Introduction to Algorithms

Lab3 – Travel around the world

Time complexity:

1. Read file
   - Step 1 Read Edge $\mathrm{Big\ O} = O(e)$ e is number of edge
   - Step 2 Sorting all node neighbor by the distance $\mathrm{Big\ O} = O(n * m\ log(m))$ n is the number of node and m is the number of the node neighbor
2. Run algorithm
   - Run the DFS + greedy algorithm and find the path $\mathrm{Big\ O} = O(n * |V| * |E|)$ n is the node and |V|*|E| is the DFS time complexity
   - Run the 2-opt algorithm $\mathrm{Big\ O} = O(n\text{^}2)$ n is the node number

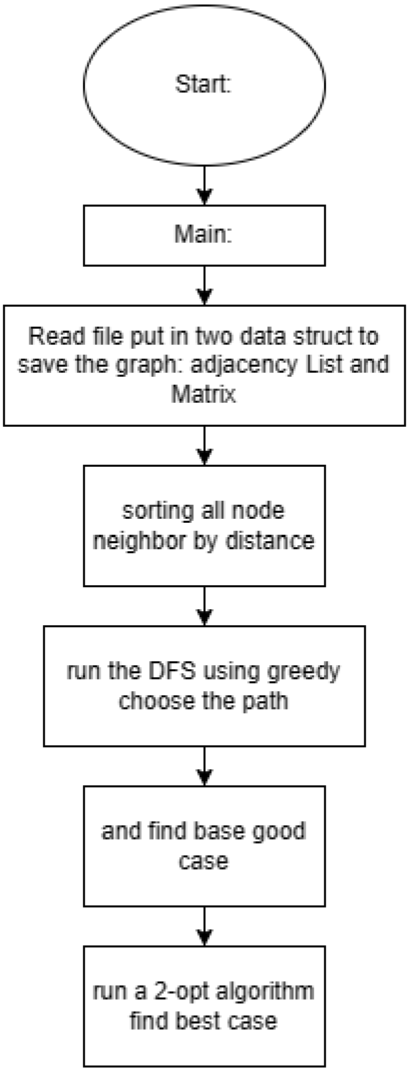
Total time complexity Big $\mathrm{Big\ O} = O(n^2)$

About Design :

In this lab, I used three algorithms to solve the problem: DFS + Greedy + 2-opt. First, I used DFS and Greedy to find all possible cases, but this approach was time-consuming. To improve efficiency, I implemented a "backCnt" counter to track the number of times the DFS algorithm had to move back. If the "backCnt" exceeded a certain threshold, the algorithm would skip the current case and move on to the next one. This allowed me to identify good cases more quickly. Next, I used the 2-opt algorithm to further optimize the best cases identified by the DFS + Greedy approach. This helped me find the best possible solution to the problem.

2-opt Algorithm : [2-opt - Wikipedia]

Flow chat :


Start:
Main:
Read file put in two data struct to save the graph: adjacency List and Matrix
sorting all node neighbor by distance
run the DFS using greedy choose the path
and find base good case
run a 2-opt algorithm find best case