WGU C950 Data Structures and Algorithms 2

By: Andrew Vigil

Date: 1/21/23

A. Identify a named self-adjusting algorithm (e.g., nearest neighbor algorithm, greedy algorithm) that could be used to create your program to deliver the packages.

Nearest neighbor algorithm is an algorithm that could be used to create my program to deliver packages.

B. Identify a self-adjusting data structure that could be used with the algorithm identified in part A to store the package data.

A hash table is a self-adjusting data structure that could be used with the algorithm identified in part A to store the package data.

1. Explain how your data structure accounts for the relationship between the data components you are storing.

For my program, I opted for a hash table data structure, chosen for its ability to facilitate easy access and retrieval of data components. Specifically, I implemented a chaining hash table, a design that assigns keys to individual buckets. Unlike traditional setups where a bucket contains a single value, here, a linked list is employed, serving as the value associated with each key. This strategic use of linked lists prevents collisions, ensuring the program's stability and preventing potential crashes.

The chosen chaining hash table brings forth numerous advantages, including the seamless execution of search, removal, and insertion operations on data components, contributing to an overall reduction in data collisions. In the context of managing packages, the hash table is tailored to utilize the package ID as the key, while the associated value comprises a list encompassing other package details such as status, weight, and any special notes. By employing the package ID as the key and maintaining an array of related data components as the value, a cohesive association is established, with all pertinent information seamlessly stored in the hash table's buckets.

C. Write an overview of your program in which you do the following:
1. Explain the algorithm’s logic using pseudocode.

Nearest Neighbor Greedy Algorithm

```
PackagestoDeliver = (1, 2, 3, ..., 40)
While PackagestoDeliver > 0
Find next package to deliver
NextPackage = cycle trhough list looking for next closest address
    Add address to next_address and package to next_package.
After NextPackage is delivered
    Remove from PackagestoDeliver
MilesDriven = distance driven to each address
Time = MilesDriven/ 18MPH
```

2. Describe the programming environment you will use to create the Python application, including *both* the software and hardware you will use.

Starting with the hardware, the programming environment I will use is a 2020 MacBook Air running MacOS Ventura for the operating system.
The software environment is PyCharm 2023.3 and I will be writing the program in Python 3.11.4.

3. Evaluate the space-time complexity of *each* major segment of the program and the entire program using big-O notation.

Hash Table:
Add, Delete: O(N)
Retrieval: O(1)

Algorithm:
The algorithm I am using has a worst time complexity of O(N^2).

In my algorithm I have a for-loop that puts the packages from the hash table into the "not delivered" list which is running a O of N.

I then have a while loop for cycling through the "not delivered" (Big O of N) and another for-loop nested inside of it for deciding which address to go to next. (Big O of N) which means the total worse case for this program is O of N^2

4. Explain the capability of your solution to scale and adapt to a growing number of packages.

The data structure and algorithm should be able to take larger inputs and have no issues as they are self-adjusting.

5. Discuss why the software design would be efficient and easy to maintain.

The software design would be efficient and easy to maintain mostly because of the hash map and the algorithm I chose. You could load the packages differently or increase the number by quite a bit before you would have to make significant changes.

6. Describe *both* the strengths and weaknesses of the self-adjusting data structure (e.g., the hash table).

The main strengths of the hash map are the speed and the ease of retrieval, insertion, or deletion. And the main disadvantage is that it is inefficient when there are many collisions

7. Justify the choice of a key for efficient delivery management from the following components:

- **delivery address**
- **delivery deadline**
- **delivery city**
- **delivery zip code**
- **package ID**
- **package weight**
- **delivery status (i.e., at the hub, en route, or delivered), including the delivery time**

Out of all the components listed, the only one that is both unique, universal, and simple is the package ID – therefore, it is the most efficient. Different packages could have the same address, deadline, city, zip, weight, and status. This could lead to a disruption of the algorithm and cause conflicts like packages not getting to the right location, not being on time, or reducing the efficiency of the overall truck route.

D. Acknowledge sources, using in-text citations and references, for content that is quoted, paraphrased, or summarized.