DATA STRUCTURES AND ALGORITHMS

PROJECT PROPOSAL

PROJECT TITLE: THE A* (A – START) ALGORITHM

GROUP MEMBERS:

- MUHAMMAD ZAIN YOUSUF
- MUHAMMAD ZAEEM BAIG
- EESHA IFTIKHAR QAZI

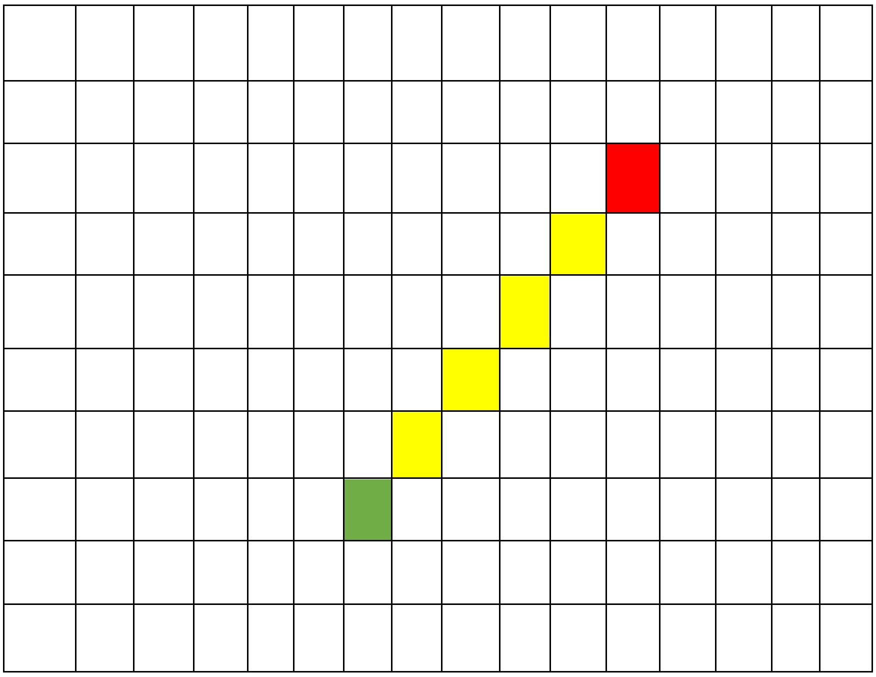
PROJECT DESCRIPTION:

The A* search algorithm is an Artificial Intelligence (AI) algorithm that is used for graph traversals and path finding. It is frequently used in computer science due to its completeness, optimality and accuracy. A* algorithm has a higher rate of precision because it utilizes heuristics to perform its search.

The A* algorithm is a pretty smart algorithm in determining the shortest path from the starting to the goal position. Consider the square grid below on which the starting point is marked from the green, and the goal is marked is red. What A* algorithm does is that it selects a node from the starting point with respect to$f(n)$, and the node with the lesser value is chosen. The path is shown from yellow. Similarly, from that node, it selects another node towards the goal position in the same manner, the node with the least value is chosen. What interesting here is the calculation of $f(n)$. It is calculated as follows:

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

Where, $g(n)$ is the distance from the initial node to the other node, and $h(n)$ is the heuristic part, and it is the approximate distance from the next node to the goal node. It can be calculated through several methods, however, we will use Euclidean distance formula $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$ to calculate $h(n)$.



PROJECT SCOPE:

1. 8-PUZZLE:

In this project, we will be solving the 8-puzzle problem. An 8-puzzle problem is a popular puzzle game that consists of 3 rows and 3 columns. We will be given 8 known tiles and one empty tile which can be replaced or moved between the tiles in order to obtain the goal state. We will be implementing A* to reach our goal state in the optimal manner. Here, the tiles will be the *Nodes*, and the start and goal configurations are the *States*. Each time when we move the empty node to reach our goal state, the initial state will get updated, and we need to keep track of each node state. Now, A* comes into play here. It generates new states, processes the data of the state, and keeps into the recorded list or visited list so that we may not end up processing one state multiple times. So, how does the A* algorithm finds the optimal path to the goal state? Again, it uses the above stated methods to do so. The state with least value of $f(n)$ is selected until we reach the destination. It generates the value of $f(n)$ as follows:

- $f(n) = g(n) + h(n)$
- Where $g(n)$ is the number of nodes traversed from the start node to current node, and
- $h(n)$ is the number of misplaced tiles by comparing the current state and the goal state.

