Optimization of Clustering of Locations and Routing Times

Jake, Jaden, Chaewon and Bobby

December 15, 2021

Abstract

The Food Bank of Western Massachusetts has been feeding the neighbors in need. Among many problems they have faced, we are tackling the problem of finding the the optimal clustering locations and the routing times according to it. In other words, we are aiming to minimize the routing time by finding the best clustering of locations.

1 Introduction

The goal of this exhibition is to discover optimal groupings of locations for deliveries of food from The Food Bank of Western Massachusetts. More specifically, our end goal is to minimize the amount of time deliveries take during the day. Currently, the organization assigns deliveries to drivers based on what's available and most pressing at the moment, which opens the door to inefficiency in grouping orders. In the worst of situations, people in need will be waiting increasingly long times for food that could have been delivered earlier with better decision-making.

As of now The Food Bank of Western Massachusetts is in need of finding a more efficient clustering of locations for their routes. The trucks are clustered inefficiently, if the premise of the problem is to be assumed. As such, even in the event that the optimal grouping is being used by the food bank already, creating a program to find such an optimal solution is a great way to ensure the organization can plan for future alterations to their fleet. For one, we must define that an optimal solution is that set of routes which is completed as fast as possible. Of importance to the project is that the food bank has five trucks, and each truck carries nine palettes. These palettes are delivered to mobile food banks and brown bag recipients by our trucks. What's important to note is that, in minimizing the time taken in routing, the program will inherently get food into

people's pantries faster.

2 Literature Review

First, We review the paper *'The Family of Vehicle Routing Problem'*. Something important about the first paper is its direct contextual relationship to our project - it lays the groundwork for the *'Capacitated Vehicle Rounding Problem'* (CVRP) for a set of vehicles with equal capacity. This, of course, is problematic for the purposes of our project, because the capacity of the new vehicle may differ from that of existing vehicles. Nevertheless, the concepts are relevant to our understanding of the task at hand. For instance, this section defines that vehicles must visit destinations represented on a graph $G = (V, E)$, where the vertices (V) are the locations of delivery or pickup and the edges (E) are the time it would take a truck to travel between two vertices. It also defines a unique food demand for each location, which models our situation well but does not account for the type of food delivered, which we may want in order to differentiate between Brown Bags and Mobile Food Banks. In this model, we can also adopt the idea of a central depot of return that is established, as our model will need to include that. Unfortunately, this model does not account for time constraints, which is unacceptable in a realistic world. Our trucks can only operate from 9 to 5 with a lunch break, so each truck must be back to the depot by the end of the shift. More so, the program does not account for the unloading time, which we will want to account for in our problem as it acts as an additional cost.

Secondly, we review *'The Vehicle Routing Problem with Time Windows'* which is the extension of the *Capacitated Vehicle Routing Problem*(CVRP). This second paper addresses the wrongs of the first - that being its exclusion of the fourth dimension: time windows. What this means is that the models in this section account for any windows of time in which an objective must be completed. This is important in addressing our issues from the last paragraph - by incorporating this time constraint, the program can account for the rigid 9-to-5 schedules of the drivers, and their lunch breaks and unloading times. We would make these windows hard, so as to not overwork the drivers and force them into overtime. In this model, though, the number of vehicles is unbounded, and the objective is to minimize the number of vehicles used. Of course, this is problematic for us - we have an exact number of vehicles we have access to, and minimizing the usage of these trucks would only put drivers out of work and food out of the hands of those who need it. The implicit

benefit that gas and labor expenditure would decrease is outweighed by the significant increase in the amount of time an objective takes. Seeing as the vehicles are sunk costs, not using one for any one day adds only the benefit of one day's salary and one day's gas expenses, which are not worth the health risk of recipients who need the food.

3 Model

3.1 Definitions

Here, we define variables and constants so that it's easier to follow along the integer program we will introduce.

