EE698G

MUTI-HEURISTIC A* ALGORITHM

Submitted by PRANAV S(14479) , PRATEEK ROY(14490)

INTRODUCTION

Heuristic A* is a search algorithm that is used extensively in path finding and graph traversal problems to find efficient paths through nodes. It enjoys a high use due to its efficiency and accuracy. A* achieves better performance by using heuristics to guide its search. A* is an informed search algorithm, or a best-first search, meaning that it solves problems by searching among all possible paths to the solution for the one that incurs the smallest cost and among these paths it first considers the ones that appear to lead most quickly to the solution. At each iteration of its main loop, A* needs to determine which of its partial paths to expand into one or more longer paths. It does so based on an estimate of the cost to go to the goal node. Specifically, A* selects the path that minimizes a function f(n) which is the sum of two functions g(n) and h(n) where they denote the cost of the path from the start node to n^{th} node and an estimate of the cost of the cheapest path from the n^{th} node to that goal. The function h(n) is called the heuristic function. For the algorithm to find the actual shortest path, the heuristic function must be admissible, meaning that it never overestimates the actual cost to get to the nearest goal node.

The efficiency of the A* algorithm is dependent on the heuristic function used. These algorithms work fast and generate high-quality solutions as long as they are given a well-designed heuristic function. However with the growing complexities of problems it is proving to be more difficult to design a single heuristic function that captures all the intricacies of the situation especially when it comes to the more versatile problems. Multi-Heuristic A* algorithm is an improved version that takes in multiple, arbitrarily inadmissible heuristic functions in addition to a single consistent heuristic, and uses all of them simultaneously to search for a complete and bounded suboptimal solution. Multi-Heuristic A* algorithm can be used in various scenarios from path planning for a robot to solving a puzzle game. This project uses a Sliding Tile Puzzle in order to do a quantitative analysis of the difference between a single heuristic A* and multi heuristic A* algorithm.

SLIDING TILE PUZZLE

The sliding tile puzzle is a combination puzzle in which the goal is to rearrange the tiles by sliding them into the vacant space available in order to put the puzzle into its original undisturbed form. In order to solve this puzzle using the A* algorithm, we define a node as a particular state of the puzzle. If one state can be reached from another with just one move then those two nodes are connected by an edge. Initially the puzzle is in one of the random states. In order to reach the final state an A* algorithm is used to traverse the nodes and edges to find the optimal solution or the shortest path to the goal. The A* algorithms traditionally used for solving this puzzle uses heuristic functions like Manhattan distance, Linear Conflict, Misplaced Tiles, N-Maxswap, Tiles out of row and column, etc. While useful all these functions have their own strengths and weaknesses depending on the complexity of the puzzle and the number of tiles in it. This experiment was designed in order to have a quantitative analysis of the difference in efficiency of using a single heuristic function versus using multiple heuristic functions. In this case the efficiency is determined by the number of unique nodes that are being explored in order to find the correct solution. The experiment was conducted with 3x3, 4x4, and 5x5 puzzles. The complexity of the puzzle was defined by the least number of steps that are necessary to reach the goal. The heuristic functions used in this experiment are:

1. Manhattan Distance: It is the distance between two points measured along axes at right angles.
2. Linear Conflict: Two tiles p and q are in a linear conflict if p and q are in the same line, the goal positions of p and q are both in that line, p is to the right of q and goal position of p is to the left of the goal position of q.
3. Misplaced Tiles: This refers the number of tiles that are not in their final position.
4. Tiles out of row and column: This refers to the number of tiles that are not in the same row or column as their final position.

Many of these heuristic functions seem almost redundant in first due to the fact that they are similar whether it be tiles just out of their position or if they are out of the correct row or column. But only the result can judge their effectiveness and independence from each other.