**Artificial Intelligence**

**Project 1**

**Team 3**

Code at: https://github.com/ro1406/CMP333-Project1.git

**8-Puzzle (Default Board State):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm (+Heuristic)** | **Nodes Expanded** | **Nodes Generated** | **Solution Cost** | **Time Taken (sec)** |
| **DFS** | - | - | - | - |
| **BFS** | 1271285 | 3589819 | 14 | 11.88 |
| **IDA** | 1965733 | 1965720 | 14 | 3.31 |
| **Djikstra** | 1163507 | 3131259 | 14 | 21.15 |
| **Greedy (Manhattan)** | - | - | - | - |
| **Greedy (Hamming)** | - | - | - | - |
| **A\* (Manhattan)** | 906 | 2439 | 14 | 0.02 |
| **A\* (Hamming)** | 5640 | 16626 | 14 | 0.10 |

DFS: Not Possible (Infinite Search Tree)

Greedy Algorithm: The algorithm does not resolve since the heuristics are not very useful without the actual path cost. Moreover, since the search tree is infinite, the algorithm gets stuck in a loop, and does not terminate. We can observe the results of using the heuristic with the path cost in the A\* algorithm, that performs the best overall, better than Djikstra which uses only the path cost. Hence the heuristic is useful but not on its own.

**8-Puzzle (New Board State below):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm (+Heuristic)** | **Nodes Expanded** | **Nodes Generated** | **Solution Cost** | **Time Taken (sec)** |
| **DFS** | - | - | - | - |
| **BFS** | 288645 | 819651 | 12 | 1.6388 |
| **IDA** | 446963 | 446952 | 12 | 0.7711 |
| **Djikstra** | 275108 | 737463 | 12 | 2.9916 |
| **Greedy (Manhattan)** | - | - | - | - |
| **Greedy (Hamming)** | - | - | - | - |
| **A\* (Manhattan)** | 68 | 183 | 12 | 0.0020 |
| **A\* (Hamming)** | 2101 | 5969 | 12 | 0.0329 |

DFS: Not Possible (Infinite Search Tree)

Greedy Algorithm: The algorithm does not resolve since the heuristics are not very useful without the actual path cost. Moreover, since the search tree is infinite, the algorithm gets stuck in a loop, and does not terminate. We can observe the results of using the heuristic with the path cost in the A\* algorithm, that performs the best overall, better than Djikstra which uses only the path cost. Hence the heuristic is useful but not on its own.

New Board State:

[1,6,5,

4,0,2,

7,3,8]

**Pacman (Default Board)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm (+Heuristic)** | **Nodes Expanded** | **Nodes Generated** | **Solution Cost** | **Time Taken (sec)** |
| **DFS** | - | - | - | - |
| **BFS** | 16707 | 36555 | 14 | 0.1139 |
| **IDA** | 30677 | 30665 | 14 | 0.0567 |
| **Djikstra** | 16723 | 36711 | 14 | 0.1944 |
| **Greedy (Euclidean)** | - | - | - | - |
| **A\* (Euclidean)** | 221 | 479 | 14 | 0.0020 |

DFS: Not Possible (Infinite Search Tree)

Greedy Algorithm: The algorithm does not resolve since the heuristics are not very useful without the actual path cost. Moreover, since the search tree is infinite, the algorithm gets stuck in a loop, and does not terminate. We can observe the results of using the heuristic with the path cost in the A\* algorithm, that performs the best overall, better than Djikstra which uses only the path cost. Hence the heuristic is useful but not on its own.

**Pacman (Custom Board Below)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm (+Heuristic)** | **Nodes Expanded** | **Nodes Generated** | **Solution Cost** | **Time Taken (sec)** |
| **DFS** | - | - | - | - |
| **BFS** | 145830 | 361141 | 16 | 1.4562 |
| **IDA** | 249699 | 249685 | 16 | 0.4206 |
| **Djikstra** | 197609 | 497291 | 16 | 3.5185 |
| **Greedy (Euclidean)** | - | - | - | - |
| **A\* (Euclidean)** | 181 | 411 | 16 | 0.0030 |

DFS: Not Possible (Infinite Search Tree)

Greedy Algorithm: The algorithm does not resolve since the heuristics are not very useful without the actual path cost. Moreover, since the search tree is infinite, the algorithm gets stuck in a loop, and does not terminate. We can observe the results of using the heuristic with the path cost in the A\* algorithm, that performs the best overall, better than Djikstra which uses only the path cost. Hence the heuristic is useful but not on its own.

Custom Board:

[ "P--------%",

"%-%%%-%%-%",

"%---%----%",

"%-%-%%%-%%",

"%-%------%",

"%----%%%--",

"%----%---."]

**Farmer Puzzle**

Initial state always set to [False,False,False,False], with goal state always [True,True,True,True].