We define our variables as follows

- x_{ij}: arc (i,j) is 1 if a vehicle goes from i to j,and it's 0 otherwise.
- u_j : accumulated demand already distributed by the vehicle when arriving at customer $j \in N$
- k : the integer number of routes required

We define our constants as follows

- let N be every location besides the depot
- let O be the start of every route, the depot
- let D be the end of every route, the depot again
- let V be every location, that is, $V = \{O\} \cup \{D\} \cup N$
- let A be arc set, $A = \{(i, j) \in V \times V : i \neq j\}$
- let c_{ij} be the cost of going from location i to location j in seconds
- let q_j be the required pallets of location j

3.2 Integer Program

We are trying to minimize the total routing times subject to the constraints. The routing time from i to j is dependent on the vehicle, so we'll call it c_{ij} Our integer program is:

$$min \sum_{(i,j)\ in\ A} c_{ij} * x_{ij}$$

Such that

$$\sum_{j \ \epsilon \ (N \cup D)} x_{ij} \ = \ 1 \qquad \forall \ i \ \epsilon \ N \tag{1}$$

$$\sum_{i \ \epsilon \ (O \cup N)} x_{ij} \ = \ 1 \qquad \forall \ j \ \epsilon \ N \tag{2}$$

$$\sum_{j \ \epsilon \ N} x_{Oj} \ = \ k \tag{3}$$

$$\sum_{i \ \epsilon \ N} x_{iD} = k \tag{4}$$

$$u_i - u_j + Q * x_{ij} \ \leq \ Q - q_j \qquad \forall \ i, j \ \epsilon \ N \ , i \neq j \tag{5}$$

$$q_i \leq u_i \leq Q \qquad \forall \ i \ \epsilon \ V \tag{6}$$

$$0 \leq k \leq |N| \tag{7}$$

3.3 Defining the Constraints

- Our objective function is that we minimize the time c that is taken to make all deliveries.
- For the first constraint (1), we ensure that every non-depot location is left exactly once, which means truck leaves from N to $N \cup D$. In other words, truck can leave to N or to D.
- For the second constraint (2), every non-depot location is entered exactly once, which means truck enters from $O \cup N$ to N. In other words, truck can enter from O or N.
- For the third constraint (3), the number of vehicle moving between the depot O and location j is exactly K for all i in N.
- For the fourth constraint (4), the number of vehicle moving between location i and the depot D is exactly K for all j in N.
- For the fifth constraint (5), for a taken route from location i to j, the carrying capacity of a vehicle plus the difference between the number of palettes had before i and the number of palettes had before j is less than or equal to the difference between the carrying capacity of the vehicle and the demand of location j in number of palettes.

- For the sixth constraint (6), accumulated demand already distributed by the vehicle when arriving at customer $i \in N$ is greater than or equal to demand of customer j and less than or equal to vehicle capacity for all i in V.
- For the seventh constraint (7), Number of runs, denoted as k, is greater than or equal to 0 and less than or equal to $|N|$.

3.4 Small Example

In small example, we are going to demonstrate our integer program in smaller scale so that it's easier to understand. Let's say we have three locations and depot labeled as L_1, L_2, L_3, and D. Diagram below will show where each location is located. Demands of each locations are as follows : $L_1 = 1$ pallet, $L_2 = 1$ pallet, and $L_3 = 2$ pallets. We have two trucks with capacity of 2 pallets per truck. There are one brown bag and two food banks recipients. Loading and unloading one pallet takes on average 12 minutes.

