Algorithm Overview

Dallin Reeves
Email: dallinreeves@gmail.com
14 September 2022

Introduction

The purpose of this program is to use Python in order to solve the traveling salesman problem. In this scenario, there are 40 packages that need to be delivered in the Salt Lake City area. This program defines a way to efficiently deliver those packages using the Nearest Neighbor Algorithm. Doing so delivers packages as quickly, and with as little travel, as possible.

A. Algorithm Identification

This program uses the Nearest Neighbor algorithm to deliver packages.

B1. Logic Comments

The following is pseudocode for the sorting and delivery algorithm that was used in this program.

Nearest Neighbor Algorithm all is completed within the truckDeliverPackages() method

truckDeliverPackages() takes one parameter: truck object

Instantiate empty list not_delivered that will hold all package IDs for undelivered packages

Enter a for loop to populate not_delivered list

    For each package in the specified truck (from parameter), append that package ID to not_delivered

Enter a while loop to determine optimal delivery procedure

    While the length of not_delivered > 0:

        nextAddress is set to 2000

        nextPackage is set to None

        For package in not_delivered:

            If the distance from the truck to the package <= nextAddress:

                nextAddress is set to the distance from the truck to the package

                nextPackage is set to package

            (This for loop ensures that the closest package to the truck will be set as the nextPackage)

        nextPackage's ID is appended to the packages in the truck

        nextPackage is removed from not_delivered

        nextAddress is added to truck mileage

        truck's address is set to nextPackage address

nextAddress / 18 is added to truck's time (average speed is 18 mph)

nextPackage's delivered time is set to truck's time

nextPackage's departure time is set to truck's departure time

The space-time complexity of this algorithm is $O(n^2)$.

B2. Development Environment

The program is written in Python 3.9.4 using PyCharm Community Edition 2022.2.1.

The hardware used to create this program is as follows:

Processor: Intel(R) Core(TM) i7-9700K CPU @ 3.60GHz

Installed RAM: 32.0 GB

System type: 64-bit operating system, x64-based processor

Operating System: Windows 11 Pro

B3. Space-Time and Big-O

The space-time complexity of the entire program is $O(n^2)$

truckDeliverPackages(): $O(n^2)$

main.py: $O(n^2)$

getAddress(): O(n)

updateStatus(): O(1)

loadPackageData(): $O(n^2)$

loadDistanceData(): O(n)

loadAddressData(): O(n)

getPackageIDs(): $O(n^2)$

getPackages(): O(1)

findDistance(): O(1)

insert(): O(n)

search(): O(n)

remove(): O(n)

B4. Scalability and Adaptability

Because this program utilizes a nearest neighbor algorithm, it will be able to scale and adapt well to varying sizes of deliveries. Whether there are 10 packages to deliver or over 100, the algorithm will be able to find the closest delivery address to the truck's current location. This current program only uses two delivery trucks, as there are only two drivers, but as more drivers and trucks become available, the program will be able to account for that as well.

B5. Software Efficiency and Maintainability

The object oriented style of programming used to develop this program ensures the software's efficiency. It is simple, and as such any future programmers that will work on expanding or maintaining the program will have an easy time understanding what each part does.

B6. Self-Adjusting Data Structures

The hash table used in this program self-adjusts in a way that will allow WGUPS to never run out of room when updating and adding more packages. One downside to using this data structure is that as the structure grows and gains more entries, the space-time complexity will also grow.

D1. Explanation of Data Structure

This program uses a chaining hash table. This data structure has the ability to add items, search for items, and delete items. Packages are inserted into this table and are looked up using their package ID number. With the package ID being used as the key, packages can then be searched for and deleted by using the ID. The nature of this data structure allows an increasing number of packages to be stored as well to accommodate future expansion of WGUPS.