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm (+Heuristic)** | **Nodes Expanded** | **Nodes Generated** | **Solution Cost** | **Time Taken (sec)** |
| **DFS** | - | - | - | - |
| **BFS** | 97 | 210 | 8 | 0.000389337540 |
| **IDA** | 186 | 180 | 8 | 0.000000000000 |
| **Djikstra** | 157 | 342 | 8 | 0.000988245010 |
| **Greedy (Hamming)** | 11 | 22 | 8 | 0.000000000000 |
| **A\* (Hamming)** | 36 | 74 | 8 | 0.000000000000 |

DFS: Not Possible (Infinite Search Tree)

Clearly, Greedy search performs the best, with the least number of nodes expanded (11), followed by A\* with 36 nodes expanded. These two algorithms outperform the others because of the use of the heuristic provided. Indicating that the heuristic is very good. In fact, we can conclude that the heuristic is more useful than the uniform cost assigned to each action, since Greedy algorithm (that only uses heuristics) outperforms the rest. Although, we do notice that IDA is very quick (as fast as Greedy and A\*), it expands the greatest number of nodes, which means the search itself is not very efficient.

**Stone Puzzle**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm (+Heuristic)** | **Nodes Expanded** | **Nodes Generated** | **Solution Cost** | **Time Taken (sec)** |
| **DFS** | 14 | 17 | 8 | 0.0 |
| **BFS** | 30 | 31 | 8 | 0.0 |
| **IDA** | 135 | 128 | 8 | 0.000506401062 |
| **Djikstra** | 30 | 31 | 8 | 0.0 |
| **Greedy (Hamming)** | 15 | 17 | 8 | 0.0 |
| **A\* (Hamming)** | 29 | 30 | 8 | 0.000995159149 |

For this puzzle, the DFS was the most efficient search, generating and expanding the least nodes, and taking a negligible time to reach the goal state. The BFS, Uniform and Greedy search also took a negligible amount of time, but the BFS expanded and generated nearly twice the amount of nodes that the DFS did. The slowest search was the A\* search, but the least efficient search (and second slowest) was the IDA. Generating a staggering 128 nodes and expanding 135, it is by far a beyond inefficient search for such a simple puzzle. In the cases of heuristics, the heuristics were good for the greedy search, as it produced almost identical results to the DFS, only expanding one extra node. Hence, the heuristic of the Hamming distance performs decently since the Greedy algorithm (using only the heuristic) performed well in this particular case. Overall, all algorithms were able to find the optimal solution.

Initial board: [O,O,\_,X,X]

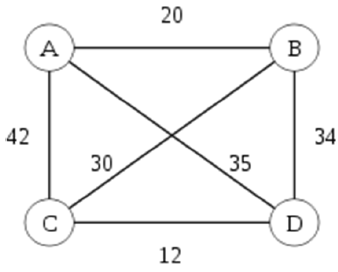
Final board: [X,X,\_,O,O]

**Travelling Salesman**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm (+Heuristic)** | **Nodes Expanded** | **Nodes Generated** | **Solution Cost** | **Time Taken (sec)** |
| **DFS** | 4 | 7 | 77 | 0.0 |
| **BFS** | 11 | 16 | 62 | 0.0 |
| **IDA** | 18 | 16 | 62 | 0.0 |
| **Djikstra** | 9 | 14 | 62 | 0.0 |
| **Greedy (Closest Neighbour)** | 4 | 7 | 62 | 0.0 |
| **A\* (Closest Neighbour)** | 6 | 10 | 62 | 0.0 |

In the case of the TSP problem, the DFS and Greedy search perform almost identically, however, the greedy search did take a different path to the solution, leading to a lower solution cost of 62 (vs 77). This could be indicative of a admissible and useful heuristic to aid the search, as the A\* search also expanded 6 nodes and generated 10 nodes, which comparatively less than the remaining searches. The IDA was the least efficient search, expanding 18 nodes and generating 16. Generally, all algorithms found the optimal solution (apart from DFS with cost 77), and the time taken is negligible for all.

Graph:



**SOKOBAN RESULTS**

**EASY:**

01\_easy.txt:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm (+Heuristic)** | **Nodes Expanded** | **Nodes Generated** | **Solution Cost** | **Time Taken (sec)** |
| **BFS** | - | - | - | - |
| **DFS** | - | - | - | - |
| **IDA** | - | - | - | - |
| **Djikstra** | - | - | - | - |
| **Greedy (Manhattan)** | 51303 | 64206 | 124 | 24.88 |
| **Greedy (Euclidean)** | 69719 | 79984 | 110 | 33.10 |
| **A\* (Manhattan)** | 9599 | 18008 | 24 | 4.03 |
| **A\* (Euclidean)** | 19897 | 36836 | 24 | 8.23 |

The other algorithms took too long to run, hence we omitted them from the results. With A\* finding the optimal solution

02\_easy.txt:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm (+Heuristic)** | **Nodes Expanded** | **Nodes Generated** | **Solution Cost** | **Time Taken (sec)** |
| **BFS** | 446 | 477 | 19 | 0.055849552155 |
| **DFS** | 303 | 330 | 53 | 0.043915987015 |
| **IDA** | 4247 | 4217 | 31 | 0.473831653595 |
| **Djikstra** | 473 | 500 | 19 | 0.062829732895 |
| **Greedy (Manhattan)** | 113 | 177 | 19 | 0.015926122665 |
| **Greedy (Euclidean)** | 96 | 150 | 19 | 0.014960289001 |
| **A\* (Manhattan)** | 277 | 324 | 19 | 0.035903453827 |
| **A\* (Euclidean)** | 317 | 350 | 19 | 0.040890932083 |

