Assignment 1: Search Space Representation

# COS 314 Artificial Intelligence

## March 2019

Contents

[COS 314 Artificial Intelligence 1](#_Toc3810972)

[March 2019 1](#_Toc3810973)

[Statement of Purpose 3](#_Toc3810974)

[Research Questions 3](#_Toc3810975)

[Research Design 3](#_Toc3810976)

[Research Findings 4](#_Toc3810977)

[Discussion of Findings 6](#_Toc3810978)

[Conclusion 8](#_Toc3810979)

### Statement of Purpose

This report is designed to compare the performance of the breadth first, hill-climbing and the A\* algorithm in solving instances of the 8-puzzle problem of differing difficulty.

### Research Questions

This report examines two major questions regarding search algorithms when applying them to the 8-puzzle problem, the 8-Puzzle problem involves moving tiles on a 3 x 3 board to get from an initial state to a goal state. The board contains 8 numbered tiles and a space. This problem is useful because there are levels of difficulty that can be testes and it is a simple puzzle to solve which will allow even a computer with a low processing power to solve without waiting relatively long for it to be solved. The algorithms that will be investigated are Breadth first, Hill-climbing and the A\* algorithm. The following factors will be investigated:

1. The speed at which the algorithm solves the puzzle, this will be measured in seconds.
2. The number of moves that the algorithm returns when solving the puzzle
3. Difference between the number of moves and the known optimum

### Research Design

The run time environment used for the experiment is as follows:

1. Windows 10 64-bit OS
2. Intel i5 7th generation core
3. 20GB of DDR4 RAM

The following restrictions fill be imposed to keep fairness:

1. Python will be used for each algorithm
2. The same algorithm for generation of children will be used
3. The same problems will be used, in the same format

The following Heuristics were used for the respective functions:

1. Breadth First: None
2. Hill-Climbing: Manhattan distance
3. A\* Algorithm: Manhattan distance + current cost

The Manhattan distance is the sum of the distances of each tile from its position in the goal state. The following matrix was created to assist with the calculation as to reduce processing time:

moveCount = [

[0, 1, 2, 1, 2, 3, 2, 3, 4], # 0

[1, 0, 1, 2, 1, 2, 3, 2, 3], # 1

[2, 1, 0, 3, 2, 1, 4, 3, 2], # 2

[1, 2, 3, 0, 1, 2, 1, 2, 3], # 3

[2, 1, 2, 1, 0, 1, 2, 1, 2], # 4

[3, 2, 1, 2, 1, 0, 3, 2, 1], # 5

[2, 3, 4, 1, 2, 3, 0, 1, 2], # 6

[3, 2, 3, 2, 1, 2, 1, 0, 1], # 7

[4, 3, 2, 3, 2, 1, 2, 1, 0] # 8]

The matrix will take the current index of the tile and the index of the tile in the goal state and use the value that returns as the distance. The Manhattan distance is also an admissible heuristic which makes the A algorithm the A\* algorithm.

The following puzzles will be used:

Table 1: 8-Puzzle problem instances

|  |  |  |  |
| --- | --- | --- | --- |
|  | Initial state | Goal state | Known  Optimum (minimum  moves) |
| 1 | 123804765 | 134862705 | 5 |
| 2 | 123804765 | 281043765 | 9 |
| 3 | 123804765 | 281463075 | 12 |
| 4 | 134805726 | 123804765 | 6 |
| 5 | 231708654 | 123804765 | 14 |
| 6 | 231804765 | 123804765 | 16 |
| 7 | 123804765 | 231804765 | 16 |
| 8 | 283104765 | 123804765 | 4 |
| 9 | 876105234 | 123804765 | 28 |
| 10 | 867254301 | 123456780 | 31 |
| 11 | 647850321 | 123456780 | 31 |
| 12 | 123804765 | 567408321 | 30 |
| 13 | 806547231 | 012345678 | 30 |
| 14 | 641302758 | 012345678 | 14 |
| 15 | 158327064 | 012345678 | 12 |
| 16 | 328451670 | 012345678 | 12 |
| 17 | 035428617 | 012345678 | 10 |
| 18 | 725310648 | 012345678 | 15 |
| 19 | 412087635 | 123456780 | 17 |
| 20 | 162573048 | 123456780 | 10 |

### Research Findings

The following results were obtained:

Table 2: Breadth First Results

|  |  |  |  |
| --- | --- | --- | --- |
|  | Number of moves | Difference from known optimum | Time elapsed |
| 1 | 5 | 0 | 0.015621662139892578 |
| 2 | 9 | 0 | 0.0 |
| 3 | 12 | 0 | 0.1405940055847168 |
| 4 | 6 | 0 | 0.0 |
| 5 | 14 | 0 | 1.4996459484100342 |
| 6 | 16 | 0 | 5.09255838394165 |
| 7 | 16 | 0 | 6.0766942501068115 |
| 8 | 4 | 0 | 0.0 |
| 9 | 28 | 0 | 1682.5457541942596 |
| 10 | 31 | 0 | 1786.0286140441895 |
| 11 | 31 | 0 | 1730.0015070438385 |
| 12 | 30 | 0 | 1743.403317451477 |
| 13 | 31 | 1 | 1737.8685419559479 |
| 14 | 14 | 0 | 1.2973709106445312 |
| 15 | 12 | 0 | 0.1965799331665039 |
| 16 | 12 | 0 | 0.18082594871520996 |
| 17 | 10 | 0 | 0.01995992660522461 |
| 18 | 15 | 0 | 2.205942392349243 |
| 19 | 17 | 0 | 19.91004490852356 |
| 20 | 10 | 0 | 0.024100303649902344 |