- It generates the value of $h(n)$ by comparing the initial and the goal states, and the number of misplaced tiles represent the value of $h(n)$ and $g(n)$ is the cost of moving the empty tile from the starting node.

- It will create the children of the current state by moving the empty in four directions, UP, DOWN, LEFT and RIGHT, and calculates the value of $f(n)$ as mentioned above.

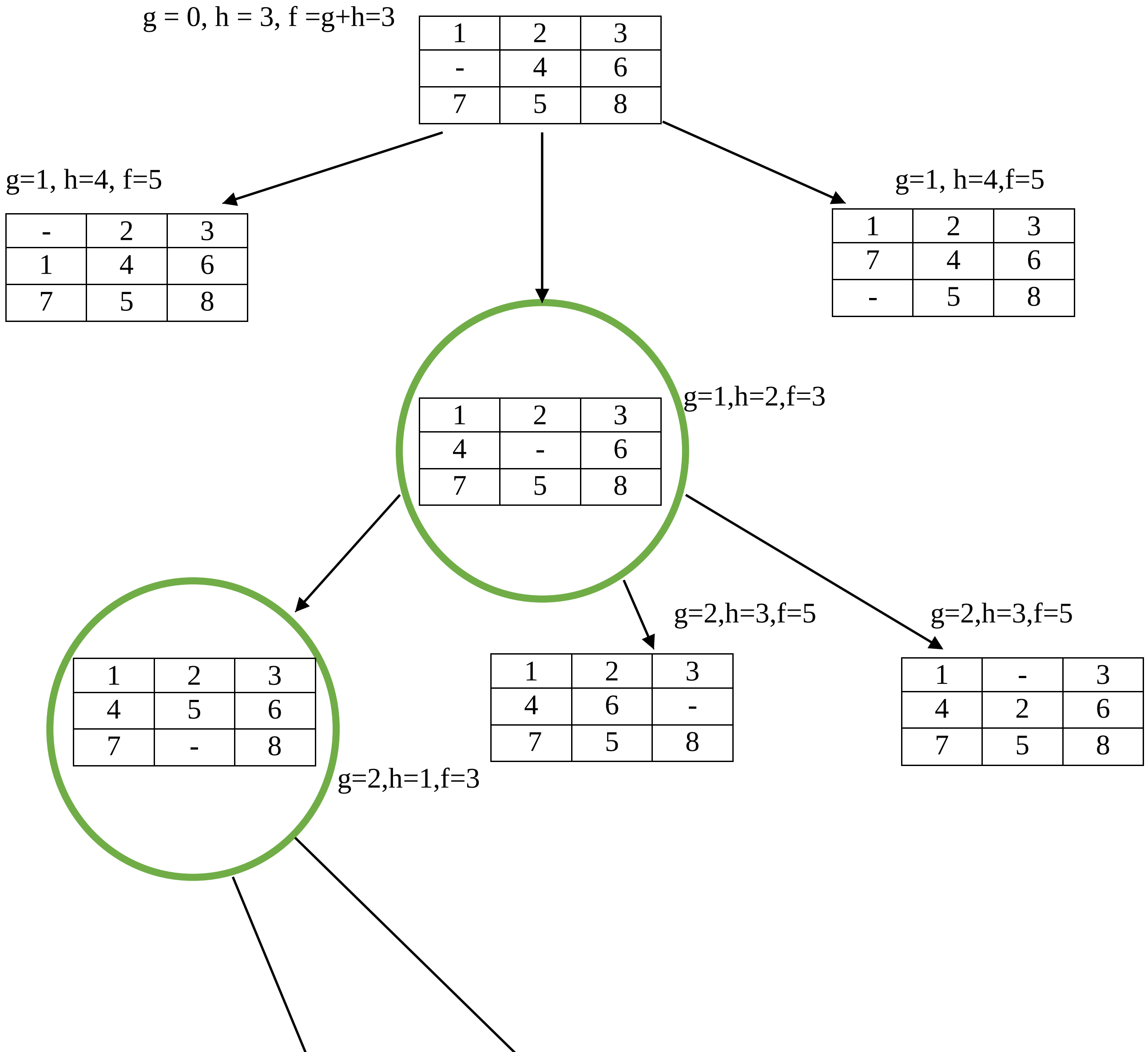
- It keeps generating sub-state of each parent state, and calculates $f(n)$ until we reach our goal state. In this manner, we can find the optimal path to solve the 8-puzzle.

