

# ARTIFICIAL INTELLIGENCE PROJECT 1

## Program design and Actions:

This implemented program reads input from the test cases and performs state space search with two algorithms, namely Breadth First search and A-star algorithm. I have created state of this algorithm to be a string values of length 9 [ex: 0123456789 ].

I have represented my action such as RIGHT, LEFT, DOWN, UP and assigned a value of (0,1,2,3) to them in an order, for every move in 3\*3 board I have assigned the possible moves of (0,1,2,3)

As soon as the move is performed, swap function will swap zero(0<sup>th</sup> element on the board) and other elements, which will be determined by the legal moves with its legal move (0,1,2,3). I implemented a list that can hold the value of multiple possibilities of legal moves (0,1,2,3).

For every position in the board, I associated the possible list of legal moves. By doing this I made sure that only right moves could be played.

For Implementing the A-star Algorithm, I have taken a cost function, which will work on the process of calculating the **Manhattan distance**(which is measured by calculating the amount of misplaced elements on the board, this is compared with the goal state to calculate this value)

$$F(n)=h(n)+g(n)$$

Here  $h(n)$  is the Manhattan cost and  $g(n)$  is the cost of the path from the starting node

We usually search the least possible value of  $f(n)$ [cost]

# Results:

## BFS(breadth first search):

**Execution time:** This algorithm is taking about 44 seconds to run on all test cases, It is taking more time to complete the difficult test cases compared to easy and medium test cases.

```
Steps: 27
4 0 2
Searching Time: 9.921611547470093

26) File: examples/difficult/3x3_5:
['D', 'R', 'U', 'U', 'L', 'D', 'D', 'R', 'R', 'U', 'L', 'U', 'R', 'D', 'L', 'L', 'D', 'R', 'R']
Steps: 19
AStar steps: 19
Searching Time: 2.296339273452759

27) File: examples/difficult/3x3_6:
['L', 'U', 'R', 'D', 'D', 'R', 'U', 'L', 'L', 'D', 'R', 'R', 'U', 'L', 'U', 'L', 'D', 'R', 'D', 'R']
Steps: 20
AStar steps: 20
Searching Time: 3.2284610271453857

28) File: examples/difficult/3x3_7:
['L', 'L', 'U', 'R', 'R', 'D', 'L', 'L', 'D', 'R', 'R', 'U', 'U', 'L', 'D', 'L', 'D', 'R', 'R']
Steps: 19
AStar steps: 19
Searching Time: 1.8276550769805908

29) File: examples/difficult/3x3_8:
['R', 'D', 'L', 'U', 'U', 'R', 'R', 'D', 'D', 'L', 'U', 'L', 'U', 'R', 'R', 'D', 'L', 'D', 'R']
Steps: 19
AStar steps: 19
Searching Time: 2.009901762008667

30) File: examples/difficult/3x3_9:
['R', 'R', 'U', 'L', 'D', 'L', 'U', 'R', 'U', 'R', 'D', 'L', 'U', 'L', 'D', 'D', 'R', 'R']
Steps: 18
AStar steps: 18
Searching Time: 1.046628713607788

LEVEL TIME: 42.60852670669556
Overall Time: 43.967615365982056

In [140]: runcell('[85]', 'C:/Users/Admin/Desktop/Bfs/Untitled27.py')
```

In the above figure we have to note that L,R,U,D are the type of operations done to reach the goal state

In the above figure, “Overall Time:” is the full time taken by the algorithm to run on all test cases, whereas, “Searching Time:” is the total time taken by the algorithm to solve that particular test case(single test case)

