COMP20003 Algorithms and Data Structures

Assignment 2 Experimentation Report

Patrick Tjahjadi (890003)

1. Introduction

The assignment involves the use of the IDA* (Iterative Deepening A-Star) to solve the 15-puzzle. This is done through recursively to solve the puzzle optimally through graph search: Depth First Search strategy. The heuristic used is the Manhattan Distance heuristic. Several experimentations using sample test cases are shown below, displaying the following attributes:

- Initial state of the puzzle and ID N.
- $h(s_0)$, heuristic estimate for the initial state.
- Thresholds the search have used to solve the puzzle.
- Solution: Number of moves optimized to solve the puzzle.
- Generated: Number of generated nodes.
- Expanded: Number of expanded nodes.
- Number of expanded nodes per second.
- Time taken to search, in seconds.

The experiment is run using the dimefox server.

2. Experimentation

N	Initial State	$h(s_0)$	Thresholds	Solution	Generated	Expanded	Nodes/Sec	Time
1	14 13 5 7 11 12 9 5 6 0 2 1 4 8 10 3	41	41, 43, 45, 47, 49, 51, 53, 55, 57	57				
2	13 5 4 10 9 12 8 14 2 3 7 1 0 15 11 6	43	43, 45, 47, 49, 51, 53, 55	55	462,522,978	200,763,945	4,516,396	44.45
3	14 7 8 2 13 11 10 4 9 12 5 0 3 6 1 15	41	41, 43, 45, 47, 49, 51, 53, 55, 57, 59	59				
4	5 12 10 7 15 11 14 0 8 2 1 13 3 4 9 6	42	42, 44, 46, 48, 50, 52, 54, 56	56	5,205,455,489	2,233,122,771	4,501,868	496.04
14	7 6 8 1 11 5 14 10 3 4 9 13 15 2 0 12	41	41, 43, 45, 47, 49, 51, 53, 55, 57, 59	59				

88	15 2 12 11 14 13 9 5 1 3 8 7 0 10 6 4	43	43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 63, 65	65				

3. Notes and Discussion

As reported from the experimentation, the program generated too many nodes than expected and takes very long to solve. There are several blank entries because the puzzle takes too long in real time to solve. The time displayed in the program and the actual time that has elapsed differs immensely.

In order to speed up the algorithm in terms of the number of expanded nodes per second, the Last Moves Heuristic was implemented. The Last Moves Heuristic prevents the program from going back to the state it was last moved. For example, if the blank position turned left, this turn, it will not turn right as it would only revert to its original state, consuming extra time and memory. Even after implementation of the Last Moves Heuristic, the program still has many aspects that needs further optimization so that less time and memory is required to solve the puzzle.

All in all, the program was optimized to expand approximately 4.5 million nodes per second at best.