Project 4

Wenjie Mo

905304148

Project 4 Report

Time complexity for each function:

1. StreetMap
   1. load()

My load function will go through each line and when it meets the line segment it will add the line segment (and its reverse) into a ExpandableHashMap. Each line of the street segment will be evaluated once. load() has time complexity O(N) if there are N lines in a mapdata file.

* 1. getSegmentsThatStartWith()

I used the find() function for search whether a certain segment is existing in the hash map and return all segments related with the segment, which has constant time complexity. Find() function has time complexity of O(1), so this function also has O(1) time complexity.

1. PointToPointRouter
   1. generatePointToPointRoute()

I used A\* algorithm in the function, which uses a STL Map map<GeoCoord, Node\*> (Node is a struct to store Geocoord, f, g, h value, and a parent pointer) for both Closelist and Openlist, since I could use the GeoCoord as the key for element searching. Closelist is used to store the Street segment I have already visited, Openlist is used to store the possible segment waiting to be explore. I did not choose to use unordered\_map because the hash function will be difficult to handle.

Retrieve an element given the GeoCoord will take log n time. The openlist will behave like a priority list to find the minimum f value, which may take n time for worst case using iterator. Although the big-O is beyond the scope of this class, I could assure that it is less than O(N) since each line segment will be met once and not every line will be visited with A\* algorithm.

1. DeliveryOptimizer
   1. optimizeDeliveryOrder()

I used simulated annealing algorithm for this function. I used two vector<DeliveryRequest>, which contains the current random delivery plan and the current optimum delivery plan.

The overall time complexity depends on my maximum temperature, annealing ratio, and chain length for searching. I decided to set my maximum temperature related to number of deliveries linearly: O(N) and the chain length related to the square of number of deliveries: O(N^2), I decided to make it N^2 because it will not exceed the N^4 time complexity which is the limit provided in the spec and also give more opportunity for route finding with larger number of deliveries so my result can be more accurate. Although the big-O is beyond the scope of this class, I could assure that it is less than O(N^4)