Artificial Intelligence

Assignment 1
Coding Problem

Problem Statement :

1. Implement A* and IDA* search to solve the 8-puzzle problem. One of the heuristics you need to implement is taking the approximation function $h_a(n) = 0$ which leads to a breadth first search. You need to think of three more heuristics and code them. Take a random set of inputs and also compare the performance of the heuristics. Performance measures will be number of nodes expanded and the time taken for complete execution of each of the heuristics.

Deliverables :-
Submit a zip or tar file containing the following.The names of zip/tar submitted should be <Roll No>_assignment1.<ext>. All the files inside the zip/tar should also follow the same naming. For eg:- 13CS30031_assignment1.tar.gz, 13CS30031_assignment1.cpp.

1. **Code (C++, Java or Python).**
2. **Report in .pdf mentioning the heuristics used and performance comparisons.**
3. **A .txt file containing 50 random instances and whether a solution was found or not. If the solution exists, you also need to print out the number of nodes expanded and time taken for execution for each of the 4 heuristics.**

Heuristic costs selection

The following heuristics were selected for both A* and IDA* search :

1. **H=0, here the heuristic cost of each node was set to 0. So basically, it is BFS in case of A*.**
2. **H = Sum of Hamming distances of current configuration and the goal configuration for all the tiles.**
3. **H = Sum of manhattan distances of current configuration and the goal configuration for all the tiles.**
4. **H = Sum of euclidean distance of current configuration and the goal configuration for all the tiles.**

Here, for both A* and IDA* a visited set was created which stores the nodes already visited so that there is no loop in the search. IDA* prunes the branches in the search graph which is finds to be unlikely to reach to goal state as its f= g+h is higher than the bound, when no goal state is reached and still stack becomes empty, it is time to increase the bound for IDA*

Since H=0 is just like BFS it took the maximum amount of time in all the simulations (50) for both A* and IDA* search. With maximum time going to about 6 seconds for A* search and around 100 seconds for IDA* search. It also expanded maximum number of nodes for both A* and IDA* searches. With 2 lac nodes expanded in A* search.

Hamming distance is not admissible but still performs better than BFS for both A* and IDA* searches with average search time less than a second in A* and max search time in case of IDA* reaching till 10 seconds. The nodes expanded are also less as compared to h=0 search both in case of A* and IDA* searches

Euclidean distance performs better than hamming distance but not better than manhattan distance in cases of nodes expanded, time taken and the path to reach the goal configuration. The time takes is in order of less than a second both in case of A* and IDA* searhes

Manhattan distance is admissible and hence provides optimal path with least number of nodes expanded and time taken both in case of A* and IDA* searches. It provides the least number of moves to get to the goal configuration for all the start states.

So, performance wise :
Manhattan distance > euclidean distance > hamming distance > h=0 (BFS)

Time taken (Maximum)/ Nodes Expanded	A*	IDA*
H=0	5.6 / 170K	600 / 33K
Hamming	1.3 / 40K	10.5 / 26K
Euclidean	0.3 / 10	1.4 / 6K
Manhattan	0.1 / 3K	0.2 / 3K

Report by :
Apurv Kumar
14CS10006