Credit Hours System

CMPN402

Cairo University

Faculty of Engineering

Machine Intelligence

Problem 5

Student name: Mahmoud Ezzat ELMaghraby

Student ID: 1170532

Tutorial: Monday (8-11)

Files:

TSP.py : with contains graph class and the problem generator

PriorityQueue.py: contains a queue class which is used in TSP to store the next node to expand

Running the code:

You just have to enter the number of nodes representing the number of cities and the program will present the route found for:

1. A* search
2. Uniform Search
3. Best first Search
4. My designed heuristic using A* search

Textual answers:

3)

MST dominates straight-line distance as is gets the shortest distance found which represents a straight line between 2 cities if they have an edge between them

4)

As showed in the screenshots the A* has found the best path with shortest cost but it consumed the largest time to finish (ignore total_time in these cases)

```
Route found for  A* Search :
0 -> 2
2 -> 8
8 -> 3
3 -> 7
7 -> 4
4 -> 9
9 -> 6
6 -> 5
5 -> 1
1 -> 0
total_distance:  121
total_time:  320
total_cost:  441
Execution time for performance:  1.039466381072998
```

```
Route found for  Best First Search :
0 -> 5
5 -> 6
6 -> 2
2 -> 1
1 -> 3
3 -> 4
4 -> 7
7 -> 8
8 -> 9
9 -> 0
total_distance:  179
total_time:  356
total_cost:  535
Execution time for performance:  0.7864692211151123
```

```
Route found for  Uniform Search :
0 -> 2
2 -> 1
1 -> 9
9 -> 4
4 -> 8
8 -> 3
3 -> 7
7 -> 6
6 -> 5
5 -> 0
total_distance:  131
total_time:  274
total_cost:  405
Execution time for performance:  0.5198092460632324
```

5)

My heuristic calculates time that will be consumed depending on traffic + distance

We will find that by using my heuristic the distance increased but the time decreased and by adding them we will find that the total cost is less

The execution time needed is slightly greater

```
Route found for  A* Search :
0 -> 3
3 -> 7
7 -> 1
1 -> 4
4 -> 8
8 -> 9
9 -> 6
6 -> 2
2 -> 5
5 -> 0
total_distance:  141
total_time:  324
total_cost:  465
Execution time for performance:  0.9808683395385742
```

```
Route found for  Traffic Heuristic with A* Search :
0 -> 3
3 -> 5
5 -> 8
8 -> 2
2 -> 6
6 -> 4
4 -> 1
1 -> 9
9 -> 7
7 -> 0
total_distance:  165
total_time:  235
total_cost:  400
Execution time for performance:  1.080148696899414
```

Code brief explanation:

A graph has n numbers of nodes and edges where each edge has one city at each end, distance and time which is used for my designed heuristic

ProblemGenerator will generate n points with random coordinates and calculate the distance between them and generate a random number for traffic on the road

The function searchPath is then used to begin searching and expanding nodes every node visited is added in visited nodes list and removed from remaining list

So there is a loop that keeps going until there are no more successors or all the nodes have been visited which is the goal

PrimeMST uses a sub graph which is constructed with the remaining nodes to avoid visiting same node again and searches for the node with the least cost found

myTrafficHeuristic function depends on traffic and distance not distance only as the others, so it may choose a longer distance but which has less traffic so the traveller will reach the destination faster, I applied it using A* search

At the end the total distance and time is calculated for each method and displayed with the route found

Assumption:

I assumed that the graph is 50 units on x axis and 50 units on y axis and the distance between any two cities is straight line between them