Table 3: Hill climbing Results

|  |  |  |  |
| --- | --- | --- | --- |
|  | Number of moves | Difference from known optimum | Time elapsed |
| 1 | 113 | 108 | 0.004996061325073242 |
| 2 | 79 | 70 | 0.0 |
| 3 | 134 | 122 | 0.0099945068359375 |
| 4 | 6 | 0 | 0.0 |
| 5 | 464 | 450 | 0.05408906936645508 |
| 6 | 746 | 730 | 0.13489770889282227 |
| 7 | 1022 | 1006 | 0.26377010345458984 |
| 8 | 4 | 0 | 0.0 |
| 9 | 358 | 330 | 0.03498125076293945 |
| 10 | 153 | 122 | 0.004999876022338867 |
| 11 | 49 | 18 | 0.004995822906494141 |
| 12 | 152 | 122 | 0.004997730255126953 |
| 13 | 89 | 59 | 0.004995822906494141 |
| 14 | 58 | 44 | 0.0 |
| 15 | 450 | 438 | 0.052347660064697266 |
| 16 | 204 | 192 | 0.014990806579589844 |
| 17 | 10 | 0 | 0.0 |
| 18 | 51 | 36 | 0.0 |
| 19 | 611 | 594 | 0.09351325035095215 |
| 20 | 206 | 196 | 0.014992952346801758 |

Table 4: A\* Results

|  |  |  |  |
| --- | --- | --- | --- |
|  | Number of moves | Difference from known optimum | Time elapsed |
| 1 | 5 | 0 | 0.0 |
| 2 | 9 | 0 | 0.0049974918365478516 |
| 3 | 12 | 0 | 0.0 |
| 4 | 6 | 0 | 0.0 |
| 5 | 14 | 0 | 0.004996776580810547 |
| 6 | 16 | 0 | 0.0049974918365478516 |
| 7 | 16 | 0 | 0.014992237091064453 |
| 8 | 4 | 0 | 0.0 |
| 9 | 30 | 2 | 5.846850872039795 |
| 10 | 31 | 0 | 2.81349778175354 |
| 11 | 31 | 0 | 2.653585195541382 |
| 12 | 30 | 0 | 0.0599675178527832 |
| 13 | 31 | 1 | 3.5331149101257324 |
| 14 | 14 | 0 | 0.00499725341796875 |
| 15 | 12 | 0 | 0.0 |
| 16 | 12 | 0 | 0.0 |
| 17 | 10 | 0 | 0.0 |
| 18 | 19 | 4 | 0.024985790252685547 |
| 19 | 17 | 0 | 0.004996538162231445 |
| 20 | 10 | 0 | 0.0 |

### Discussion of Findings

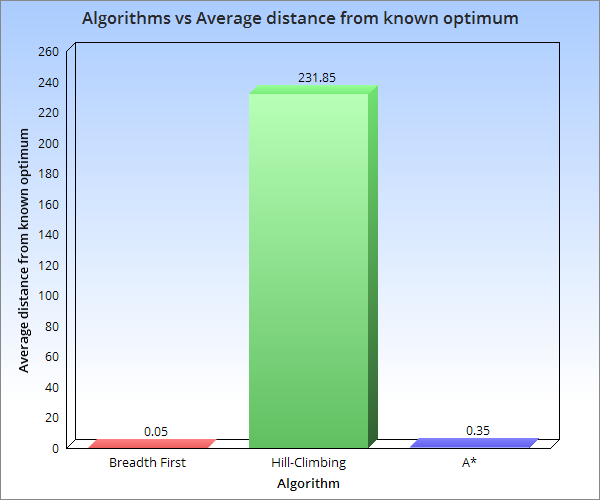


Figure 1: Average distance

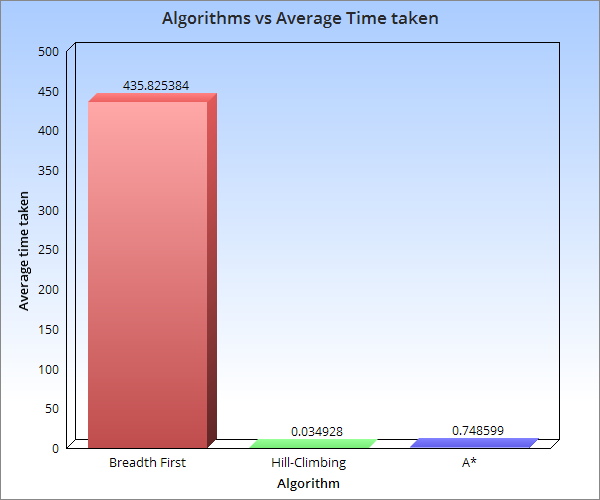


Figure 2: Average time

From the data retrieved Breadth first has the best average distance from known optimum with Hill-climbing being the worst. Hill-climbing is the fastest with breadth first being the slowest. With A\* not being relatively slow and returning a path relatively close to the optimal it is a better choice than the other two algorithms, it’s faster than breadth first and has a better solution than Hill-climbing.

### Conclusion

In conclusion the A\* algorithm is a more efficient algorithm, when an efficient heuristic such as the Manhattan distance is used with the cost taken so far.