Travis Bittner

C950 Task 2

F.)

1. Describe two or more strengths of the algorithm used in the solution.

One major strength of the algorithm is that it prioritizes packages that have tighter deadlines and constraints. The algorithm assigns packages to the correct trucks and accounts for delays and changes in deliveries. Another strength of the algorithm is its scalability. The algorithm is designed to take a variable number of packages without any parameter adjustments.

1. Verify that the algorithm satisfies all of the requirements

The algorithm satisfies all of the requirements by ensuring that packages that need to be delivered before a deadline are delivered on time. Packages that belong on specific trucks are assigned correctly. Packages that need to be grouped together are loaded onto a single truck. It also ensures truck 3 doesn’t depart until truck 1 or 2 is finished, as well as making sure all packages are delivered. To ensure these conditions are met, I used a nearest neighbor algorithm that manages priority and constraint packages first. Once these packages are loaded and routes are added, I run an optimization algorithm to ensure the routes are ordered using nearest neighbor.

1. Identify two other named algorithms that are different than the algorithm in the solution and would meet all the requirements.

Two other named algorithms that would satisfy these requirements are Simulated Annealing and Ant Colony Optimization algorithms. Simulated Annealing works by finding the shortest path and using probability to determine the viability, as well as a “temperature” or cost. It works by repeatedly iterating until it finds an acceptable path. The Ant Colony Optimization algorithm works similarly and is also inspired by nature. It utilizes “pheromones” to determine best path and iterates many times to find the most traveled path by the “ants”. This in turn handles the routing for the shortest path given the available hubs/packages.  
  
Both algorithms are different than the nearest neighbor algorithm I used because they utilize iterations and in a sense learning to develop the best route. The nearest neighbor simply finds the next closest drop off point and adds those packages to the truck.

G.) Describe what you would do differently, other than the two algorithms identified in part F3, if you did this project again, including details of the modifications that would be made.

If I could do this again, I would try to utilize a SQLite database for easier data access and manipulation. Mapping everything to a graph after building it was a lot of extra work, but the map needed to be transformed into usable data. Having this relationship in a database and querying it as needed would’ve cut back on a lot implementation of helper functions for querying and connecting package and hub data.

H.) Verify that the data structure used in the solution meets *all* requirements in the scenario.

The data structures used in the solution meet the requirements because they were built with those requirements as constraints. No data structures used are dictionaries or from external libraries. The graph is a unique exception because it just maps nodes and edges using lists, it technically has a search() function but it just searches for the hub based on an address. The graph does not have an insert() function as it is a 2D list of {node\_idx, edge\_weight}. The hash table satisfies all requirements because it is dynamic, has a search function, and an insert function. It holds the packages and uses the package ID as the key.

Two other data structures that could satisfy these requirements could be a min heap and a double-ended queue. The min heap would satisfy these because it would allow for the root to be the smallest index or package id. Inserting and searching is done easily due to it being a binary tree. It will also heapify and balance itself to maintain efficiency. A double-ended queue could also be another data structure that satisfies these requirements, it can be searched and inserted into.