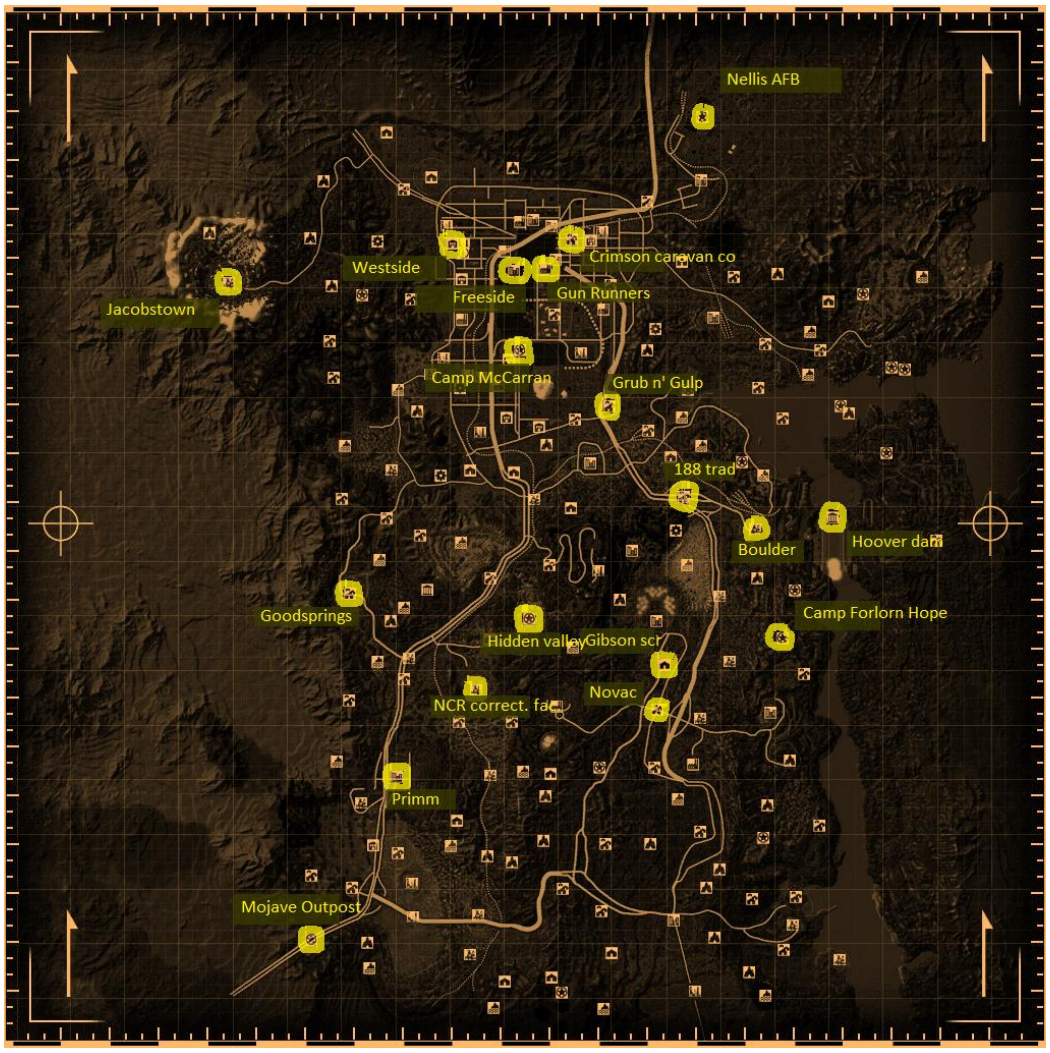
# Example 4 – Search Algorithms

## Problem Statement:

In order for a speed runner to get the optimal time in a game, the video game character must travel to a certain city from a starting point. In order to get the most optimal time a search algorithm has been developed to discover which cities the character should visit in order to get to the designated city in the fastest time. In order to do this the following map was provided from Fallout New Vegas.



