# ASSIGNMENT AIR-1

**Roll No:** 41205

**Problem Statement:**

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

**Objective:**

1. To under the A\* algorithm
2. Apply A\* algorithm for the 8-puzzle problem

**Outcome:** One will be able to write the A\* algorithm to solve the 8-puzzle problem.

**Pre-requisites:**

1. 64-bit Linux OS
2. Programming Languages: Python

**Hardware Specification:**

1. x86\_64 bit
2. 2/4 GB DDR RAM
3. 80 - 500 GB SATA HD
4. 1GB NIDIA TITAN X Graphics Card

**Software Specification:**

1. Ubuntu 14.04

# Theory:

* In computer science, A\* (pronounced as "A star") is a computer algorithm that is widely used in path finding and graph traversal, the process of plotting an efficiently traversable path between multiple points, called nodes. The A\* algorithm combines features of uniform-cost search and pure heuristic search to efficiently compute optimal solutions.
* A\* algorithm is a best-first search algorithm in which the cost associated with a node is f(n) = g(n) + h(n), where g(n) is the cost of the path from the initial state to node n and h(n) is the heuristic estimate or the cost or a path from node n to a goal.
* Thus, f(n) estimates the lowest total cost of any solution path going through node n. At each point a node with lowest f value is chosen for expansion. Ties among nodes of equal f value should be broken in favor of nodes with lower h values. The algorithm terminates when a goal is chosen for expansion.
* For Puzzle, A\* algorithm, using these evaluation functions, can find optimal solutions to these problems. In addition, A\* makes the most efficient use of the given heuristic function in the following sense: among all shortest-path algorithms using the given heuristic function h(n). A\* algorithm expands the fewest number of nodes.
* The main drawback of A\* algorithm and indeed of any best-first search is its memory requirement. Since at least the entire open list must be saved, A\* algorithm is severely space- limited in practice and is no more practical than best-first search algorithm on current machines. For example, while it can be run successfully on the eight puzzles, it exhausts available memory in a matter of minutes on the fifteen puzzles.
* A star algorithm is very good search method, but with complexity problems
* To implement such a graph-search procedure, we will need to use two lists of node:
* \_OPEN: nodes that have been generated and have had the heuristic function applied to them but which have not yet been examined (i.e., had their successors generated). OPEN is actually a priority queue in which the elements with the highest priority are those with the most promising value of the heuristic function.
* \_CLOSED: Nodes that have already been examined. We need to keep these nodes in memory if we want to search a graph rather than a tree, since whether a node is generated; we need to check whether it has been generated before.

**Algorithm:**

1. Initialize the open list
2. Initialize the closed list
3. While the open list is not empty
   1. Find the node with the least f on the open list, call it "q"
   2. Pop q off the open list
   3. Generate q's 8 successors and set their parents to q
   4. For each successor
      1. If successor is the goal, stop search
      2. If a node with the same position as successor is in the OPEN list which has a lower f than successor, skip this successor
      3. Ff a node with the same position as successor is in the CLOSED list which has a lower f than successor, skip this successor
      4. Otherwise, add the node to the open list
   5. End (for loop)
4. Push q on the closed list
5. End(while loop)

**Test Cases:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Input | Expected Output | Actual Output | Result |
| 1 | Start State:  [[7 1 5]  [0 8 2]  [4 3 6]]  End State:  [[1 5 2]  [7 0 8]  [4 3 6]] | Solved in 5 steps | Solved in 5 steps | Success |
| 2 | Start State:  [[0 5 6]  [1 8 2]  [3 4 7]]  End State:  [[1 5 6]  [3 2 7]  [4 8 0]] | Solved in 6 steps | Solved in 6 steps | Success |

**Output:**

Initial and final states

Text

Description automatically generated

Steps to solve

A picture containing text, electronics

Description automatically generated

**Conclusion:** We have implemented the A\* algorithm to solve the 8-puzzle problem.