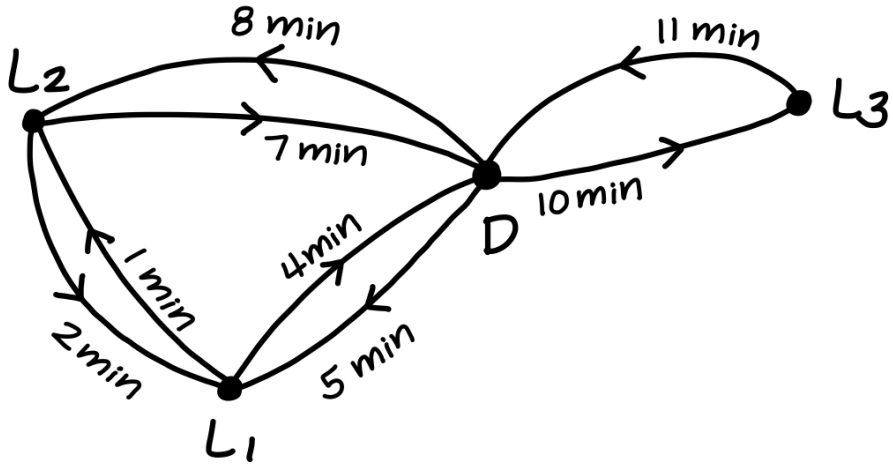

Figure 1: Diagram of the locations and routes

Below are the lists of time it takes for the truck to drive from one location to another in vector notation.

- $D\vec{L}_1 = 5$ minutes
- $L_2\vec{D} = 7$ minutes
- $L_1\vec{L}_2 = 1$ minutes
- $D\vec{L}_3 = 10$ minutes

- $L_3\vec{D} = 11$ minutes
- $L_2\vec{L}_1 = 2$ minutes
- $D\vec{L}_2 = 8$ minutes
- $L_1\vec{D} = 4$ minutes
- $L_1\vec{L}_3 = 45$ minutes
- $L_2\vec{L}_3 = 50$ minutes
- $L_3\vec{L}_1 = 45$ minutes
- $L_2\vec{L}_3 = 50$ minutes

According to the routing time listed, we can see that clustering L_1 and L_2 together minimizes routing time compared to clustering either L_1 or L_2 with L_3. Let's prove this by simple computation.

$D\vec{L}_1 + L_1\vec{L}_2 + L_2\vec{D} < D\vec{L}_3 + L_3\vec{D} + D\vec{L}_1 + L_1\vec{D}$

$\implies 5 + 1 + 7 < 10 + 11 + 5 + 4$

$\implies 13 < 30$

This shows that clustering L_1 and L_2 takes less time than clustering L_1 and L_3.

$D\vec{L}_1 + L_1\vec{L}_2 + L_2\vec{D} < D\vec{L}_3 + L_3\vec{D} + D\vec{L}_2 + L_2\vec{D}$

$\implies 5 + 1 + 7 < 10 + 11 + 8 + 7$

$\implies 13 < 36$

This shows that clustering L_1 and L_2 takes less time than clustering L_2 and L_3. Since We covered all cases, we prove clustering L_1 and L_2 is ideal.

Now we write an integer program according to small example.

$$min \sum_{(i,j)\ in\ A} c_{ij} * x_{ij}$$

Such that

$$\sum_{j\ \epsilon\ (N \cup D)} x_{ij} \ = \ 1 \qquad \forall\ i\ \epsilon\ N \tag{8}$$

$$\sum_{i\ \epsilon\ (O \cup N)} x_{ij} \ = \ 1 \qquad \forall\ j\ \epsilon\ N \tag{9}$$

$$\sum_{j\ \epsilon\ N} x_{Oj} \ = \ k \tag{10}$$

$$\sum_{i\ \epsilon\ N} x_{iD} = k \tag{11}$$

$$u_i - u_j + Q * x_{ij} \ \leq \ Q - q_j \qquad \forall\ i, j\ \epsilon\ N\ , i \neq j \tag{12}$$

$$q_i \leq u_i \leq Q \qquad \forall\ i\ \epsilon\ V \tag{13}$$

$$0 \leq k \leq 3 \tag{14}$$

Let's find the optimal solution. Let K_1 be the total time taken to load two pallets at the depot, drive to L_1, unload one pallet, drive from L_1 to L_2, unload one pallet, and drive back to the depot. This can be written as

$$\begin{aligned} K_1 &= 2 * 12 + D\vec{L}_1 + 12 + L_1\vec{L}_2 + 12 + L_2\vec{D} \\ &= 24 + 5 + 12 + 1 + 12 + 7 \\ &= 61 \end{aligned}$$

Let K_2 be the total time taken to load two pallets at the depot, drive to L_3, unload two pallets, and drive back to the depot. This can be written as

$$\begin{aligned} K_2 &= 2 * 12 + D\vec{L}_3 + 2 * 12 + L_3\vec{D} \\ &= 24 + 10 + 24 + 11 \\ &= 69 \end{aligned}$$

From the computation, we conclude that optimal solution is 69 minutes to deliver to all locations.

