Josiah Lee
jlee434
861 287 993

CS170 Project 1 Report

Design

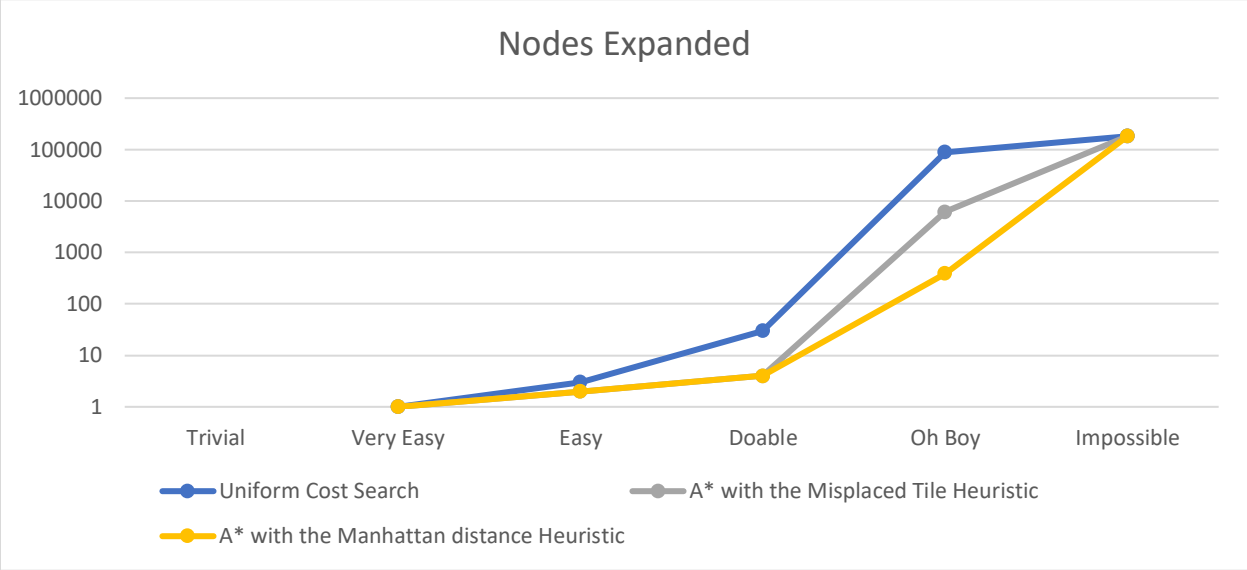
- Objects
  - puzz8 object
    - values
      - g - weight from distance from initial
      - h - weight from distance from goal, measured by heuristic
      - puzz - array of length 9, with puzzle
      - priority – g + h
      - parent – pointer to parent object
    - functions
      - print – prints puzzle in 3x3 format
      - updatePriority – sets priority to g+h
- Helper Functions
  - expand - Adds nodes to frontier if not explored yet (max of 4 added possible)
  - switch - Creates a new puzzle with two spots switched
    - Allows for easy node generation
  - find - Finds a value in an array, and returns an index
    - Allows switch problem to know which indexes to switch
  - **isValidPuzz8** (Checks if puzzle is impossible and valid)
    - Checks first if array is length 9 and that 0-8 are present
    - Sums up inversions, checks if its even
    - If all checks pass return 1, else return 0
  - **printParents** (Traces up to root)
    - Follows Parent pointer up till root, then prints all in order
- I implemented a graph search, and compared three heuristics
  - No heuristic - Uniform Cost Search
  - Misplaced Tiles
  - Manhattan Distance
- Explored – **immutable set**, checks for duplicates inherently
- Frontier – **heap priority queue**, compares objects allows for easy pop push

Findings

- As the problems grow more difficult, the more important the heuristic becomes
- For Trivial or Easy problems the heuristic barely improves performance
- A good heuristic can be far better than an alright heuristic, and an alright heuristic is far better than no heuristic