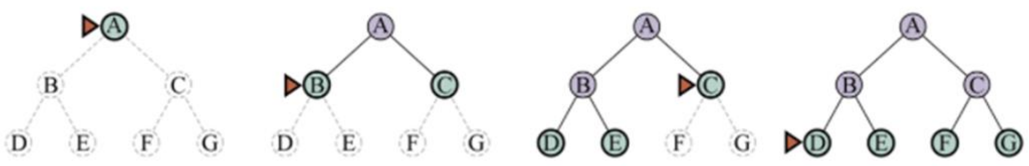
Source: (https://fallout.fandom.com/wiki/Fallout:_New_Vegas_merchants)

In order to finish the game, the character must travel from Goodsprings to Nellis AFB. The goal of the searching algorithm is to find the optimal route. In order to implement this a breadth first search (BFS) and a Depth first search (DFS) will be used.

## Description of the AI technique:

There are many different searching algorithms but for this example the focus will be on uninformed searching strategies. The main technique within the uninformed searching strategies that will be focused on is the BFS and DFS. The BFS searches each layer one by one until the goal node is found. Once all the nodes on a single layer have been explored an iteration of the search is finished and a new interaction begins from any one of the nodes that had been searched. A visual representation of this process is seen below:

Figure 3.8

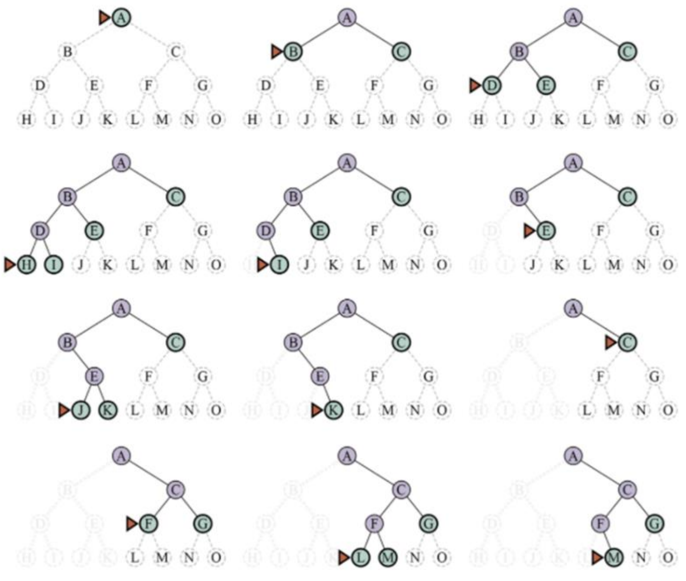


Breadth-first search on a simple binary tree. At each stage, the node to be expanded next is indicated by the triangular marker.

Russell S, Norvig, P. (2020)

The search starts at A and in this case the first iteration would be complete once B and C have been explored as they are the next layer down from A. This process continues with new iterations until all the nodes have been explored. This method is complete and will provide the minimum cost for getting to the designated goal if all the costs are the same. This is because once you start a BFS and you find the specified vertex, you know that the path you've traced so far is the shortest path to the node because if there were a shorter path, the BFS would have found it already. This makes the BFS optimal for finding the shortest path.