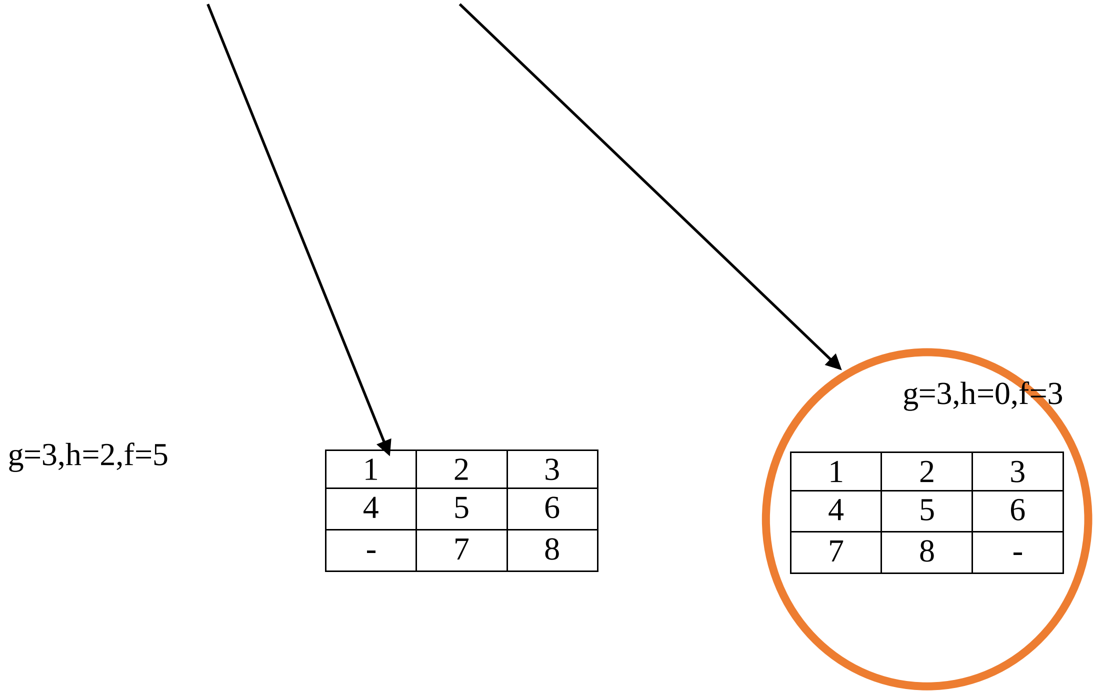
Initial State

1	2	3
-	4	6
7	5	8

Goal State

1	2	3
4	5	6
7	8	-





g=3,h=2,f=5

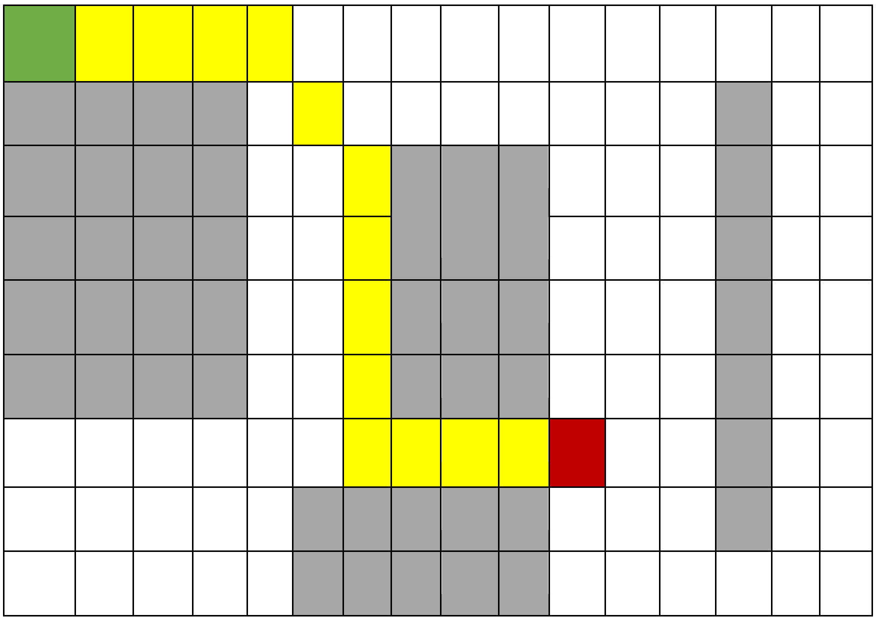
1	2	3
4	5	6
-	7	8

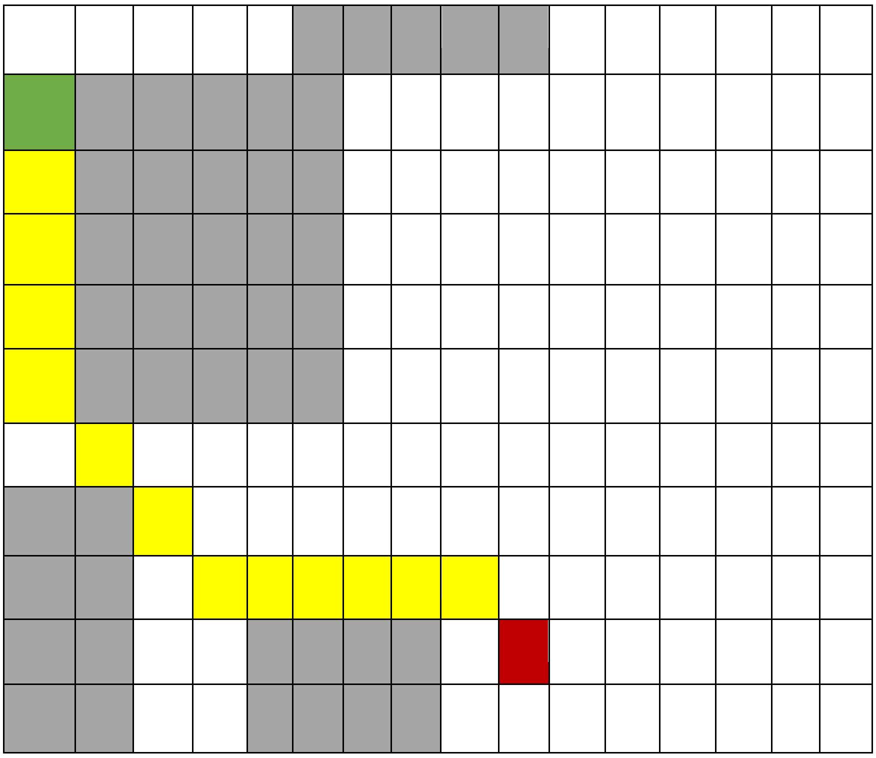
g=3,h=0,f=3

1	2	3
4	5	6
7	8	-

2. MAZE SOLVING:

A-star algorithm has vast applications, and is used in variety of fields and programs. From robotics, machine learning and AI based applications to games, maps and graphs, programmers highly rely upon A-star algorithm. Therefore, in our project too, we will be implementing A-star for path-finding. As it is stated above, A-star is widely used in Game Development, and in our project, we will be giving a grid of specific dimension along with some obstacles in it. We need to implement A-star to find the shortest and the smartest path from the starting point to the goal. Similar to what is implemented in games, if you want to go from one place to another or find a specific character in the game, you are directed towards it, and is given the path to the desired location by the game engine itself. On the backhand, it is actually A-star that is implemented in the game. Therefore, we will be solving this kind of a problem in our project. Following are a few examples to give the insights of our project.





