Problem 1: The Blocks World

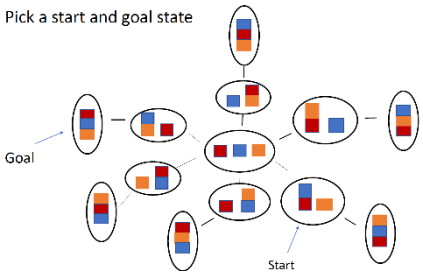
My strategy for implementing the Blocks World assignment was to create my own python script file, which is titled: BlocksWorld.py. My world has three (3) blocks which are enumerated as 'a', 'b', and 'c'. The way I represented the various states is as follows: ['a_b_c'] which means all three blocks are on the table whereas ['ac_b'] represents block c is on the top of block a and block b is on the table. I decided to represent the Blocks World as a graph representing all thirteen states along with their successor states. The graph is a dictionary composed of strings as keys and sets of lists for the values.

In the instructions, it was noted that the relative position of a block or stack of blocks on the table (left or right) was irrelevant. Therefore I created additional function called reorder to take care of relative position. There are three search algorithms which can be implemented in Blocks World, which are depth-first, breadth-first, and best-first searches. In the best-first search algorithm, I created a heuristic which evaluates the path from start to goal. Given the lack of complexity of the graph which spans out and each path has one successor, the heuristic was quite basic. Nonetheless, the implemented heuristic does the job in our Blocks World program.

In order to run BlocksWorld.py, you will need to call the BlocksWorld() function. Below are examples of output.

```
====== RESTART: E:/Winter Quarter 2018/CSC480/Homework_2/BlocksWorld.py ======
>>>
>>> BlocksWorld()
What is your strategy you would like to implement onto Block World? You may enter: dfs, bfs, best.    dfs
What is your starting position? cab
What is your goal position? bca
[['cab', 'ab_c', 'a_b_c', 'b_ca', 'bca']]
>>> BlocksWorld()
What is your strategy you would like to implement onto Block World? You may enter: dfs, bfs, best.    bfs
What is your starting position? bac
What is your goal position? a_cb
[['bac', 'ac_b', 'a_b_c', 'a_cb']]
>>> BlocksWorld()
What is your strategy you would like to implement onto Block World? You may enter: dfs, bfs, best.    best
What is your starting position? cba
What is your goal position? a_bc
This is the path ['cba', 'ba_c', 'a_b_c', 'a_bc'] and here is the length of the path 3.
>>>
```

Problem 2: The 8 Puzzle Game



In this problem, our task was to create ar instructor's best first search estimator which proves our estim ing the number of tiles that are in the correct position. The Man cided to implement. The

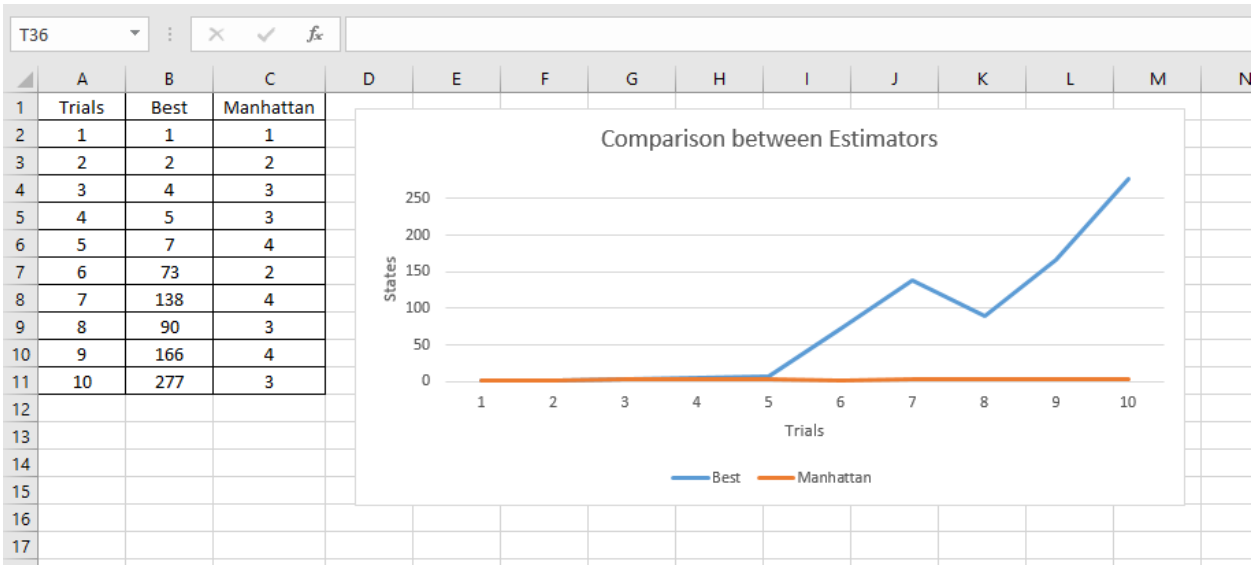
goal of the Manhattan distance is to calculate the distance between two points on the horizontal and vertical paths instead of diagonal (which is the Pythagorean Theorem).

This problem was extremely challenging given the complexity and how to incorporate the Manhattan distance into python class files provided to us. Nevertheless, it was awarding after the program was finally able to work. I had to modify both files: eight_puzzle_state_2.py and search.py. The modifications in the eight_puzzle_state_2.py were: including additional classes (__len__, __getitem__, __iter__, __index__). In the search.py file, I added the search strategy which was the Manhattan distance and instructions how to find the new states into the Manhattan distance function. Also, I created the Manhattan distance function which calculates the distance for each successor state and returns the successor state which has the least Manhattan distance.

Below is sample output (which will be graphed later) of the best and manhattan distance estimators:

```
 RESTART: E:\Winter Quarter 2018\CSC480\Homework_2\hw2_python_latest\search.py
>>> compare(['best','manhattan'],easiest = 1, hardest =10, trials =10, max_states =100)
,best,manhattan
1,1,1
2,2,2
3,4,2
4,5,3
5,5,2
6,16,4
7,54,3
8,27,4
9,90,3
10,73,2
>>> compare(['best','manhattan'],easiest = 1, hardest =10, trials =10, max_states =1000)
,best,manhattan
1,1,1
2,2,2
3,4,3
4,5,3
5,7,4
6,73,2
7,138,4
8,90,3
9,166,4
10,277,3
>>>
 RESTART: E:\Winter Quarter 2018\CSC480\Homework_2\hw2_python_latest\search.py
>>> compare(['best','manhattan'],easiest = 1, hardest =10, trials =10, max_states =100)
,best,manhattan
1,1,1
2,2,2
3,4,2
4,5,3
5,7,3
6,8,3
7,36,4
8,60,4
9,87,3
10,46,3
```

Second part of the assignment was to graph on the runs in excel, which can be seen below.



I decided to plot the second run with max_states =1000. It is observed that Manhattan is the better of the two estimators.

Problem 3: The Jealous Agents Problem

"Three actors and their three agents want to cross a river in a boat that is capable of holding only two people at a time. Each agent is very aggressive and when given the chance would convince an actor to sign a new contract with the new agent ("poaching"). Since each agent is worried their rival agents will poach their client, we add the constraint that no actor and a different agent can be present by themselves on one of the banks, as this is the situation in which poaching occurs."

See the following page for successor states diagram.