Test Cases

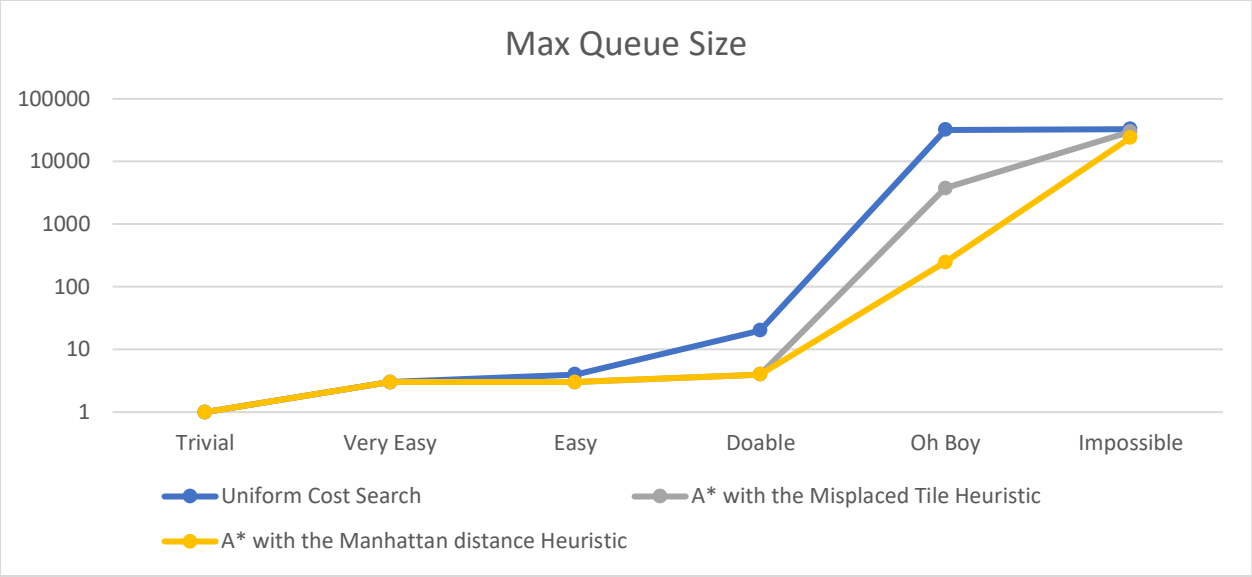
Trivial	Very Easy	East	Doable	Oh Boy	Impossible
1 2 3 4 5 6 7 8 0	1 2 3 4 5 6 7 0 8	1 2 0 4 5 3 7 8 6	0 1 2 4 5 3 7 8 6	8 7 1 6 0 2 5 4 3	1 2 3 4 5 6 8 7 0
Solved in 0 Moves	Solved in 1 Moves	Solved in 2 Moves	Solved in 4 Moves	Solved in 22 Moves	Unsolvable



Nodes Expanded

	Trivial	Very Easy	Easy	Doable	Oh Boy	Impossible
Uniform Cost Search	0	1	3	30	89179	181440
A* with the Misplaced Tile Heuristic	0	1	2	4	6145	181440
A* with the Manhattan distance Heuristic	0	1	2	4	388	181440

- For the simpler puzzles, trivial, very easy, and easy, all three searches expanded similar amounts of nodes
- For the Doable puzzle, the A* searches began to outstrip the Uniform Cost Search
- The Manhattan Distance and Misplaced Tile Heuristics Performed exactly the same from the Doable puzzle, but showed extreme differences for the Oh Boy puzzle, with the Manhattan distance Heuristic beating the Misplaced Tile Heuristic by almost a factor of 20
- All three algorithms expanded the maximum of 181,440, or (9!/2), nodes for the impossible puzzle before failing.



Max Queue Size

	Trivial	Very Easy	Easy	Doable	Oh Boy	Impossible
Uniform Cost Search	1	3	4	20	32152	32828
A* with the Misplaced Tile Heuristic	1	3	3	4	3753	29883
A* with the Manhattan distance Heuristic	1	3	3	4	248	24005

- Space wise the three algorithms performed similarly for the Trivial, Very Easy, and Easy Puzzles.
- For the Doable Puzzle the Uniform Cost Search Used a bit less than 10x the space as the two A* searches.
- For the Oh Boy, the Manhattan Distance Heuristic shines as it performs almost 100x better than the Uniform Cost Search and 10x better than the Misplaced Tile Heuristic.
- For the Impossible Puzzle, all three Algorithms used up similar amounts of space.
- It is important to note that the Uniform cost search performed just as poorly on the Oh Boy puzzle and the Impossible puzzle, suggesting that after a certain point all puzzles are the same space wise, and that point comes very early for the Uniform Cost Search