G1. First Status Check

Time checked: 8:40 AM

```
What time would you like to see status of packages? Format: HH:MM:SS 08:40:00
Package #1      Status: En route to destination     Deadline: 10:30 AM
Package #2      Status: At Hub      Deadline: EOD
Package #3      Status: En route to destination     Deadline: EOD
Package #4      Status: At Hub      Deadline: EOD
Package #5      Status: At Hub      Deadline: EOD
Package #6      Status: En route to destination     Deadline: 10:30 AM
Package #7      Status: At Hub      Deadline: EOD
Package #8      Status: At Hub      Deadline: EOD
Package #9      Status: At Hub      Deadline: EOD
Package #10     Status: At Hub      Deadline: EOD
Package #11     Status: At Hub      Deadline: EOD
Package #12     Status: At Hub      Deadline: EOD
Package #13     Status: En route to destination     Deadline: 10:30 AM
Package #14     Status: Delivered at 8:06:20        Deadline: 10:30 AM
Package #15     Status: Delivered at 8:13:00        Deadline: 9:00 AM
Package #16     Status: Delivered at 8:13:00        Deadline: 10:30 AM
Package #17     Status: At Hub      Deadline: EOD
Package #18     Status: En route to destination     Deadline: EOD
Package #19     Status: Delivered at 8:39:20        Deadline: EOD
Package #20     Status: Delivered at 8:37:40        Deadline: 10:30 AM
Package #21     Status: At Hub      Deadline: EOD
Package #22     Status: At Hub      Deadline: EOD
Package #23     Status: At Hub      Deadline: EOD
Package #24     Status: At Hub      Deadline: EOD
Package #25     Status: Delivered at 8:24:20        Deadline: 10:30 AM
Package #26     Status: At Hub      Deadline: EOD
Package #27     Status: At Hub      Deadline: EOD
Package #28     Status: En route to destination     Deadline: EOD
Package #29     Status: En route to destination     Deadline: 10:30 AM
Package #30     Status: En route to destination     Deadline: 10:30 AM
Package #31     Status: En route to destination     Deadline: 10:30 AM
Package #32     Status: En route to destination     Deadline: EOD
Package #33     Status: En route to destination     Deadline: EOD
Package #34     Status: Delivered at 8:13:00        Deadline: 10:30 AM
Package #35     Status: En route to destination     Deadline: EOD
Package #36     Status: En route to destination     Deadline: EOD
Package #37     Status: En route to destination     Deadline: 10:30 AM
Package #38     Status: En route to destination     Deadline: EOD
Package #39     Status: En route to destination     Deadline: EOD
Package #40     Status: En route to destination     Deadline: 10:30 AM
```

G2. Second Status Check

Time checked: 9:40 AM

```
What time would you like to see status of packages? Format: HH:MM:SS 09:40:00
Package #1      Status: Delivered at 8:47:00        Deadline: 10:30 AM
Package #2      Status: At Hub      Deadline: EOD
Package #3      Status: Delivered at 9:37:20        Deadline: EOD
Package #4      Status: At Hub      Deadline: EOD
Package #5      Status: At Hub      Deadline: EOD
Package #6      Status: Delivered at 9:01:20        Deadline: 10:30 AM
Package #7      Status: At Hub      Deadline: EOD
Package #8      Status: At Hub      Deadline: EOD
Package #9      Status: At Hub      Deadline: EOD
Package #10     Status: At Hub      Deadline: EOD
Package #11     Status: At Hub      Deadline: EOD
Package #12     Status: At Hub      Deadline: EOD
Package #13     Status: Delivered at 9:21:20        Deadline: 10:30 AM
Package #14     Status: Delivered at 8:06:20        Deadline: 10:30 AM
Package #15     Status: Delivered at 8:13:00        Deadline: 9:00 AM
Package #16     Status: Delivered at 8:13:00        Deadline: 10:30 AM
Package #17     Status: At Hub      Deadline: EOD
Package #18     Status: En route to destination     Deadline: EOD
Package #19     Status: Delivered at 8:39:20        Deadline: EOD
Package #20     Status: Delivered at 8:37:40        Deadline: 10:30 AM
Package #21     Status: At Hub      Deadline: EOD
Package #22     Status: At Hub      Deadline: EOD
Package #23     Status: At Hub      Deadline: EOD
Package #24     Status: At Hub      Deadline: EOD
Package #25     Status: Delivered at 8:24:20        Deadline: 10:30 AM
Package #26     Status: At Hub      Deadline: EOD
Package #27     Status: At Hub      Deadline: EOD
Package #28     Status: Delivered at 8:44:20        Deadline: EOD
Package #29     Status: En route to destination     Deadline: 10:30 AM
Package #30     Status: Delivered at 9:35:20        Deadline: 10:30 AM
Package #31     Status: Delivered at 8:56:20        Deadline: 10:30 AM
Package #32     Status: Delivered at 8:56:20        Deadline: EOD
Package #33     Status: En route to destination     Deadline: EOD
Package #34     Status: Delivered at 8:13:00        Deadline: 10:30 AM
Package #35     Status: Delivered at 9:16:00        Deadline: EOD
Package #36     Status: Delivered at 9:06:40        Deadline: EOD
Package #37     Status: Delivered at 9:32:00        Deadline: 10:30 AM
Package #38     Status: Delivered at 9:32:00        Deadline: EOD
Package #39     Status: Delivered at 9:21:20        Deadline: EOD
Package #40     Status: Delivered at 8:50:40        Deadline: 10:30 AM
```