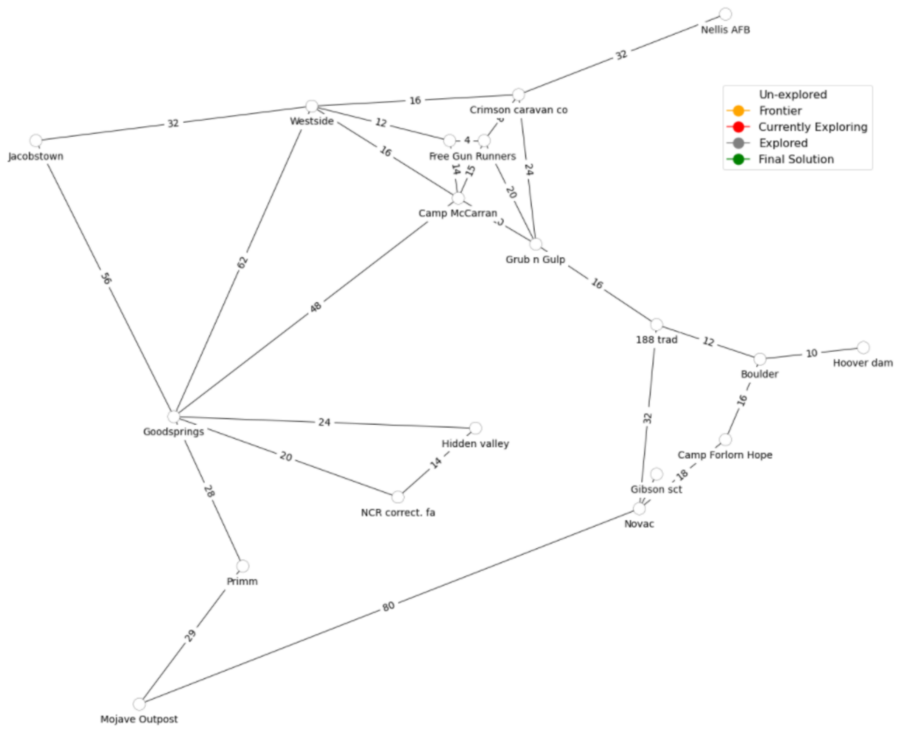
In order to calculate what is considered the shortest path a cost function is implemented. The cost function is the cost from moving from one node to another. In the case of searching through cities the cost would be the distance from one city to the next and the shortest path would be the path were the shortest amount of distance between the cities is covered. The important specification with distances, however, is the fact that they are not all the same. The distance from one city may be more than the distance from another city. This means that the BFS search may not find the shortest path. To find the most optimal path using the uninformed searching strategies a depth first search (DFS) will also be used to compare. The difference with a DFS is that the search continues down each node until it finds the end where there are no longer any more branching nodes and then continues searching from the closest nearby node. To visualise this the following diagram is provided:



Russell S, Norvig, P. (2020)

## How the AI technique is used to solve the problem:

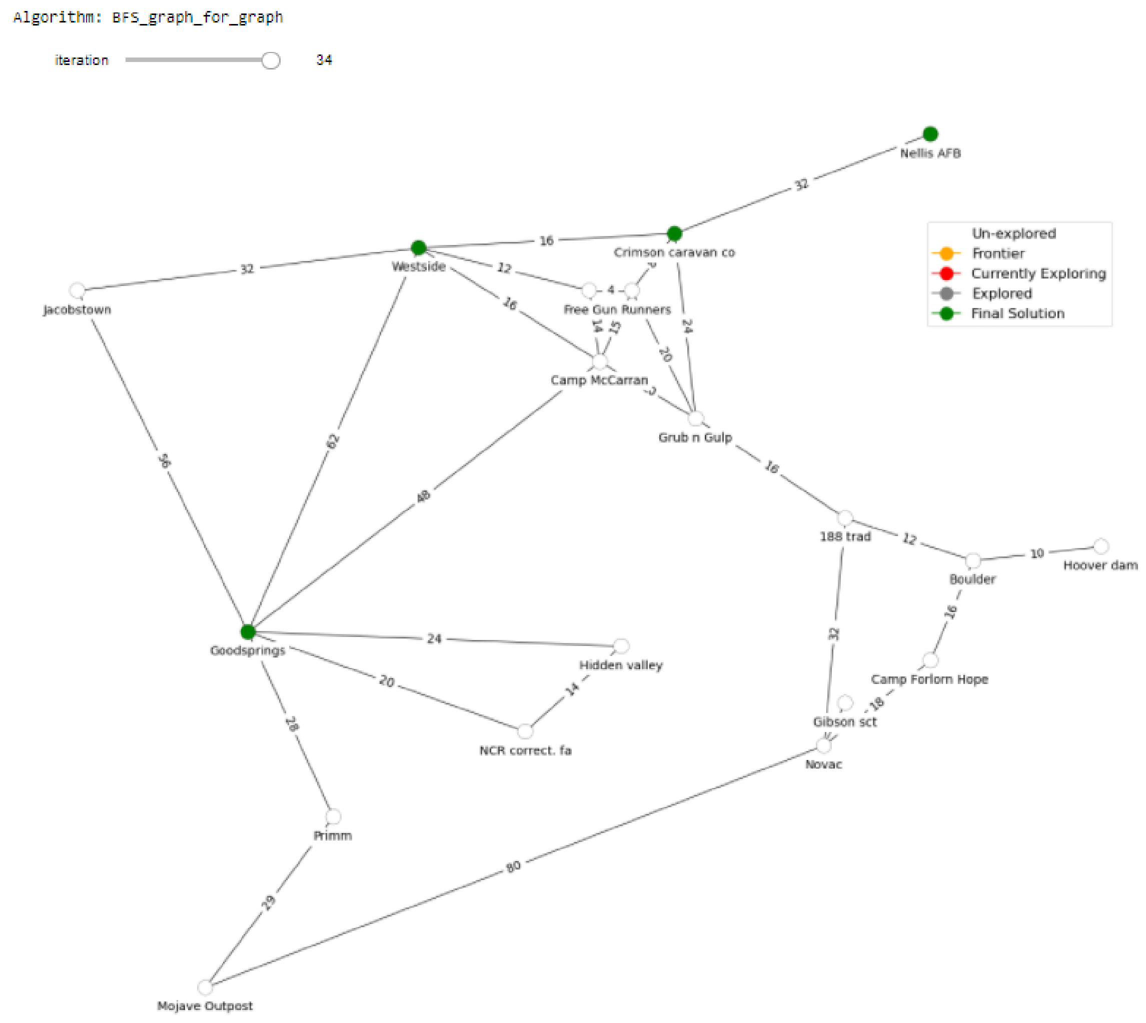
The first step when implementing this AI technique is to develop a map that the program can recognise. The same distances between the cities in the game and between the nodes had to be found, so that the optimal route discovered is actually useful in game. To do this the in-game map was scaled. To scale the map the real-life size of the map was found, being 22020.08 km^2, then by looking at the map it can be seen that there are 19 by 19 squares meaning that each square is 61km^2 and therefore the length of one side of a square on the map is 8km. Using this measurement system, the following nodes were generated:



