Programming Assignment-1

Name: Sanya Garg

ID : 2018A7PS0261G

To solve the 15 puzzle problem I used A* algorithm and the heuristic function is calculated using Manhattan distance and linear row collision.

Manhattan distance: It is the horizontal and vertical distance of the current state from the goal state.

Linear row and column collision: If all the numbers are present in the correct row or correct column, but order is incorrect then we add the misplaced value in the heuristic function

So we calculate the value of f(n) = heuristic function + depth

Depth is the no of moves made to reach the current state from the initial state. A heap is used to store the value of f(n), so that every time we pop out the minimum value of f(n).

Heap: contains tuple of f(n),the current state, depth, the actions.

Algorithm:

For each move (UP/DOWN/ RIGHT/LEFT) , a child is generated. The f(n) is calculated for the new state and the min f(n) ie. is the path cost is further branched to reach the goal state.

A* Algorithm gives an optimal solution, if heuristic function is admissible.

Heuristic function is calculated using Manhattan distance and linear row collision. The reason this heuristic works is that it never over estimates the solution (optimal cost).

We observe that f(n) is less than or equal to the value of C*(optimal cost).

Reason for using linear row collision:

However for the inputs given we observe that the value of f(n) is same for many children, this is because the at every move the block was moving closer to the goal by 1 step in at least 2 children.

Hence the effective branching factor is very high.

In order to reduce this branching and reduce the time taken in linear row and column collision.

Initial State	Nodes Generated	Time taken
1	31	0.001s
2	16204	1.1559 s
3	90552	6.6921 s
4	173351	16.2501s

NOTE : I have imported copy library.