CS561 - ARTIFICIAL INTELLIGENCE

ASSIGNMENT-1:DFS,BFS

Intaj Choudhury – 2211MC09

Ankit Anand – 2311MC04

Khushbu Bharti – 2311MC21

Question

The task is to check if we can reach from any random start grid to the mentioned target grid by moving the Blank space('B'). In one step, the Blankspace can move either top or down or left or right.

1. Compare Breadth First Search (BFS) and Depth First Search (DFS) with respect to the number of steps required to reach the solution if they are reachable.

Solution

1. Algorithm:

Step 1 : Take initial state as input from the user and target state is fixed.

Step 2 : Inversion count for all elements is calculated and added.
If the sum is odd , then the problem is unsolvable (goal state cannot be reached from initial state), the "problem cannot be solved" is printed and program gets exited.

Step 3 : BFS function is called and a queue data structure is used to store the visited states.
Number of steps is counted for bfs to reach the goal state.

Step 4 : DFS function is called and a stack(LIFO) data structure is used to store the visited states.
Number of steps for dfs is calculated to reach the goal state.

Step 5 : Numbers of steps in both the functions is compared and which one performed better is declared.

The order followed in both the bfs and dfs is :
Right, Left, Up, Down.

Target :
[1, 2, 3]
[4, 5, 6]
[7, 8, -1]

Sample Input :
[3, 2, 1]
[4, 5, 6]
[8, 7, -1]

Case 1 : When BFS performs better than DFS.

```
Enter Initial State :
Enter 1 for Random grid.
Enter 2 for user input .
Enter your choice : 2
Enter row  1 :  1 3 2
Enter row  2 :  4 6 5
Enter row  3 :  7 8 -1

-------Initial State-------

[1, 3, 2]
[4, 6, 5]
[7, 8, -1]


-------Target State-------

[1, 2, 3]
[4, 5, 6]
[7, 8, -1]


problem solved with BFS in 24663steps

problem solved with DFS in 117724steps

BFS took less number of steps than DFS
```

Case 2 : When DFS performs better than BFS.

```
Enter Initial State :
Enter 1 for Random grid.
Enter 2 for user input .
Enter your choice : 2
Enter row  1 :  1 2 -1
Enter row  2 :  4 5 3
Enter row  3 :  7 8 6

-------Initial State-------

[1, 2, -1]
[4, 5, 3]
[7, 8, 6]


-------Target State-------

[1, 2, 3]
[4, 5, 6]
[7, 8, -1]


problem solved with BFS in 6steps

problem solved with DFS in 2steps

DFS took less number of steps than BFS
```

Case 3 : When puzzle is not solvable.

```
Enter Initial State :
Enter 1 for Random grid.
Enter 2 for user input .
Enter your choice : 2
Enter row  1 :  1 2 3
Enter row  2 :  5 4 6
Enter row  3 :  7 8 -1

-------Initial State-------

[1, 2, 3]
[5, 4, 6]
[7, 8, -1]


-------Target State-------

[1, 2, 3]
[4, 5, 6]
[7, 8, -1]



problem cannot be solved
```

Output of sample input given in assignment:

```
Enter Initial State :
Enter 1 for Random grid.
Enter 2 for user input .
Enter your choice : 1

-------Initial State-------

[3, 2, 1]
[4, 5, 6]
[8, 7, -1]


-------Target State-------

[1, 2, 3]
[4, 5, 6]
[7, 8, -1]


problem solved with BFS in 131207steps

problem solved with DFS in 152813steps

BFS took less number of steps than DFS
```

