvable, it will solve the puzzle step by step while keeping track to update the search cost.

The biggest taking away from the project for me is the problem-solving skills to develop an 8-puzzle solver. It was a very challenging project I spent a lot of time with to break down complex problems in order to come up with a solution. This project also helps me to understand data structure better such as priority queues, hash sets, and arrays. Understanding when and how to use these data structures can significantly impact the speed and memory usage of my program.

II. Table Analysis:

Search Cost:

Counter for each h1 and h2	d	Average t (ms)	A*(h1)	A*(h2)
10	2	0.05 0.02	5	2
10	4	0.70, 0.20	15	5
10	8	0.90, 0.67	94	30
10	12	1.86, 0.75	445	191
10	16	2.75, 0.80	2891	446
10	20	4.67, 2.00	6883	4186

Total cases = (10 + 10 + 10 +10 +10 +10) *2 = 120 cases.