Here, in this demonstration, yellow denotes the path, green starting point and red destination. We can see the how smartly the algorithm chooses the path to the goal, and cuts between the obstacles marked by the gray color. That is why it is said that this is the algorithm with "brains" in it.

3. STRING COMBINATION:

In this application of A*, we will be given an unarranged string, a starting state, and we need to convert the string into the arranged form that is our goal state using different combinations. We can do this by generating random combinations of our starting goal, and we may reach to our goal state, however, that is really inefficient, and we may never reach our goal state and end up in an infinite loop. Here, A* again gives the optimal path to reach to our goal state using the same technique.

First, it generates children of state by rotating the string in random manner and generates $f(n)$ by calculating the distance of each letter as follows:

- Calculates $g(n)$ each time when we move away from the starting letter. In this case, it keeps adding the index of the letter.
- Calculates $h(n)$ by comparing the index of the letter in the starting state to the goal state
- Adds $g(n)$ and $h(n)$ to generate $f(n)$ of each child.

Secondly, it stores the child and its $f(n)$ value in in a Priority Queue, and when we are done with generating child, we simply *get* the child state from Priority Queue which stores the value in an arranged ascending order. In this manner, we can get the child of minimum cost value of $f(n)$ and reach our goal in the shortest path possible.

One example of this application is given below:

```
Enter starting string: ytpnho

Enter goal string: python

Starting...

0) ytpnho

1) yptnho

2) pytnho

3) pythno

4) python
```

PROJECT OUTCOME:

The one of the most prominent reasons why we chose this algorithm is that it is an Artificial Intelligence algorithm. It is commonly known as the algorithm with brains. Thus, it would give some exposure of AI, and how it works since it is an emerging field, and most of us want to pursue this in future. Furthermore, we will be introduced to some new concepts of data structures and algorithms such as how to create nodes and the heuristics part in the algorithm. Therefore, we will have some background of AI algorithms and heuristic, and some handful knowledge of how to create classes and nodes.

Moreover, the other dominant outcome of this project is that we will be introduced to github. It is an online platform for developers where we can collaborate with our colleagues to write codes and other documentation, and it can be shared with others too easily. Therefore, it will be really beneficial for us in our corporate future as we will get acquainted with github now.

Other than that, we will learn how to work in a team more productively, and it will help us to hone our soft skills which are key aspects for becoming successful in today’s world.

LIBRARIES/OTHER RESOURCES TO BE USED:

In this project, we may need to use the following resources:

- We will need to create new classes for nodes, for parent and children nodes.
- Best search or informed search algorithms.
- Dijkstra algorithm implementation.
- Lists, tuples and basic python data structures.
- We may need to use Priority Queue for the implementation of A* algorithm.

References

A* Search Algorithm. (2018, September 07). Retrieved June 17, 2020, from https://www.geeksforgeeks.org/a-search-algorithm/

A* search algorithm. (2020, June 04). Retrieved June 17, 2020, from https://en.wikipedia.org/wiki/A*_search_algorithm

Payne, T. (2013, December 21). *Let's Learn Python #20 - A* Algorithm* [video]. YouTube. https://www.youtube.com/watch?v=ob4faIum4kQ&t=194s

Swift, N. (2020, May 29). Easy A* (star) Pathfinding. Retrieved June 17, 2020, from https://medium.com/@nicholas.w.swift/easy-a-star-pathfinding-7e6689c7f7b2

Sonawane, A. (2020, January 08). Solving 8-Puzzle using A* Algorithm. Retrieved June 17, 2020, from https://blog.goodaudience.com/solving-8-puzzle-using-a-algorithm-7b509c331288