Experiment No. 3

Title: Implementation of Best First Search to solve 8- puzzle problem.

Aim:
To implement the Best First Search algorithm to solve the 8 Puzzle Problem efficiently. The algorithm will utilize the count of misplaced tiles as a heuristic to guide the search process towards finding the optimal solution.

Objectives:

1. Develop a program that efficiently implements the Best First Search algorithm for solving the 8 Puzzle Problem.
2. Utilize the count of misplaced tiles as a heuristic function to guide the search process towards finding the optimal solution while minimizing computational resources.

Problem Statement:

The 8 Puzzle Problem involves a 3x3 grid with eight numbered tiles and one empty space. The objective is to rearrange the tiles into a target configuration by sliding them into the empty space. The challenge lies in finding the shortest sequence of moves to reach the target configuration from a given initial state. To solve this problem, we will employ the Best First Search algorithm, which explores the search space based on a heuristic function that estimates the cost to reach the goal state.

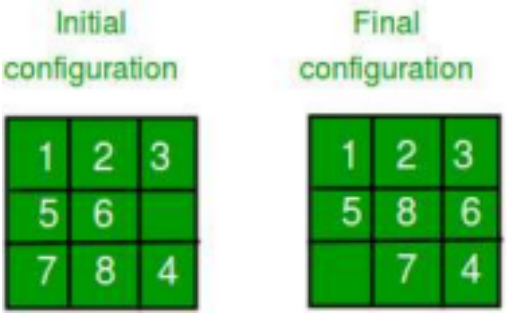
Software/s:

Python (version 3.x)

Theory:

8-Puzzle Problem:
Given a 3×3 board with 8 tiles (every tile has one number from 1 to 8) and one empty space. The objective is to place the numbers on tiles to match the final configuration using the empty space. We can slide four adjacent (left, right, above, and below) tiles into the empty space.

For example,



Best First Search (BFS):

Best First Search is a heuristic search algorithm that explores the search space by selecting the most promising node to expand next, based on a heuristic evaluation function. In the context of the 8 Puzzle Problem, the heuristic function estimates the cost to reach the goal state from a given state. One commonly used heuristic for this problem is the count of misplaced tiles, which calculates the number of tiles that are not in their correct position relative to the goal state.

Algorithm:

The steps involved in implementing Best First Search for solving the 8 Puzzle Problem are as follows:

1. Initialize: Start with the initial state of the puzzle and create an initial node representing this state.

2. Expansion: Expand the current node by generating all possible successor states resulting from valid moves (e.g., moving a tile into the empty space).

3. Evaluation: Evaluate each successor state using the heuristic function (count of misplaced tiles) to estimate the cost to reach the goal state from that state.

4. Selection: Select the successor state with the lowest heuristic value as

the next node to expand. This step prioritizes states that are closer to the goal state.

5. Repeat: Repeat steps 2-4 until the goal state is reached or no more nodes are left to explore.

By following these steps, the Best First Search algorithm navigates through the search space efficiently, guided by the count of misplaced tiles heuristic, to find the optimal solution to the 8 Puzzle Problem.

Pseudo Code:

```
function BestFirstSearch(initialState):
 priorityQueue = Priority Queue
 priorityQueue.enqueue(initialState,
heuristic(initialState))
 while priorityQueue is not empty:
 currentState = priorityQueue.dequeue()

 if currentState is goal state:
 return constructSolutionPath(currentState)

 successors = generateSuccessors(currentState)

 for successor in successors:
 priorityQueue.enqueue(successor, heuristic(successor))

return "No solution found"

function heuristic(state):
 countMisplacedTiles = 0
 for each tile in state:
 if tile is not in its correct position:
 countMisplacedTiles++
 return countMisplacedTiles

function constructSolutionPath(goalState):
 path = []
 currentState = goalState
 while currentState has parent:
 path.prepend(currentState)
 currentState = currentState.parent
 path.prepend(initialState)
 return path
```

Conclusion:

_____
_____
_____
_____
_____
_____
_____
_____
_____
__

References:

S. Russel, P. Norvig, “Artificial Intelligence – A Modern Approach”, Third Edition, Pearson Education, 2015.

Department of Artificial Intelligence and Data Science