Assignment AIR-1

Title: 8-Puzzle problem using A* algorithm

Problem Statement
Solve 8-puzzle problem using A* algorithm. Assume any initial configuration and define goal configuration clearly.

Objective
To learn and understand use and need of A* algorithm
To apply A* algorithm to real time problem
To implement A* algorithm using suitable programming language.

Outcome
To be able to learn about A* algorithm
To be able to apply A* algorithm to gaming problem
To be able to implement A* algorithm using suitable programming language like prolog/python/java

Software and hardware requirements:
Fedora 20, 4GB RAM, 500 GB HDD, Java jdk, Python libraries

Theory related concepts
- A* algorithm is a heuristic search algorithm for finding paths in a graph.
- Consider a square grid having many obstacles and we are given a starting cell and a target cell.
- We want to search target cell from the starting cell as quickly as possible.

- At each step A* algorithm picks the node according to the value 'f' which is equal to the sum of 'g' and 'h'
- At each step, it picks the node cell having least 'f' and process that node

$$f = g + h$$

where g = movement cost to movement from the starting point to a given grid following the path generated to get there

h = movement cost (estimated) to move from that given square on the grid to the final heuristic which is nothing but a kind of smart guess

Algorithm

1. Initialize the open list
2. Initialize the closed list
   put the starting node on the open list
3. While the open list is not empty
   i) Find the node with the left 'f' in the open list called it 'q'
   ii) Pop 'q' off the open list
   iii) Generate 'q's successor
4. For each successor
   i) if successor is the goal, stop search successor
      g = q.g + distance (successor.g)
      successor.h = distance from goal to successor
      successor.f = successor.g + successor.h
   ii) If a node with the same position as successor in the open list which has a lower 'f' than successor, skips this successor
   iii) If a node with the same position as successor is in the closed

list which has a lower 'f' than successor, skip this successor
otherwise, add the node to open list
5. End for
6. Push q on the closed list
7. End while

Test Cases

Initial configuration	Final configuration
1 2 X	1 2 3
4 5 3	4 5 6
7 8 6	7 8 X

Output

1	2	3
4	5	X
7	8	6

→

1	2	3
4	5	6
7	8	X

The puzzle was solved in 18 moves

Conclusion
Thus we successfully implemented A* algorithm for 8-puzzle problem