**ARTIFICIAL INTELLIGENCE**

**EXPERIMENT NO 1 - TOY PROBLEM**

**Q1. What is a toy problem**

**Ans:**  It is a concise and exact description of the problem which is used by the researchers to compare the performance of algorithms. A toy problem is useful to test and demonstrate methodologies.In the field of artificial intelligence, classical puzzles, games and problems are often used as toy problems. These include sliding-block puzzles, N-Queens problem, missionaries and cannibals problem, tic-tac-toe, chess,Hanoi tower and others.

**Name of Toy Problem - 8 Puzzle Problem**

**Q2. How many approaches do you have for solving the toy problem which you have taken?**

Ans. Here I'll show an BFS approach to solve the discussed problem. We can also use DFS to solve this toy problem.

It is the best one from other techniques. It is used to solve very complex problems

We also know the eight puzzle problem by the name of N puzzle problem or sliding puzzle problem.

N-puzzle that consists of N tiles (N+1 titles with an empty tile) where N can be 8, 15, 24 and so on.

In our example N = 8. (that is square root of (8+1) = 3 rows and 3 columns).

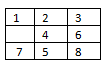
In the same way, if we have N = 15, 24 in this way, then they have rows and columns as follow (square root of (N+1) rows and square root of (N+1) columns).

That is if N=15 than number of rows and columns= 4, and if N= 24 number of rows and columns= 5.

So, basically in these types of problems we have given a initial state or initial configuration (Start state) and a Goal state or Goal Configuration.

Here We are solving a problem of 8 puzzle that is a 3x3 matrix.

**Initial state Goal state**

****

### **Solution:**

The puzzle can be solved by moving the tiles one by one in the single empty space and thus achieving the Goal state.

**Rules of solving puzzle**

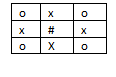
Instead of moving the tiles in the empty space we can visualize moving the empty space in place of the tile.

The empty space can only **move in four directions** (Movement of empty space)

1. Up
2. Down
3. Right or
4. Left

The empty space **cannot move diagonally** and can take **only one step at a time**.

**All possible move of a Empty tile**

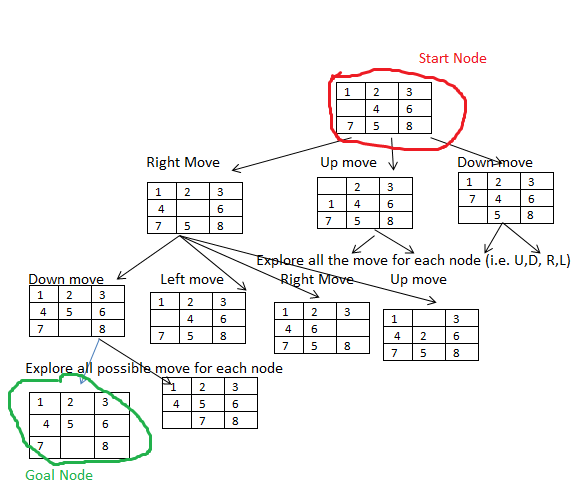
****

**o- Position** total possible moves are **(2)**, **x - position** total possible moves are **(3)** and

**#-position** total possible moves **are (4)**

Let's solve the problem without **Heuristic Search** that is **Uninformed Search or Blind Search (** [**Breadth First Search**](https://www.goeduhub.com/6137/implementation-bfs-for-problem-using-lisp-prolog-java-python) **and** [**Depth First Search**](https://www.goeduhub.com/6687/implementation-dfs-for-water-jug-problem-using-lisp-prolog)**)**

**Breath First Search to solve Eight puzzle problem**

****

**Note:** If we solve this problem with depth first search, then it will go to depth instead of exploring layer wise nodes.

**Time complexity:** In worst case time complexity in **BFS is O(b^d) know as order of b raise to power d.** In this **particular case it is (3^20).**

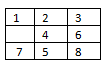
**b-**branch factor

**d-**depth factor

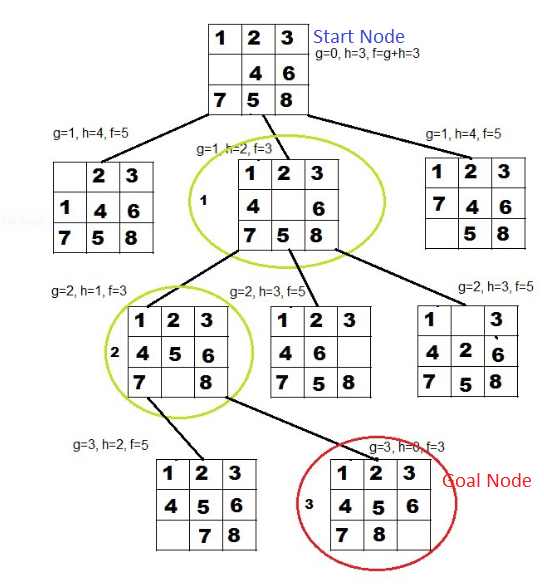
Let's solve the problem with **Heuristic Search** that is **Informed Search (**[**A\***](https://www.goeduhub.com/5072/implement-a-search-algorithm-part-1) **,** [**Best First Search (Greedy Search)**](https://www.goeduhub.com/7451/write-prolog-program-first-search-applied-eight-puzzle-problem)**)**

To solve the problem with Heuristic search or informed search we have to calculate Heuristic values of each node to calculate **cost function. (f=g+h)**

**Initial state Goal state**

** **

**Note:** See the initial state and goal state carefully all values except (4,5 and 8) are at their respective places. **so, the heuristic value for first node is 3.**(Three values are misplaced to reach the goal). And l**et's take actual cost (g) according to depth.**

****