­

CS170 – AI

8-Puzzle Algorithms

Rashid Goshtasbi

Python

Resources Listed:

Sorting Algorithm for PuzzleList in code:

<http://quiz.geeksforgeeks.org/bubble-sort/>

class setup for python:

<http://codereview.stackexchange.com/questions/76906/python-8-puzzle-and-solver>

idea for setting up uniform costs search:

<http://stackoverflow.com/questions/12755893/uniform-cost-search-algorithm>

Report:

Puzzle: 1 2 3

4 8 0

7 6 5

Results:

* Uniform Cost Search:
  + Nodes Expanded: 509
  + Size of Queue: 328
  + Depth: 5
* Misplaced Tile Heuristic
  + Nodes Expanded: 33
  + Size of Queue: 23
  + Depth: 5
* Manhattan Distance Heuristic:
  + Nodes Expanded: 15
  + Size of Queue: 11
  + Depth: 5
* Thoughts
  + Manhattan was very effective in this test because with the high amount of nodes expanded, we can take the best distance and costs and make less moves based off it
  + Misplaced tile wasn’t much off either and still a huge different than uniform cost search thanks to the count of checking is the tile is misplaced or not but since it doesn’t take in account how far the tile is from the correct spot, it cost us an additional 18 more moves

Puzzle: 1 2 3

4 0 6

7 5 8

Results:

* Uniform Cost Search:
  + Nodes Expanded: 30
  + Size of Queue: 21
  + Depth: 2
* Misplaced Tile Heuristic
  + Nodes Expanded: 7
  + Size of Queue: 6
  + Depth: 2
* Manhattan Distance Heuristic:
  + Nodes Expanded: 7
  + Size of Queue: 6
  + Depth: 2
* Thoughts
  + Not sure if algorithm is wrong or if misplaced tile was enough for this puzzle as it was the same for Manhattan distance. If it was right, then my assumption was that calculating if the tile was correct or not was more than

Puzzle: 1 2 3

4 5 6

7 8 0

Results:

* Uniform Cost Search:
  + Nodes Expanded: 0
  + Size of Queue: 0
  + Depth: 0
* Misplaced Tile Heuristic
  + Nodes Expanded: 0
  + Size of Queue: 0
  + Depth: 0
* Manhattan Distance Heuristic:
  + Nodes Expanded: 0
  + Size of Queue: 0
  + Depth: 0
* Thoughts
  + This was test case to show that algorithm recognizes base case scenarios

Final Report: I think my algorithm would’ve worked better if I didn’t use bubblesort but I kept on redoing my program because I couldn’t get a solid foundation. Mispalced Tile Heuristic was good to work with even for best case because it would calculate the costs of the tiles if they’re in the correct place or not. Manhattan was better just because it would calculate what misplaced tile heuristic would but also calculate how far it was from the distance tile it should be placed at. That’s why so many less puzzles were queued to try because of it.