03\_easy.txt:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm (+Heuristic)** | **Nodes Expanded** | **Nodes Generated** | **Solution Cost** | **Time Taken (sec)** |
| **BFS** | 32652 | 37393 | 21 | 5.886620521545 |
| **DFS** | 6273 | 6470 | 285 | 3.383734703064 |
| **IDA** | 52560 | 52527 | 33 | 6.662277936935 |
| **Djikstra** | 33748 | 38393 | 21 | 6.719211578369 |
| **Greedy (Manhattan)** | 375 | 574 | 63 | 0.090455532074 |
| **Greedy (Euclidean)** | 653 | 1064 | 55 | 0.145571708679 |
| **A\* (Manhattan)** | 1055 | 1713 | 25 | 0.501669645309 |
| **A\* (Euclidean)** | 2491 | 3753 | 25 | 0.496952295303 |

**MODERATE**:

04\_moderate.txt:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm (+Heuristic)** | **Nodes Expanded** | **Nodes Generated** | **Solution Cost** | **Time Taken (sec)** |
| **BFS** | 5066 | 5680 | 23 | 0.649682760239 |
| **DFS** | 2697 | 2738 | 79 | 0.529447317123 |
| **IDA** | 31429 | 31383 | 47 | 3.815632104874 |
| **Djikstra** | 4773 | 5460 | 23 | 0.638025522232 |
| **Greedy (Manhattan)** | 386 | 521 | 27 | 0.055817842484 |
| **Greedy (Euclidean)** | 355 | 472 | 23 | 0.046904563904 |
| **A\* (Manhattan)** | 1337 | 1777 | 23 | 0.205453395844 |
| **A\* (Euclidean)** | 1712 | 2249 | 23 | 0.231512308121 |

05\_moderate.txt:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm (+Heuristic)** | **Nodes Expanded** | **Nodes Generated** | **Solution Cost** | **Time Taken (sec)** |
| **BFS** | 27260 | 30672 | 31 | 4.456086158752 |
| **DFS** | 4107 | 4188 | 139 | 1.341627836227 |
| **IDA** | 15167 | 15129 | 39 | 2.126607894897 |
| **Djikstra** | 28371 | 31727 | 31 | 5.464051723480 |
| **Greedy (Manhattan)** | 269 | 412 | 35 | 0.041227102280 |
| **Greedy (Euclidean)** | 444 | 586 | 35 | 0.068816184998 |
| **A\* (Manhattan)** | 3125 | 3871 | 33 | 0.514642477036 |
| **A\* (Euclidean)** | 3120 | 3726 | 31 | 0.509682178497 |

06\_moderate.txt:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm (+Heuristic)** | **Nodes Expanded** | **Nodes Generated** | **Solution Cost** | **Time Taken (sec)** |
| **BFS** | 7417 | 7529 | 27 | 1.856709241867 |
| **DFS** | 2198 | 2241 | 165 | 0.592744827271 |
| **IDA** | 361571 | 361413 | 159 | 90.856912136078 |
| **Djikstra** | 7436 | 7563 | 97 | 1.708682298660 |
| **Greedy (Manhattan)** | 2971 | 3213 | 97 | 0.722657918930 |
| **Greedy (Euclidean)** | 3058 | 3264 | 111 | 0.789869070053 |
| **A\* (Manhattan)** | 5556 | 5744 | 99 | 1.406153678894 |
| **A\* (Euclidean)** | 5559 | 5749 | 97 | 1.377694606781 |

07\_moderate.txt:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm (+Heuristic)** | **Nodes Expanded** | **Nodes Generated** | **Solution Cost** | **Time Taken (sec)** |
| **BFS** | 775 | 898 | 17 | 0.088764429092 |
| **DFS** | 1010 | 1023 | 37 | 0.164557695389 |
| **IDA** | 1986 | 1970 | 17 | 0.210469245911 |
| **Djikstra** | 787 | 913 | 17 | 0.094267368317 |
| **Greedy (Manhattan)** | 62 | 97 | 17 | 0.015957355499 |
| **Greedy (Euclidean)** | 70 | 113 | 19 | 0.010936975479 |
| **A\* (Manhattan)** | 334 | 434 | 17 | 0.044881105423 |
| **A\* (Euclidean)** | 390 | 501 | 17 | 0.050864219666 |

Overall, we can observe that Greedy using Manhattan distance usually takes the least number of nodes expanded on average, occasionally finding optimal or suboptimal solutions. Hence we can conclude that the heuristics used are not necessarily admissible, however, they perform very well, given that it find a solution with the least number of nodes explored.

However, A\* did not perform as expected, which may be attributed to having a non-admissible heuristic. However, it did find the optimal solution in some cases.

Overall, the best algorithm was BFS, exploring a fairly low number of nodes compared to IDA, Djikstra,etc, and always finding the optimal solution.