4 Results and Analysis

4.1 Overview of Our 3 Models

4.1.1 First Model

Through our optimizing process, we iterated through 3 different models before settling on the best. Our initial model used a k-index on the routing variables, which would allow us to specify timing constraints further down the line. However, using the k-index meant our full model had too many variables to run in a reasonable amount of time.

4.1.2 Second Model

Our second attempt was to run the k-index model on a smaller set of locations. We used a k-means model to produce the groupings of locations, which are pictured below.

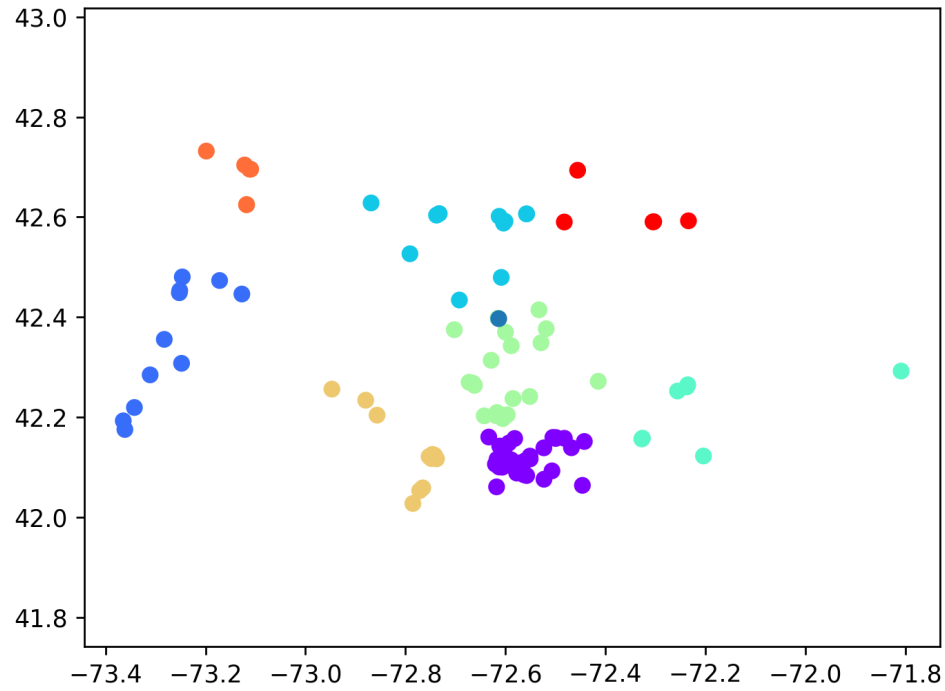

Figure 2: 8 Groupings Produced by K-Means

Unfortunately, the groupings produced by this K-Means were still too large for the integer program to quickly find solutions. We had to reduce the group size all the way down to groups of 10 locations for the integer program to run in a reasonable amount of time. However, the results produced could no longer truly be considered optimal since the integer program was very constrained by the small groupings. Running the integer program on small 10-sized groups meant that the integer program could no longer "see the bigger picture", and so the combination of all 13 groups into one solution did not make sense.

4.1.3 Third and Final Model

To remedy this, we shifted to a model which removed the k-index to reduce the number of variables to figure out. Due to this change, the model was able to run in a reasonable amount of time on the full 125 location dataset, and produce meaning full results which we can provide to the foodbank. These results are given and analyzed below.

