Dustin Fox

Dr. Terry Soule

Analysis of algorithms

21 February 2017

Project 1

For this project, I created one solver class that holds all five of the required algorithms: BFS (no closed list), BFS (with closed list), DFS, A\* (with Manhattan distance), and A\* (with Straight Line Distance). In addition to creating the algorithms, I also did some research into the most efficient data structures to use for them. I sped up my searches quite a bit by using a c++ standard library unordered\_set for my closed list, which allowed for 0(1) when searching the closed list.

In order to test the efficiency of these searches I set up my program to allow the user to randomize the board as much as they want. Randomization can be done by hand or using a random move generator. Then once you have your preferred configuration you can have each algorithm solve the same board individually, that way I can get a 1:1 comparison between all of the algorithms. This also means that I did not stick to a fixed random number of randomizations; instead, I increased and decreased the number in order to get the full range of “moves per solution.” Even for fun, I manually entered one of the two existing 31 move configurations to see how each algorithm would handle it (see RAW DATA on page 6). In addition, in order to keep my computer from running out of memory; I limited all the algorithms to 3,000,000 nodes.

What I found was that the Worst algorithm was either the BFS without a closed list, or the DFS. The BFS without a closed list could not solve a board that had more than 13 moves without hitting the 3,000,000-node limit. The DFS could quickly compute a solution to any configuration, but it would take a ridiculous amount of moves as you can see in the data. The best overall was the A\* with Manhattan distance, which had a slight advantage over the Straight Line Distance when solving the maximum move configuration (31 moves).

**RAW DATA 1 of 2**

|  |  |  |
| --- | --- | --- |
|  | BFS (no closed) |  |
| Moves | Nodes | Time (seconds) |
| 3 | 25 | 0 |
| 5 | 275 | 0 |
| 8 | 2912 | 0.03 |
| 10 | 59646 | 0.71 |
| 13 | 467606 | 4.83 |
| 15 | 3000000 | 30 |
| 15 | 3000000 | 30 |
| 16 | 3000000 | 30 |
| 17 | 3000000 | 30 |
| 21 | 3000000 | 30 |
| 22 | 3000000 | 30 |
| 23 | 3000000 | 30 |
| 27 | 3000000 | 30 |
| 31 | 3000000 | 30 |
| AVG | AVG | AVG |
| 16.143 | 1966461.714 | 19.68357143 |

|  |  |  |
| --- | --- | --- |
|  | DFS |  |
| Moves | Nodes | Time (seconds) |
| 97633 | 144526 | 2 |
| 83067 | 166601 | 2.34 |
| 57822 | 192339 | 2.75 |
| 73620 | 175018 | 2.4 |
| 59611 | 60681 | 0.78 |
| 90275 | 156481 | 2.18 |
| 98517 | 142902 | 2 |
| 101510 | 134766 | 1.6875 |
| 78331 | 81452 | 1.4 |
| 101009 | 136474 | 1.7 |
| 86630 | 91893 | 1.3 |
| 76709 | 172487 | 2.17188 |
| 73331 | 176408 | 2.2 |
| 54151 | 54956 | 0.7 |
| AVG | AVG | AVG |
| 80872.6 | 134785 | 1.829241429 |

|  |  |  |
| --- | --- | --- |
|  | BFS (closed) |  |
| Moves | Nodes | Time (seconds) |
| 3 | 14 | 0 |
| 5 | 54 | 0 |
| 8 | 200 | 0 |
| 10 | 856 | 0.03 |
| 13 | 2533 | 0.125 |
| 15 | 7330 | 0.75 |
| 15 | 9719 | 0.82 |
| 16 | 10708 | 0.63 |
| 17 | 25833 | 1.17 |
| 21 | 111058 | 2.04 |
| 22 | 121898 | 2.32 |
| 23 | 236719 | 3.6 |
| 27 | 460149 | 6.62 |
| 31 | 519140 | 6.12 |
| AVG | AVG | AVG |
| 16.143 | 107586.5 | 1.730357143 |

**RAW DATA 2 of 2**

|  |  |  |
| --- | --- | --- |
|  | A\*1 |  |
| Moves | Nodes | Time (seconds) |
| 3 | 4 | 0 |
| 5 | 7 | 0.03 |
| 8 | 13 | 0.01 |
| 10 | 22 | 0.01 |
| 13 | 69 | 0.02 |
| 15 | 143 | 0.05 |
| 15 | 103 | 0.3 |
| 16 | 316 | 0.06 |
| 17 | 79 | 0 |
| 21 | 1054 | 0.04 |
| 22 | 569 | 0.05 |
| 23 | 2502 | 0.14 |
| 27 | 2155 | 0.09 |
| 31 | 16778 | 2.375 |
| AVG | AVG | AVG |
| 16.143 | 1701 | 0.226785714 |

|  |  |  |
| --- | --- | --- |
|  | A\*2 |  |
| Moves | Nodes | Time (seconds) |
| 3 | 4 | 0 |
| 5 | 7 | 0 |
| 8 | 13 | 0 |
| 10 | 22 | 0 |
| 13 | 100 | 0.03 |
| 15 | 167 | 0 |
| 15 | 168 | 0 |
| 16 | 398 | 0 |
| 17 | 183 | 0 |
| 21 | 1309 | 0.04 |
| 22 | 1358 | 0.06 |
| 23 | 4086 | 0.23 |
| 27 | 10877 | 1.53 |
| 31 | 41486 | 21 |
| AVG | AVG | AVG |
| 16.143 | 4298.4 | 1.635 |