G3. Third Status Check

Time checked: 1:10 PM

```
What time would you like to see status of packages? Format: HH:MM:SS 13:10:00
Package #1      Status: Delivered at 8:47:00        Deadline: 10:30 AM
Package #2      Status: Delivered at 10:38:00       Deadline: EOD
Package #3      Status: Delivered at 9:37:20        Deadline: EOD
Package #4      Status: Delivered at 10:32:00       Deadline: EOD
Package #5      Status: Delivered at 10:58:40       Deadline: EOD
Package #6      Status: Delivered at 9:01:20        Deadline: 10:30 AM
Package #7      Status: Delivered at 10:43:20       Deadline: EOD
Package #8      Status: Delivered at 11:02:00       Deadline: EOD
Package #9      Status: Delivered at 10:58:40       Deadline: EOD
Package #10     Status: Delivered at 10:52:40       Deadline: EOD
Package #11     Status: Delivered at 12:21:20       Deadline: EOD
Package #12     Status: Delivered at 12:47:40       Deadline: EOD
Package #13     Status: Delivered at 9:21:20        Deadline: 10:30 AM
Package #14     Status: Delivered at 8:06:20        Deadline: 10:30 AM
Package #15     Status: Delivered at 8:13:00        Deadline: 9:00 AM
Package #16     Status: Delivered at 8:13:00        Deadline: 10:30 AM
Package #17     Status: Delivered at 11:33:20       Deadline: EOD
Package #18     Status: Delivered at 10:29:20       Deadline: EOD
Package #19     Status: Delivered at 8:39:20        Deadline: EOD
Package #20     Status: Delivered at 8:37:40        Deadline: 10:30 AM
Package #21     Status: Delivered at 10:26:40       Deadline: EOD
Package #22     Status: Delivered at 11:58:40       Deadline: EOD
Package #23     Status: Delivered at 12:22:40       Deadline: EOD
Package #24     Status: Delivered at 11:48:40       Deadline: EOD
Package #25     Status: Delivered at 8:24:20        Deadline: 10:30 AM
Package #26     Status: Delivered at 11:54:20       Deadline: EOD
Package #27     Status: Delivered at 11:18:00       Deadline: EOD
Package #28     Status: Delivered at 8:44:20        Deadline: EOD
Package #29     Status: Delivered at 9:58:40        Deadline: 10:30 AM
Package #30     Status: Delivered at 9:35:20        Deadline: 10:30 AM
Package #31     Status: Delivered at 8:56:20        Deadline: 10:30 AM
Package #32     Status: Delivered at 8:56:20        Deadline: EOD
Package #33     Status: Delivered at 9:53:20        Deadline: EOD
Package #34     Status: Delivered at 8:13:00        Deadline: 10:30 AM
Package #35     Status: Delivered at 9:16:00        Deadline: EOD
Package #36     Status: Delivered at 9:06:40        Deadline: EOD
Package #37     Status: Delivered at 9:32:00        Deadline: 10:30 AM
Package #38     Status: Delivered at 9:32:00        Deadline: EOD
Package #39     Status: Delivered at 9:21:20        Deadline: EOD
Package #40     Status: Delivered at 8:50:40        Deadline: 10:30 AM
OPTIONS:
```

