**University of Alaska, Anchorage**

**CSCE A405, Artificial Intelligence Fall 2020**

**Dr. Frank Moore**

**Programming Assignment 1:**

**Informed Search Techniques**

**Andrew Baluyut:** [**adbaluyut@alaska.edu**](mailto:adbaluyut@alaska.edu)

**Hannah Crayton:** [**hecrayton@alaska.edu**](mailto:hecrayton@alaska.edu)

**Description of Each Search Technique**

Breadth-First Search:

* Our function bfs() implements this search algorithm.
* It receives parameters start\_state and goal\_state.
* BFS looks to expand nodes across the same level until it expands into the goal state.
* Using FIFO, it implements f(n) = g(n) functionality. G(n) = a weight of 1.
* It is the least efficient way of our 4 techniques to arrive to the goal.

Greedy Best-First Search using the Misplaced Tiles heuristic:

* Our function gbfs() implements this search algorithm.
* It receives parameters start\_state and goal\_state.
* GBFS looks to expand nodes based on which node has the state with the least misplaced tiles, when a weight minimum is discovered on >1 nodes of the same level, it will implement breadth-first search functionality to continue forward.
* Using a sorted queue, it implements f(n) = h(n) functionality. H(n) = # of misplaced tiles.

A\* Search using the Misplaced Tiles heuristic:

* Our function aStarMisplacedTiles() implements this search algorithm.
* It receives parameters start\_state and goal\_state.
* A\* looks to expand nodes based off the sum of: the cost it took to be in its current location + the estimated cost of arriving to its goal via a given path. In this case, the cost is given off of the # of misplaced tiles.
* Using a sorted queue, it implements f(n) = g(n) + h(n) functionality. G(n) = cost to be at current node. H(n) = estimated cost (based off misplaced tiles) to reach goal.

A\* Search using the Manhattan Distance heuristic:

* Our function aStarManhattanDistance() implements this search algorithm.
* It receives parameters start\_state and goal\_state.
* A\* looks to expand nodes based off the sum of: the cost it took to be in its current location + the estimated cost of arriving to its goal via a given path. In this case, the cost is given off of the Manhattan distance.
* Using a sorted queue, it implements f(n) = g(n) + h(n) functionality. G(n) = cost to be at current node. H(n) = estimated cost (based off Manhattan distance) to reach goal.

**Summary of Test Results**

These are our results for each search technique for several representative runs. We tested our solution for 10 start,goal pairs with a variety of minimum solution path lengths. We note the relationship between the number of nodes expanded during each search and the length of the solution path. We note the differences in the performance of each search technique. Finally, we identify characteristics of the puzzle problem that make it easier, or more difficult to solve.

-Enter results and answers here.-

We are submitting our report along with our .py files.

\*\*PS.