Joshua Cords

Tang

CS 420

24 October 2016

A\* - Project 1 Report

**Requirements**

Create A\* algorithm implemented on an 8-Puzzle. Use two heuristics, Misplaced Tiles and Manhattan Distance and compare their results in search space and time.

Input User selects heuristic and either inputs a puzzle, generates a random puzzle, or reads puzzles in from a file.

Output Outputs puzzle solution along with the number of nodes created and solution duration.

**Method**

Implemented A\* pseudocode and Misplaced Tiles and Manhattan Distance heuristics. A timer was used to keep track of the time elapsed, via System.nanoTime, the number of nodes created was stored and results were rounded to the nearest whole number.

Misplaced Tiles heuristic calculates h() by counting the number of misplaced tiles on an 8-puzzle, tiles include the numbers 1-8. Manhattan Distance heuristic is more complicated; it sums the number of spaces away each tile is from its goal position. While this calculation with take more computation, the Manhattan Distance heuristic should dominate the Misplaced Tiles heuristic and should out-perform it on large enough solutions.

**Testing**

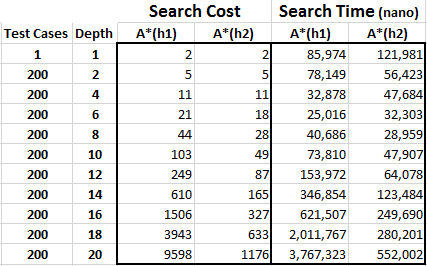
Each heuristic was tested over 200 sample puzzles of each even size from 2 to 20. Additionally, a puzzle of solution depth 1 was tested 200 times before the main execution in an attempt to reduce the impact of a ramp up factor on the results.

**Findings**

Here are the output and results from the two A\* Heuristics.

h1 : Misplaced Tiles heuristic

h2 : Manhattan Distance heuristic



**Analysis**

After analyzing the results of the experiment, it was clear the Manhattan Distance heuristic begins to vastly outperformed the Misplaced Tiles heuristic at a solution depth of 8. At that solution depth h2 performed 24% in Search Cost and 17% better in Search Time. By the average puzzle solution depth of 20, h2 is performing 78% and 74% better respectively.

From the results, it appears that puzzles of solution depths less than 6 do not follow the expected pattern and take longer to solve than puzzles of larger solution depths and Search Costs. This is likely attributed to the algorithm and driver overhead as well as the computer’s ramp up speed as it allocates resources to the executing program.

Interestingly it seems the Misplaced Tiles heuristic outperforms the Manhattan Distance heuristic in Search Time at solution depth 6. This is likely due to the increased computations of the heuristic without a large decrease in Search Space. However, as the Search Space differential increases the extra computational overhead is overshadowed by larger Search Space that must be explored.