4.2 Results from the Third Model

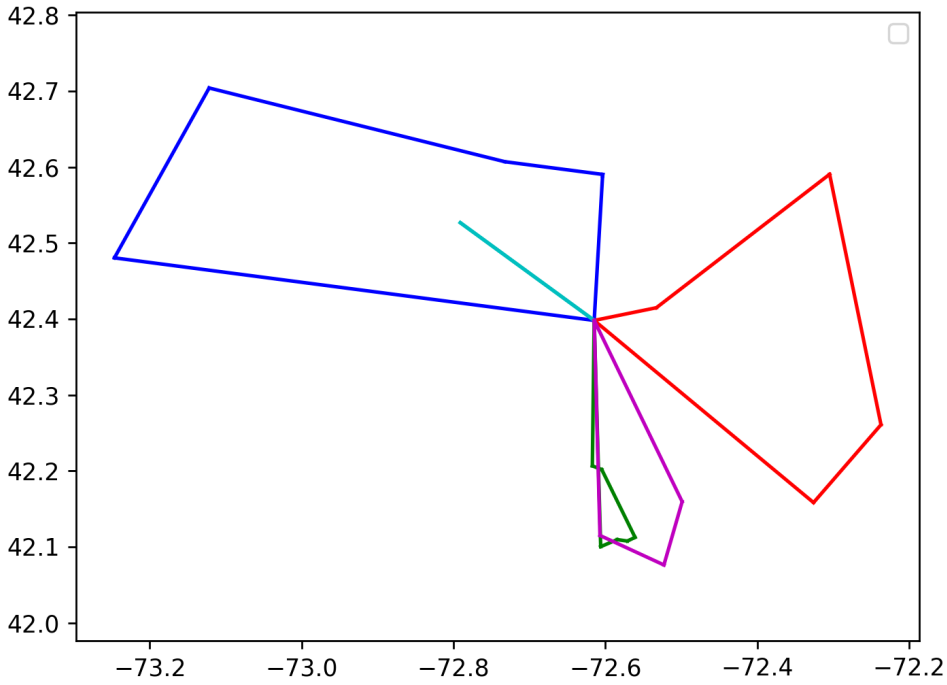

Figure 3: Integer program results for the first 20 locations

Pictured above is the optimal solution for running are integer program on the first 20 locations provided by the food bank. As you can see, the integer program was able to find a solution to visit these 20 locations using 5 routes, each pictured in a different color.

The results are consistent with what we would expect. Each location is touched by exactly one truck, except for the depot pictured in the middle. Stops are grouped by physical locations, with nearby locations typically being fulfilled by the same route. Additionally, no route is delivering over capacity, although routes are trying to get as close to capacity as possible to maximize the amount of food delivered. So, our integer program is working as intended.