H. Screenshots of Code Execution

Completion of code that includes final truck mileage (89.1 miles)

```
OPTIONS:
Type 'all' if you would like to see status of all packages
Type 'search' to look up a specific package
Type 'miles' if you would like to see total mileage of all trucks
Type 'quit' if you would like to quit the program
miles

Total miles driven: 89.10000000000001

OPTIONS:
Type 'all' if you would like to see status of all packages
Type 'search' to look up a specific package
Type 'miles' if you would like to see total mileage of all trucks
Type 'quit' if you would like to quit the program
quit

Process finished with exit code 0
```

I1. Strengths of Chosen Algorithm

The nearest neighbor algorithm was chosen for its many strengths. For one, it is very adaptable to change. No matter how many packages are on a given truck, the nearest neighbor algorithm used in this program will find the closest package to the truck's current location. Another strength is its simplicity. As the program will need to be maintained over the years by various programmers, the simple and easy to understand nature of the nearest neighbor algorithm will allow just about any programmer to jump right in and assist in maintenance.

I3. Other possible Algorithms

Two additional algorithms that could have been used that would also meet the requirements for this scenario are:

1. Tabu search
2. Ant colony optimization (ACO)

I3A. Algorithm Differences

Tabu search is a metaheuristic search method. It uses a local search procedure in order to find the optimal solution. Other local search methods will often get stuck on a solution where many

different solutions are equally optimal, even if they are not the best solution. For example, in the nearest neighbor algorithm that this program utilizes, several different package addresses could hypothetically be the same distance from the truck, leading the algorithm to sort of spin its tires and iterate more than it needs to. Tabu search differs from this by allowing the acceptance of "worse" solutions if no improved solution can be found, in order to get unstuck.

As the name implies, ant colony optimization (ACO) is based on the actions of an ant colony. This algorithm mimics how real ants will leave a trail of pheromones so that other ants can find the shortest path back home or to food. In ACO, virtual ants leave "pheromones" that track each edge of the path. As several of these virtual ants have tried out different paths, following ants will begin to hone in on the shortest and best route available as well. This method is more involved than the nearest neighbor algorithm used in this program, as the algorithm does not concern itself with previous possible solutions like the ACO, only the closest package.

J. Different Approach

If I were to attempt this project again, one thing I would do differently is set up a different method to load packages onto the trucks. This program currently utilizes manual package loading done within the code itself. This new modification would take any packages that need to be delivered and find an efficient way to load packages into trucks based on deadlines, special notes, and proximity to each other in order to assist in optimal delivery.

K1. Verification of Data Structure

The chaining hash table used in this program satisfies all requirements, allowing for the addition, search, and removal of packages to and from the hash table. As shown in the screenshots provided, all packages are delivered on time and with a total truck mileage of 89.1, satisfying all requirements.

K1A. Efficiency

The worst case scenario for space time complexity of this data structure is O(n). This means that the number of packages that need to be delivered will affect the time needed to complete the look-up function in a linear fashion.

K1B. Overhead

As the number of packages increases, the hash table will just simply grow in size. There is no real worry of "running out of room". This data structure is great at space optimization.

K1C. Implications

Changes to the number of trucks or cities would have minimal effect on the look-up time and space usage of this data structure. Neither trucks nor cities are held within this hash table. The only effect on look-up time and space usage would be the assumed increase in number of

packages stored within the hash table if more cities are being serviced and more trucks are being utilized.

K2. Other Data Structures

Two additional data structures that could have been used in place of the hash table that was used in this program are:

1. Stack
2. Queues

K2a. Data Structure Differences

Stacks are a last in, first out data structure. In other words, the last element placed in the stack will be the first one accessed. As long as we ensured that the next package to be delivered was the last package added to the stack, it could be popped off of the stack when delivered. Unlike the hash table used in this program, stacks can only add and remove elements from the “top” of the stack.

Queues are similar to stacks, but they are a first in, first out data structure. In order for this to work, we would need to do the opposite as with stacks. We would need to ensure that the first package to be delivered is the first one entered into the queue and is at index n-1 where the last package to be delivered is at index 0. Unlike the hash table used in this program, queues can only add elements to the “back” of the queue and remove them from the “front”. In our hash table, each element is accessible at all times, able to be searched, updated, and deleted.