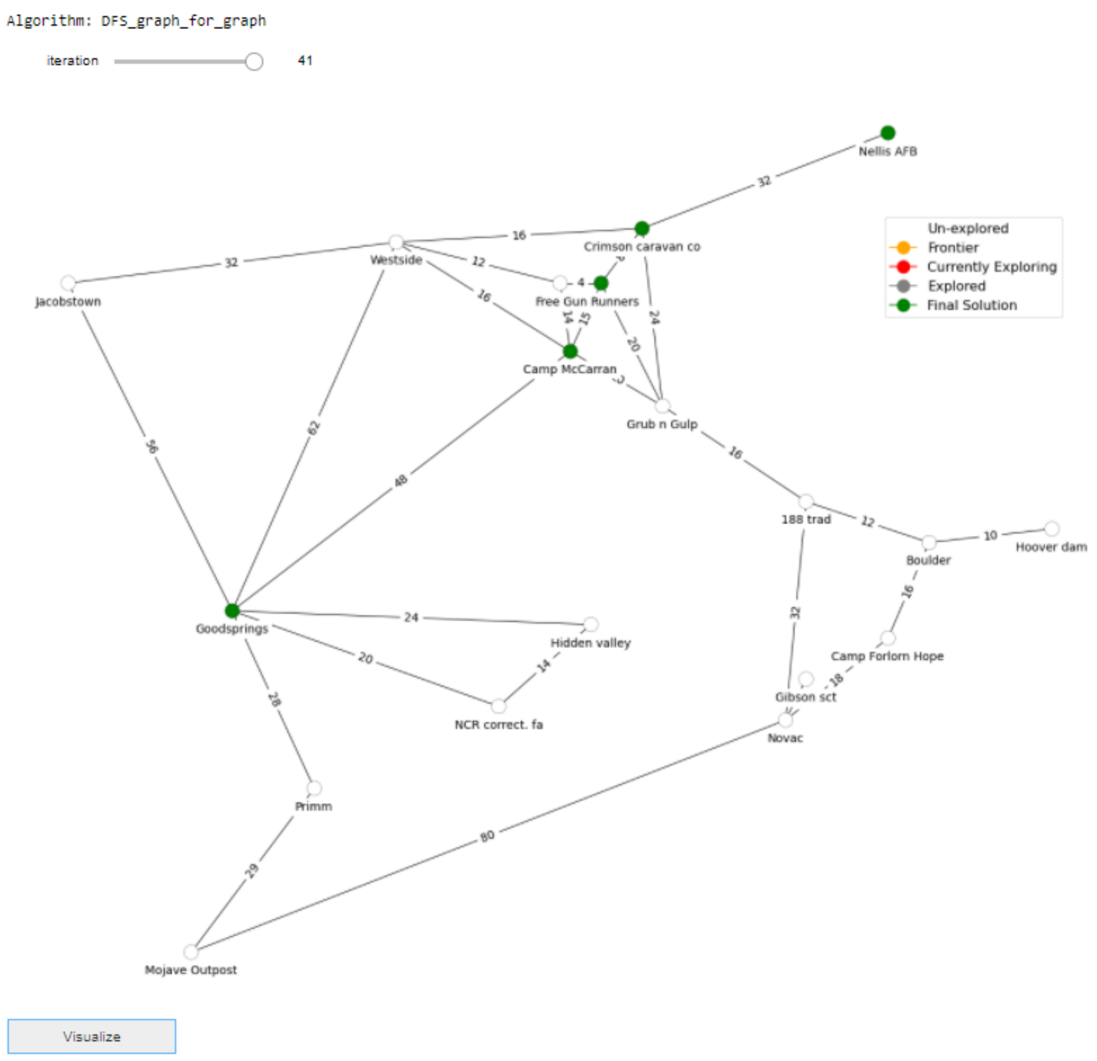
Each of the numbers between the nodes is the cost from moving from one node to the other for example moving from Mojave Outpost to Novac would have a cost of 80 while moving from Goodsprings to camp McCarran would have a cost of 48. Once the map was set up the BFS was implemented to search through the nodes one layer at a time. To understand what was going on visually the key in the top right of the map colours the node depending on if the node is being explored, whether it has been explored, or whether the final solution has been found. If a node is yellow, it means it is the upcoming frontier. To understand this essentially the BFS and DFS have a searching tree and each time a new layer is searched the nodes from that layer are added to the tree, this helps the BFS remember which nodes have already been explored. This is called expanding the frontier. The nodes added to the tree that haven't been searched yet are called the frontier nodes and are labelled in yellow until they are eventually searched. The frontier of the search tree will continue to get expanded until the goal node is reached. For this problem the search will begin at the root node Goodsprings and will continue until the goal node Nellis AFB is reached, providing the optimal route.

