Shiv Patel

Data Structure and Algorithms II – C950

8/20/2024

Task 2

F.1. **Nearest Neighbor Algorithms:**

* My solution uses the nearest neighbor algorithm to determine the most efficient route for the trucks. This helps to minimize the total distance traveled by trucks by making sure that next package selected is the closest to the truck’s current location. This overall reduce the mileage and delivery time.
* Another strength is that the code is organized into modular functions that handles

F.2. **Verification:**

* The use of the nearest neighbor algorithm make sure that the truck follows the efficient rout, and traveled minimum distance. Function of the solution allows user to check the status of any package at a given time, and the solution also have a user interface. The ‘truck\_info’ and total\_mileage\_by\_all function provides a summary of the truck with its return time after delivering all packages and also output the total mileage traveled by all truck combined. This meets the scenario’s requirement to provide a summary of trucks.

F.3. **Other Algorithms:**

* **Dijkstra’s Algorithm:** It finds the all possible paths from the starting point to all delivery points. And then it selects the shortest path. It consider the overall delivery route and finds the shortest possible way. Dijkstra algorithm, guarantees to find the shortest path whereas nearest neighbor algorithm may not always find the shortest path. On other hand, the Nearest Neighbor only looks at next possible location and makes decision step by step.
* **Genetic Algorithm:** This algorithm explores a wide range of delivery routes by generating multiple routes. This allows the algorithm to avoid getting trapped in minima and provides better solution over a period of time. This algorithm can produce different solution on different runs due to its nature unlike the nearest neighbor algorithm, which will always runs the same path for the same inputs.

**G.** If I were to approach this project again, I would consider the following improvements:

* **Package Prioritization:** I would include more detailed system that factor attribute like priority packages. Instead of treating all packages equally in terms of routing, the algorithm would prioritize deliveries that are listed in priority list. This would help in more efficient customer focused system.
* **User interface for Manual Adjustment:** I would develop more detailed interface that can edit and change delivery status and delivery schedule. This would include rerouting to another address, giving a priority to a package, changing delivery times based on user inputs.

**H.** The data based used here is Hash table, which s used to store and retrieve package details. It allows for fast access to the package data based on unique package ID which makes it ideal for this scenario where there are function like lookup and update etc.

**H.1. Other Data Structure:**

* **Binary Search Tree:** unlike a hash table, which has (O(1)) average access time, a BST provides logarithmic time access when it is balanced. A BST also maintains an ordered sequence of stored elements, which can be useful is the application requires sorted data. However, it can be less efficient in terms of access times if tree becomes unbalanced. BST could meet the requirements if ordered data are priority.
* **Linked List:** this could be useful if the solution on the database are involve in frequent insertion and deletion and are in sequential. The primary difference between linked list and hash Table is in the access time and the way to organize the data. A linked list does not provide quick access like hash table. It requires transversal through the list to find a element. However, linked list are more memory efficient. Additionally, linked list are better where data needs to be accessed sequentially or when the list needs to be modified frequently.