C950 Task-1 WGUPS Algorithm Overview

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

Cade Fisher
WGU Email: cfis208@wgu.edu
Thursday, August 1, 2024

C950 Data Structures and Algorithms II

A. Algorithm Identification

The nearest neighbor algorithm, a type of greedy algorithm, will be used to determine the delivery routes for the packages. This algorithm is self-adjusting because it dynamically selects the nearest next location to minimize travel distance at each step.

B. Data Structure Identification

I've decided to go with a queue. It's a great fit for this task because queues are efficient at managing data in a first-in, first-out (FIFO) manner, which aligns well with the order of deliveries.

B1. Explanation of Data Structure

In the queue, packages are managed in the order received and sorted by a nearest neighbor algorithm. This allows efficient access to all data components, including address, city, state, zip, delivery deadline, and weight. By processing packages in a FIFO manner and optimizing the order, the queue ensures an organized and efficient delivery flow.

C1. Algorithm's Logic

```
// Initialize the path with the starting location
    CREATE path AS LIST
    ADD startLocation TO path

// Initialize the queue with package addresses
    CREATE unvisitedQueue AS QUEUE
    WHILE packageQueue IS NOT EMPTY
            DEQUEUE package FROM packageQueue
            ADD package.address TO unvisitedQueue
     ENDWHILE

// Set the current location to the start location
    SET currentLocation TO startLocation

// While there are unvisited addresses
    WHILE unvisitedQueue IS NOT EMPTY
    // Sort unvisited addresses by distance from the current location
            SET sortedAddresses TO sortByDistance(currentLocation, unvisitedQueue)

    // Select the closest address
            SET closestAddress TO sortedAddresses[0]

    // Add the closest address to the path
            ADD closestAddress TO path
```

```
        // Update the current location to the closest address
            SET currentLocation TO closestAddress

        // Remove the closest address from unvisited queue
            REMOVE closestAddress FROM unvisitedQueue
        ENDWHILE

    // After visiting all addresses, return to the starting location
        ADD startLocation TO route
```

C2. Development Environment

I am using PyCharm 2024.1 on a macOS system with an M1 chip for my development environment.
I am using Python version 3.12.3 for this project.

C3. Space and Time Complexity Using Big-O notation

1. Initializing the route and queue
    a. Space Complexity: O(n)
    b. Time Complexity: O(n)
2. Sorting addresses by distance
    a. Space Complexity: O(n)
    b. Time Complexity: O(n log n)
3. Selecting the closest address and updating the route
    a. Space Complexity: None
    b. Time Complexity: O(n)
4. Total Complexity
    a. Space Complexity: O(n)
    b. Time Complexity: $O(n) * O(n \log n) = O(n^2 \log n)$

C4. Scalability and Adaptability

The nearest neighbor algorithm scales by dynamically selecting the closest package, optimizing for the shortest travel distance at each step. The nearest neighbor is adaptable because it recalculates the next stop in real time based on the current location. However, as the number of packages increase, the time complexity O(n^2 log n) might become a limitation, leading to slower performance for large data sets.

The queue is efficient at managing data in a FIFO manner, ensuring each package is processed in the order it was received. The queue handles a growing number of packages smoothly as each enqueue and dequeue operation has O(1) time complexity.

C5. Software Efficiency and Maintainability

The software will be efficient, as the nearest neighbor algorithm and queue will handle packages with manageable time complexity. It will also be easy to maintain, with clear, well-organized code and helpful comments that will make it simple for others to understand, fix, and update.

C6. Self-Adjusting Data Structures

The queue efficiently manages data in a first-in, first-out manner, with quick enqueue and dequeue operations. However, it struggles with random access and can become less efficient for large datasets or complex updates, as it's designed for sequential processing.

C7. Data Key

I would use the package ID as the key for efficient delivery management. It is unique to each package, enabling quick and accurate retrieval and updates of details like delivery address, status, and weight.

D. Sources

Traveling Salesman Problem Visualizer. (n.d.). https://tspvis.com/

Lysecky, R., & Vahid, F. (n.d.). *C 950: Data Structures and Algorithms II*.

Big-O Algorithm Complexity Cheat Sheet (Know Thy Complexities!) @ericdrowell. (n.d.). https://www.bigocheatsheet.com/

Make School. (2021, May 4). *What is Pseudocode in Python? Day 21* [Video]. YouTube. https://www.youtube.com/watch?v=3WSrPxINIt8