Optimization Final Report - Vehicle Routing Problem Revisit with Random Locations & Stochastic Travel Times

Qinwen (Katherine) Wang, Xinyao Han

Project Overview

With the growth of e-commerce and emerging online shopping platforms such as Amazon Prime, the demand for on-time, direct-to-customer deliveries has been ever increasing. However, customers are usually active at different locations during specific time windows on one day, and are therefore not able to receive packages directly at a fixed location all day long. For companies that would like to add more personalizations, it is important for them to give customers the option to choose the time and location they would like the goods to be handed. We view this problem as a vehicle routing problem, and specifically as a variant with random delivery locations and stochastic travel times. The project aim is to find optimized routes that can significantly maximize the efficiency of the fleet.

We use the same data of Augustin et al. (2018), which is a combination of realistic and simulated data, generated with Monte-Carlo methods and an enhanced greedy randomized adaptive search procedure (GRASP). The data includes customer related data: customer demands, coordinates for all the locations traveled by customers, a time window for each location, which represents the time the client stays at a location. We measure the travel time using distance, which is computed with coordinates for each of the two locations. It also includes general side information: total number of customers, locations, available vehicles, and capacity for each vehicle.

Specifically, one instance we use is a dataset with 15 customers, where deliveries can be made at 49 locations (some of them may be physically the same place, but at different times). The time horizon is defined to be 720 minutes, and each vehicle can carry a load of up to 750 demand units. We hope to solve for an optimized route for one vehicle to satisfy customer demands. We consider 5 vehicles.

Linear Programming Formulation

We follow the notation in Reyes et al. (2017) and made modifications upon implementation in Julia.

Location & Routing

The problem can be viewed as a directed graph $G = (V, A)$, where V is the set of locations traveled by customers and the location of the depot, A is the set of arcs that connect nodes. Each arc $a_{ij} \in A$ has a cost proportional to traveling time t_{ij}, which is also proportional to the distance between node i and j. We therefore calculate the cost for each arc using the coordinates. We assume that the cost between node i and j is the same as the cost between node j and i.

Let C be the list of all customers to serve, Z the set of vehicles (we will have $|Z|$ routes in total), and T represents the length of the planning horizon. Then each customer customer c ∈ C corresponds with a demand quantity d_c, and a subset of nodes $V_c \in V$ representing the locations visited by customer c during the planning horizon. We also make the assumption that different customers do not share any visited location in common. For mathematical formulation, $c \in C$ and $c' \in C/\{c\}$, $V_c \cap V_{c'} = 0$. To have a better understanding of the problem formulation, Figure 1 illustrates the vehicle routing problem with 4 customers. For example, customer 3 has traveled to 3 locations during the planning time, and customer 4 stays at the same location, possibly his or her house.

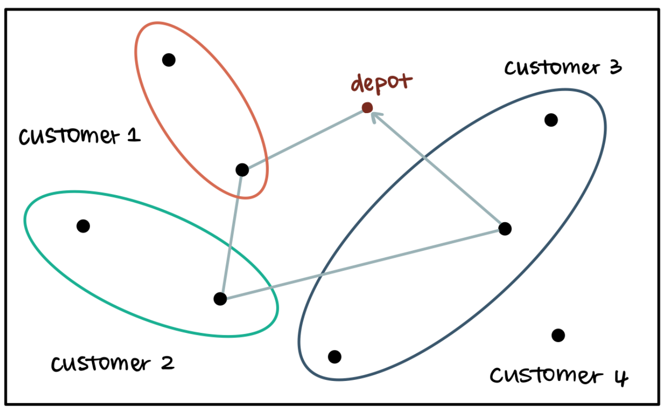

Figure 1: Sample route with 4 Customers and 8 Locations

Time Window

To allow customization in delivery time, we incorporate the time window of staying at each location for every customer. For mathematical formulation, each node $i \in V_c$ has a time window $[a_i^c, b_i^c]$. a_i^c represents the starting time of staying at location i for customer c, and b_i^c represents the time for customer c to leave location i. Therefore, the time window should not be overlapping. Consider customer c visited K locations during the planning time, then let i_L^c be the Lth location traveled by customer c, where $L \in [2, ..., K]$. We made the assumption that a customer start the day and end the day at the same location, therefore $i_1^c = i_K^c$, $\forall c \in C$.

Vehicle Delivery

For a feasible solution, we want to have a list of nodes or arcs indicating the route of a vehicle. We therefore defined a binary variable x_{ijk} where $x_{ijk} = 1$ if the vehicle travels from node i to node j for vehicle k and 0 otherwise. Each customer has a non-overlapping time window and a set of possible locations, each vehicle $z \in Z$ has a capacity constraint (We assume that all the vehicles have the same capacity Q).

Let τ_{ck} be the departure time for vehicle k from customer c (at any location), and y_z^c be the remaining capacity on vehicle z after service to customer c. As Figure 1 shown, we assume all vehicles should start from the depot, delivering to customers and then traveling back to depot. We use a dummy customer to represent the depot, where $c = \{1\}$, traveled to 0 locations $V_1 = \{0\}$, $\tau_1 = 0$, $d_1 = 0$, and $[a_1^1, b_1^1] = [0, T]$.

