C950 Task-1 WGUPS Algorithm Overview

(Task-1: The planning phase of the WGUPS Routing Program)

Harrison Plummer
ID #010186537
WGU Email: hplumm6@wgu.edu
11/14/2023

C950 Data Structures and Algorithms II

A. Algorithm Identification

I plan on implementing a nearest neighbor algorithm. I'll use the special notes from the packages to load the first items onto the truck, and load the remaining packages based on the nearest neighbor algorithm. Then the nearest neighbor algorithm will be used to deliver all of the packages.

B. Data Structure Identification

I will use a hash table to store the package data for the algorithm.

B1. Explanation of Data Structure

The direct hash table will store the packages in key-value pairs. A package class will hold the package data. The key will be part of the package class (ID), and the value will be the item as a whole.

C1. Algorithm's Logic

```
FOR each truck in array of trucks
    WHILE truck is not empty
        GET current location from truck
        GET list of package addresses from truck
        SET shortest distance to -1
        FOR each package address in list of package addresses
            GET distance with current location and package address from distance csv
            IF distance is less than shortest distance OR shortest distance is -1
                SET shortest distance to distance
        ADD shortest distance to truck total miles
        SET delivered time to CALCULATE current package delivered time + shortest
        distance / truck speed
        SET current location on truck to nearest neighbor address
        REMOVE nearest neighbor package from truck
    ENDWHILE
ENDFOR
```

C2. Development Environment

I will be using PyCharm Community Edition with Python 3.11.5 on an M1 MacBook Air (2020).

C3. Space and Time complexity using Big-O notation

The direct hash table has O(1) search, insertion, and deletion (*Big-O Algorithm Complexity Cheat Sheet (Know Thy Complexities!) @Ericdrowell*, n.d.). I plan to use a loop to decide what package goes on which truck, which will have O(n) time complexity. The delivery nearest neighbor algorithm has three loops (truck, package, compare packages to find nearest neighbor). The outer truck and package loops will add up to the total number of packages plus the number of trucks (O(n + n/16) or O(n)) so the time complexity for all 3 loops will be O(n^2). The entire program's time complexity is O(n^2).

C4. Scalability and Adaptability

This solution will be able to easily adapt to any number of packages. The direct hash table will be set to an initial size of 40 but will be able to resize if a package with id greater than 40 is inserted. The nearest neighbor algorithm was chosen because it is self-adjusting and is therefore scalable. The one thing that would be difficult to scale is the special notes on packages. I am planning to have some cases in the truck loading part of the program check for special notes and load certain notes into certain trucks. This would not scale, and someone would have to manually check every special note.

C5. Software Efficiency and Maintainability

The program has a time complexity of O(n^2), keeping the program efficient. Maintainability would be relatively easy. The hash table resizes itself, to there is nothing to maintain there. The nearest neighbor algorithm also scales, so there shouldn't be any maintenance. As mentioned in C4, the truck loading algorithm will have some checks for special notes. This wouldn't be feasible to maintain at scale. What I would implement instead is that any packages with special notes go to a human (or some type of machine learning bot) for review. They would either manually assign it to a truck, or place an ignore on the note and send it through the algorithm again.

C6. Self-Adjusting Data Structures

The biggest benefit of direct hash tables is the speed of operations. All operations have O(1) time complexity. They are more complex to implement than a regular array, but make up for that with the previously mentioned speed. They do have a slight weakness in the form of space. Unless the direct hash table is exactly full, it will always be using space to hold empty buckets.

C7. Data Key

I chose the package id. In my data management classes I learned that a primary key must be non-null, and unique. It is also best if the simplest solution is chosen (i.e. a single field key, instead of a multiple field key). I followed those rules when choosing a key. The only item from the packages file that met all the criteria was the package id.

D. Sources

Big-O Algorithm Complexity Cheat Sheet (Know Thy Complexities!) @ericdrowell. (n.d.).

https://www.bigocheatsheet.com/