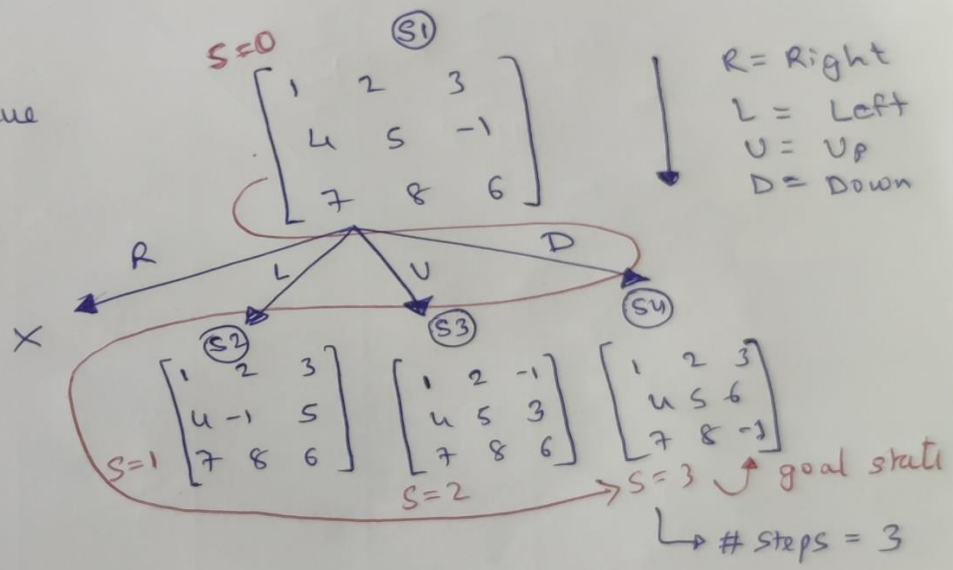
2. Comparison of BFS and DFS

Initial State = $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & -1 \\ 7 & 8 & 6 \end{bmatrix}$

Goal State = $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & -1 \end{bmatrix}$

① BFS

Use Queue



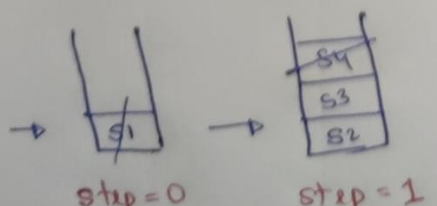
② DFS

Use Stack

(i) Push S1 in stack

(ii) Pop S1 and push its childmen in stack in R, L, U, D order.

(iii) Again repeat till you reach goal state.



S4 = goal

steps = 1

Breadth First Search (BFS)

- Breadth-first search is the most common search strategy for traversing a tree or graph. This algorithm searches breadth-wise in a tree or graph, so it is called breadth-first search.
- BFS algorithm starts searching from the root node of the tree and expands all successor node at the current level before moving to nodes of next level.
- The breadth-first search algorithm is an example of a general-graph search algorithm.
- Breadth-first search implemented using FIFO queue data structure.

Time Complexity: Time Complexity of BFS algorithm can be obtained by the number of nodes traversed in BFS until the shallowest Node. Where the d= depth of shallowest solution and b is a node at every state.

$T(b) = 1+b+b^2+b^3+....+ b^d= O(b^d)$

Space Complexity: Space complexity of BFS algorithm is given by the Memory size of frontier which is $O(b^d)$.

Depth First Search (DFS)

- Depth-first search is a recursive algorithm for traversing a tree or graph data structure.
- It is called the depth-first search because it starts from the root node and follows each path to its greatest depth node before moving to the next path.
- DFS uses a stack data structure for its implementation.
- The process of the DFS algorithm is similar to the BFS algorithm.

Time Complexity: Time complexity of DFS will be equivalent to the node traversed by the algorithm. It is given by:

$T(n)= 1+n+n^2+ n^3 +....+n^m=O(n^m)$

Where, m= maximum depth of any node and this can be much larger than d (Shallowest solution depth)

Space Complexity: DFS algorithm needs to store only single path from the root node, hence space complexity of DFS is equivalent to the size of the fringe set, whichis **O(bm)**.

- If Depth of state space >>>>Branching factor, then BFS perform better.
- If Depth of state space <<<<Branching factor, then DFS perform better