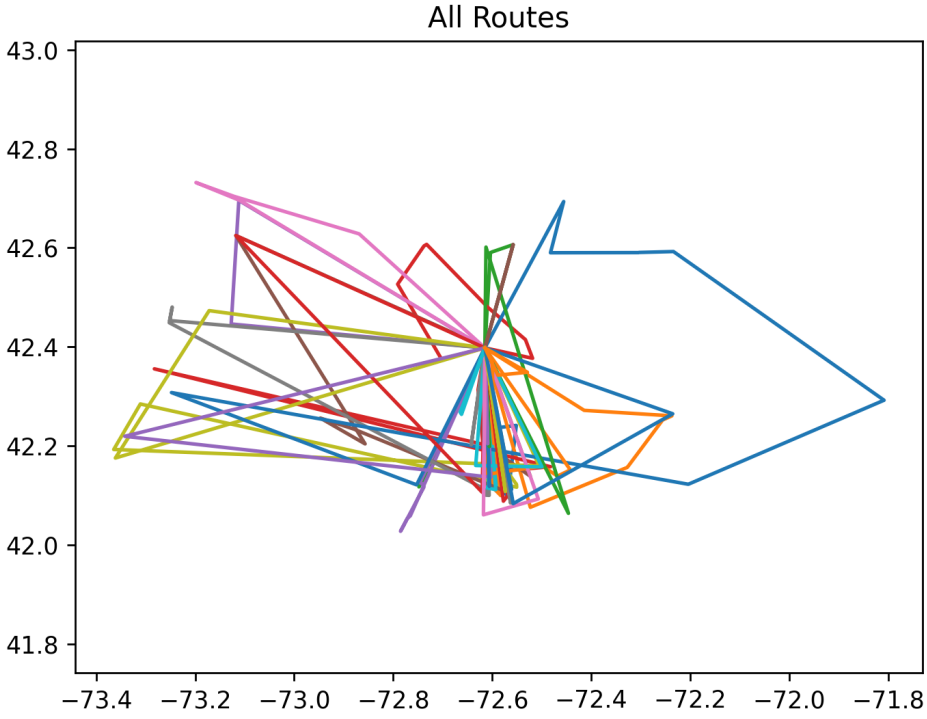

Figure 4: Integer program results for the full list of locations

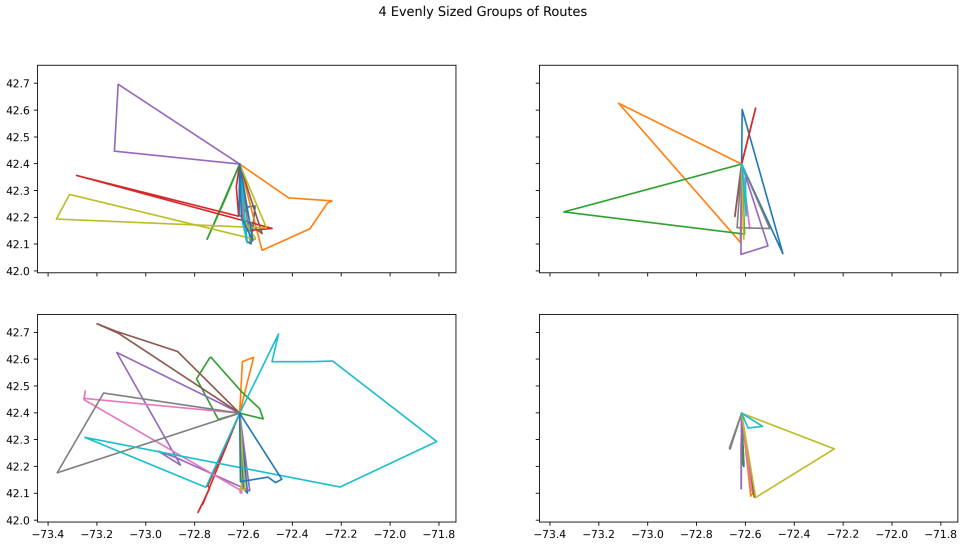

Figure 5: Routes above split into 4 groups for clarity

Above are the solutions produced by our integer program for the full list of 125 drop-off locations, though it should be noted that these may not actually be optimal as the integer program was