## Solution Results:

The final solution the BFS found can be seen below:



The searching algorithm took 34 iterations and found a solution with a total cost of 110 meaning that in game using the algorithm the character could get to Nellis AFB in 110km. To test whether the BFS was the most optimal searching algorithm for the uninformed searching strategy the following map was also tested with DFS. The results for the DFS are displayed below:



The DFS took longer to find the final solution with 41 iterations, however, found a more optimal path only costing 103, meaning the character would only have to travel 103km in game. This is superior to the BFS which makes sense seeing as the cost for each of the nodes are vastly different.

Overall, the searching algorithms provided a much more optimal path than purely guessing which city to go through with the DFS finding the most optimal path for the speed runner to take. To improve the searching algorithm an informed algorithm could have been used such as A* since the solution isn't very deep nor is the data set large and an estimated cost function could have been developed due to the distances between the cities being known. A greedy best first search could have also been used if the searching strategy was informed which may have improved the number of iterations required, however, overall, the problem statement was solved by finding the optimal path using an uninformed searching strategy.

## References:

Fallout: New Vegas merchants. (2021). Retrieved from https://fallout.fandom.com/wiki/Fallout:_New_Vegas_merchants

Lu, H. 2021, ‘Module 5 -- Search algorithms Part I’, UTS Online Subject 41040 lecture notes, UTS, Sydney, viewed 21 September 2021, < https://canvas.uts.edu.au/courses/18678/modules>.

Russell, S. J., Norvig, P. (2020). Artificial intelligence: A modern approach, 4th Edition. Pearson.