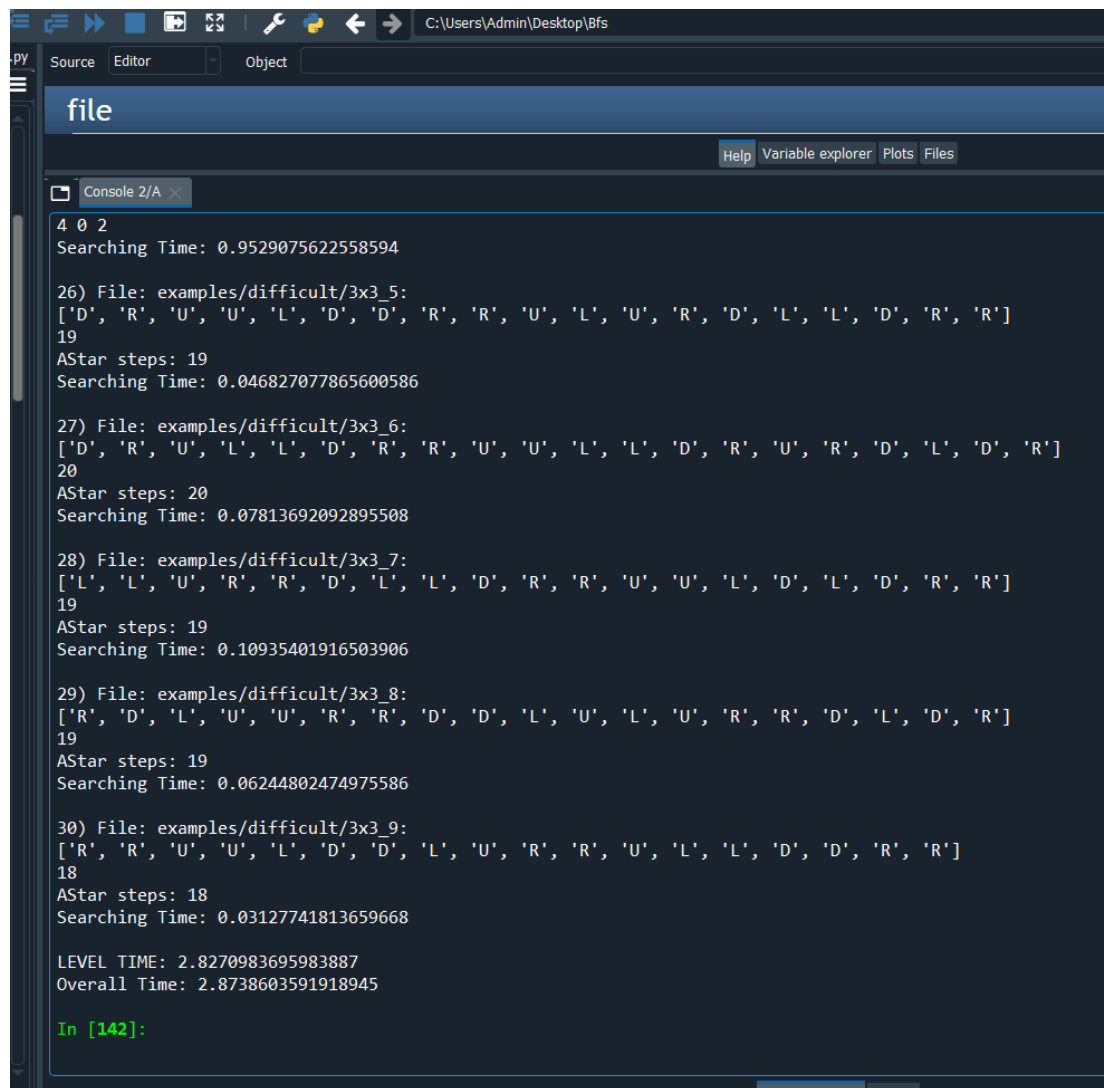
## A- Star Algorithm:

### Execution time:

This algorithm takes about 2 to 3 seconds to complete all test cases.

and D, R, L, U (down, right, left, up) are possible actions that are being performed to reach the goal state. Here also I saw that it took more time to complete the difficult cases, but it did not matter because even if it takes more time compared to the other test cases(easy),

It is still being completed in less than a second. So comparatively A-star is faster than Bfs algorithm.



```
4 0 2
Searching Time: 0.9529075622558594

26) File: examples/difficult/3x3_5:
['D', 'R', 'U', 'U', 'L', 'D', 'D', 'R', 'R', 'U', 'L', 'U', 'R', 'D', 'L', 'L', 'D', 'R', 'R']
19
AStar steps: 19
Searching Time: 0.046827077865600586

27) File: examples/difficult/3x3_6:
['D', 'R', 'U', 'L', 'L', 'D', 'R', 'R', 'U', 'U', 'L', 'L', 'D', 'R', 'U', 'R', 'D', 'L', 'D', 'R']
20
AStar steps: 20
Searching Time: 0.07813692092895508

28) File: examples/difficult/3x3_7:
['L', 'L', 'U', 'R', 'R', 'D', 'L', 'L', 'D', 'R', 'R', 'U', 'U', 'L', 'D', 'L', 'D', 'R', 'R']
19
AStar steps: 19
Searching Time: 0.10935401916503906

29) File: examples/difficult/3x3_8:
['R', 'D', 'L', 'U', 'U', 'R', 'R', 'D', 'D', 'L', 'U', 'L', 'U', 'R', 'R', 'D', 'L', 'D', 'R']
19
AStar steps: 19
Searching Time: 0.06244802474975586

30) File: examples/difficult/3x3_9:
['R', 'R', 'U', 'U', 'L', 'D', 'D', 'L', 'U', 'R', 'R', 'U', 'L', 'L', 'D', 'D', 'R', 'R']
18
AStar steps: 18
Searching Time: 0.03127741813659668

LEVEL TIME: 2.8270983695983887
Overall Time: 2.8738603591918945

In [142]:
```

## **Discussion:**

The time of completion of the program surprised me, I knew it would be faster, but it was way faster than I expected. One of my biggest challenges in this course was implementing BFS and A-star algorithms in python, as I have to learn the languages and simultaneously build the project. In this project I learnt the importance of usage of heuristics and other techniques used to implement the project. I found this project to be very cool, because it helped me in implementing a game and solving it in least amount of time as possible. This project makes me realize that most of the real-world games are solvable, (eg: AlphaGo, chess) if we can develop a good method to crack it.