stopped with a 26.5% gap from proving optimality. The bottom figure depicts all the routes from the top figure in 4 evenly sized groups to provide more clarity in viewing the routes. Additionally, here are the routes in text form, with numbers corresponding to the "ids" given by the company.

```
Total routes: 42
Locations Satisfied | Pallets Deliverd | Path
1 | 09 | O -> 101 -> D
1 | 09 | O -> 104 -> D
1 | 09 | O -> 105 -> D
1 | 09 | O -> 111 -> D
1 | 09 | O -> 114 -> D
1 | 09 | O -> 117 -> D
1 | 09 | O -> 118 -> D
1 | 09 | O -> 119 -> D
1 | 09 | O -> 120 -> D
2 | 10 | O -> 21 -> 121 -> D
2 | 10 | O -> 25 -> 98 -> D
2 | 10 | O -> 31 -> 109 -> D
2 | 10 | O -> 32 -> 107 -> D
2 | 10 | O -> 37 -> 113 -> D
2 | 10 | O -> 41 -> 108 -> D
2 | 10 | O -> 43 -> 106 -> D
2 | 10 | O -> 76 -> 99 -> D
2 | 10 | O -> 95 -> 102 -> D
2 | 10 | O -> 97 -> 96 -> D
2 | 10 | O -> 100 -> 82 -> D
2 | 10 | O -> 103 -> 94 -> D
2 | 10 | O -> 110 -> 92 -> D
2 | 10 | O -> 112 -> 59 -> D
2 | 10 | O -> 115 -> 22 -> D
2 | 10 | O -> 116 -> 38 -> D
2 | 10 | O -> 122 -> 55 -> D
```

```
2 | 10 | 0 -> 123 -> 87 -> D
2 | 10 | 0 -> 124 -> 58 -> D
4 | 10 | 0 -> 35 -> 6 -> 40 -> 4 -> D
4 | 07 | 0 -> 44 -> 47 -> 7 -> 16 -> D
4 | 10 | 0 -> 72 -> 27 -> 8 -> 57 -> D
4 | 10 | 0 -> 78 -> 77 -> 34 -> 5 -> D
4 | 10 | 0 -> 93 -> 28 -> 11 -> 85 -> D
5 | 10 | 0 -> 24 -> 81 -> 80 -> 79 -> 50 -> D
6 | 09 | 0 -> 9 -> 26 -> 68 -> 67 -> 18 -> 15 -> D
6 | 10 | 0 -> 19 -> 86 -> 2 -> 88 -> 12 -> 51 -> D
6 | 10 | 0 -> 36 -> 84 -> 33 -> 83 -> 23 -> 75 -> D
6 | 10 | 0 -> 54 -> 53 -> 29 -> 52 -> 39 -> 30 -> D
6 | 10 | 0 -> 63 -> 71 -> 70 -> 65 -> 64 -> 69 -> D
6 | 10 | 0 -> 74 -> 13 -> 20 -> 73 -> 42 -> 14 -> D
7 | 10 | 0 -> 49 -> 1 -> 48 -> 17 -> 56 -> 10 -> 62 -> D
9 | 10 | 0 -> 91 -> 90 -> 89 -> 66 -> 61 -> 60 -> 3 -> 45 -> 46 -> D
```

The two graphs are automatically generated each time the integer program runs using matplotlib. The text output is also automatically generated on each run of the integer program.

As in the small example, these results are also logical in that they all start/end at the depot, reach near to capacity, and points along paths are generally grouped geographically. The value of the optimal solution was 202620 seconds with a 26.5% gap. This solution was produced after running for 3 hours, though 30% gap solutions were produced after just 5 minutes, making running this model on an evolving data set highly feasible. One unexpected result is the number of 1-stop routes produced by the integer program, which all correspond to drop-offs requiring 9 pallets. We would have expected the integer program to produce fewer routes that had an extra 3 pallets to spare, though it is possible that given more time to run the integer program may have further minimized the appearance of such routes.

4.3 Future Work

In the future, our second and third models could be combined to produce a very fast model that does not sacrifice much on optimality. Using K-Means hurt our second solution because we had to push the number of groupings to the extreme to reduce the run-time enough. However, our third model is able to find decent solutions in a reasonable amount of time. By using K-Means to produce a handful of groupings, say of size 40, we could potentially greatly reduce the run-time of our third model without sacrificing much on optimality.

5 Conclusion

Ultimately we decided that the third model (no k-indices) produces superior results to the second model (K-means clustering with k-indices). The routes produced by this integer program appear to require less time to complete than the company's current routing scheme. So, we recommend that the company uses the routes produced here wherever possible. We note that our integer program was unable to account for timing constraints, so we recommend that the company reach out to locations and ask if they are timing flexible. Then, they can use the integer program on the subset of locations that are timing-flexible to produce optimized routes. Because the integer program can achieve good results relatively fast, we hope that the company can use the integer program on an as needed basis with the necessary locations for the immediate future, thus easily accommodating for unforeseen problems by simply modifying their input data set.

References

[1] Madsen O. B. G. &; Ropke S. Desaulniers, G. *Chapter 5 : The vehicle routing problem with time windows*, p.119-159, 2014.

[2] Toth P. &; Vigo D. Irnich, S. *Chapter 1 : The family of vehicle routing problems*, p.1-33, 2014.