CSE 6140 / CX 4140 Assignment 2

Ting Liao GTID: 903278773
(collaborator: Chin Wang GTID: 903506612 / Shen-Yi Cheng GTID: 903514405)

Report of Section 3. Programming Assignment

Part a.

Data structure

- UnionFind functions:
  I used the divide and conquer idea to divide the UnionFind function into two separate sub functions. For the Find function, I utilized the recursive idea to let the function call itself again and again until we found the root value. For the Union function, I simply set that node1 will connect to the root value of node 2 in order to connect two different group of nodes.

- Array.Sort
  In order to make sure we assigned the node with lower weight into our function first, I simply sort the graph array by each node's weight. Therefore, it can provide us the correct output.

Time complexity

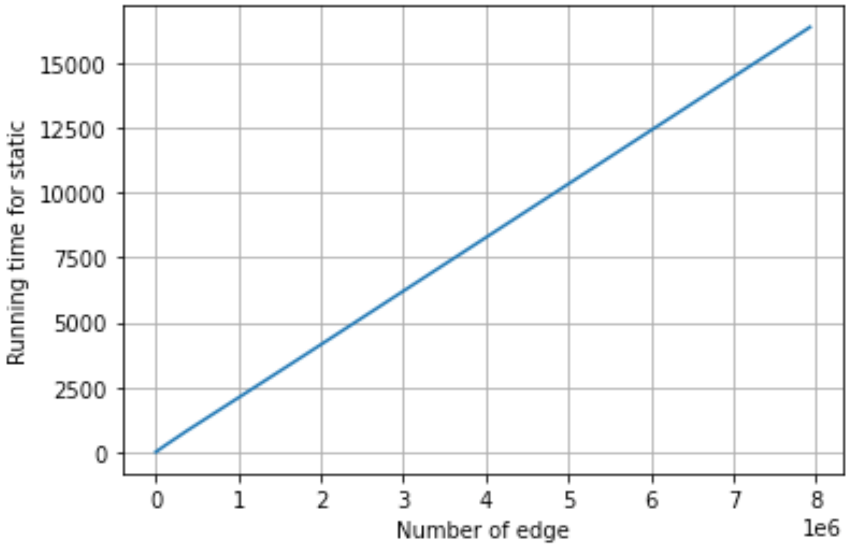
In this MST program, I used the Kruskal's Algorithm because I considered that this Algorithm has much simpler type of data structure than the Prim's Algorithm. Besides, its theoretical run time complexity should be O(E log(E)).

- Function **computeMST()**:
  I used a for-loop to loop through each item in the G, graph array, and use the Find function and the Union function to get our MST. Here, the time complexity is O(E) * (O(logE) + O(logE)), so it should be O(E logE).

- Function **recomputeMST()**:
  Well, the recomputeMST function sort the data first, and then simply call the function computeMST() again to find the MST. I didn't use any heapq or dynamic programming data structure, so the time complex should be the same as computeMST function. Here, the time complexity is O(E logE).

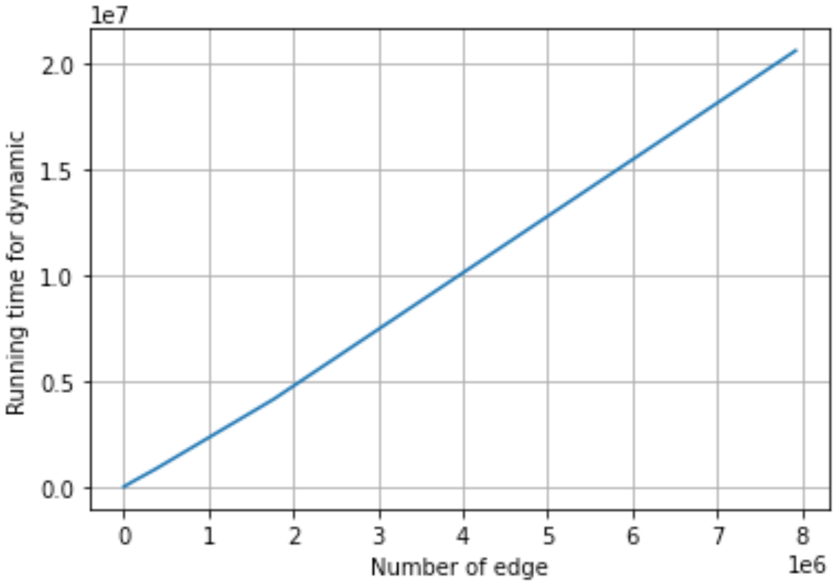
Part b.

Static:



From the plot, it showed that the running time is O(N logN), which is exactly the same as my expectation. To be more specific, the line looked like a straight line, but value of y_scale slightly increased faster after roughly x = 2.3 and it was why the plot represented O(N logN) instead of O(N).

<u>Dynamic</u>:



For the dynamic part, my recomputeMST() simply called the function computeMST() again to find the MST, so the running time should be the same as static part. Also, the plot showed that the running time is O(N logN). However, if I implemented the heap data structure, it might avoid the waste of time of doing the sort array process, the running time of my program could be O(N), which would be much faster than my current program behavior.