John Zheng

jz5pt

4/25/19

postlab11

prelab complexity

For my prelab, the program first reads in the vertex from the file and that is O(n) because it depends on how many vertices are in the file. Then the topological function goes through the adjacency list. Since each element in the adjacency list is just the vertex and its neighboring nodes, we can assume that the number is irrelevant. Thus, the time complexity of this program is O(n). The space complexity is also dependent on the size of the file and how many vertices are in it. Thus, space complexity is also O(n)

inlab complexity

Since this program was implemented using brute force, and is dependent on the number of cities, the time complexity is O(n!). n! because of the number of permutations the program requires. The space complexity of this program is O(n). Each permutation generated is stored in the same variable and no additional variable was added.

Acceleration techniques

1. Nearest neighbor algorithm

    a. This algorithm simply just calculates the shortest path from the current city to the next city until the final city. From all of the acceleration techniques, this is by far the easiest to understand and would be very simple to implement.

2. Ant colony
    a. The name describes the algorithm pretty well. Imagine sending a colony of ants out on different paths and identify the one with the shortest path. The problem with this technique is that the shortest path is not guaranteed. However, with a big enough sample, we can be pretty sure that the result is close to the shortest path.

3. 2-opt
    a. This algorithm simply finds all the routes that cross each other and sort them in a way that they don't. This way the shortest path is guaranteed because traveling along the parameter is always shorter than going across each vertex.

https://pdfs.semanticscholar.org/080f/f5f7eea2cf18d13ab939fa5a2d7e0f12e234.pdf