Based on the definitions above, we have the following formulations to solve the problem:

$$\min \quad \sum_{k \in Z} \sum_{\substack{i,j \in V \\ i \neq j}} t_{i,j} x_{i,j,k}$$

$$\text{s.t.} \quad \sum_{\substack{j \in V \\ i \neq j}} x_{i,j,k} = \sum_{\substack{j \in V \\ i \neq j}} x_{j,i,k} \qquad \forall i \in V \tag{1}$$

$$\sum_{k \in Z} \sum_{i \in V_c} \sum_{\substack{j \in V \\ i \neq j}} x_{i,j,k} = 1 \qquad \forall c \in C, c \neq 1 \tag{2}$$

$$\sum_{i \in V_c} a_i^c \sum_{\substack{j \in V \\ i \neq j}} x_{i,j,k} \leq \tau_{c,k} \leq \sum_{i \in V_c} b_i^c \sum_{\substack{j \in V \\ i \neq j}} x_{i,j,k} \qquad \forall k \in Z, c \in C, c \neq 1 \tag{3}$$

$$\tau_{c,k} + \sum_{i \in V_c} \sum_{j \in V_{c'}} t_{i,j} x_{i,j,k} \leq \tau_{c',k} + T\{1 - \sum_{i \in V_c} \sum_{j \in V_{c'}} x_{i,j,k}\} \qquad \forall c \in C, \quad \forall k \in Z, c' \in C, c' \neq c \tag{4}$$

$$y_z^c + Q\{1 - \sum_{i \in V_c} \sum_{j \in V_{c'}} x_{i,j}\} \geq d_{c'} + y_z^{c'} \qquad \forall c \in C, \quad \forall c' \in C, c' \neq c, \quad \forall z \in Z \tag{5}$$

$$\sum_{i \in V} x_{1,i,k} \leq |Z| \tag{6}$$

$$0 \leq y_z^c \leq Q - d_c \qquad \forall c \in C, \quad \forall z \in Z \tag{7}$$

$$\sum_{j \in V} x_{1,j,k} = 1 \qquad \forall k \in Z \tag{8}$$

$$\sum_{j \in V} x_{j,|V|,k} = 1 \qquad \forall k \in Z \tag{9}$$

$$x_{i,j,k} \in \{0,1\} \qquad \forall i,j \in V \tag{10}$$

$$\tau_{c,k} \in [0,T] \qquad \forall c \in C \tag{11}$$

$$y_z^c \in [0,Q] \qquad \forall c \in C, \quad \forall z \in Z \tag{12}$$

<u>Formulation Interpretation</u>

The objective is to minimize the total travel time of all the vehicles by serving all the customers.
Constraint (1) models that the number of vehicles arriving at a customer and leaving a customer for a location should remain the same for all customers.
Constraint (2) models that each customer c is visited by exactly one vehicle v.
Constraint (3) models that each customer c is visited by the vehicle v during his stay at a certain location (time window).

Constraint (4) models that the time vehicle v leaves the previous customer plus the transportation time from the previous customer to the next customer should be within the time window of the next customer.
Constraint (5) models that the amount of customers visited by a vehicle should satisfy the capacity constraints of this vehicle.

For both Constraints (4) and (5), we utilized integer optimization modeling. To be more specific, we'll take Constraint (5) as an example. If the vehicle drops at location i to deliver to customer c, and then travels to location j to deliver to customer c', then $x_{ijk} = 1, i \in V_{c}, j \in V_{c'}$ based on our formulation. And the constraint becomes $y_z^{c} \geq d_{c'} + y_z^{c'}$, making sure that the amount of capacity left in vehicle z after serving customer c should satisfy the demand for customer c'. If $x_{ijk} = 1$, then the constraint becomes $y_z^{c} + Q \geq d_{c'} + y_z^{c'}$, which ensures capacity constraint for vehicle z.

Constrain (6) models that the vehicles leaving the depot can't exceed the total number of vehicles.
Constrain (7) models the capacity limit when a vehicle visits and satisfies one customer.
Constrain (8) models that all vehicles leave from the depot.
Constrain (9) models that all the vehicles come back to the depot.
Constrain (10), (11), (12) are variables constraints on x_{ijk}, τ_{ck} and y_z^{c}. We constrain τ_{ck} to be less than the maximum travel time (planning horizon), y_z^{c} to be less than the maximum capacity for each vehicle.

Implementation & Results

Here's a snapshot of the customer schedule we use for implementation:

```
0  0  0  [0,720]
1 38  1  [0,720]
2 42  2  [0,65]   3 [212,213]   4 [354,370]   5 [531,542]  6 [637,720]
3 58  7  [0,87]   8 [140,236]   9 [297,325]  10 [338,720]
4 34  11 [0,2]   12 [332,334]  13 [664,720]
5 49  14 [0,20]  15 [256,310]  16 [475,480]  17 [621,720]
6 55  18 [0,27]  19 [133,150]  20 [391,440]  21 [617,670] 22 [672,720]
7 35  23 [0,720]
8 48  24 [0,720]
9 39  25 [0,35]  26 [50,78]    27 [145,289]  28 [472,479] 29 [603,625]  30 [650,720]
10 61 31 [0,69]  32 [312,349]  33 [592,720]
11 59 34 [0,70]  35 [83,166]   36 [341,342]  37 [506,549] 38 [652,720]
12 44 39 [0,116] 40 [178,257]  41 [320,401]  42 [417,616] 43 [633,720]
13 56 44 [0,720]
14 71 45 [0,64]  46 [263,279]  47 [497,612]  48 [666,720]
15 22 49 [0,720]
16 0  50 [0,720]
```

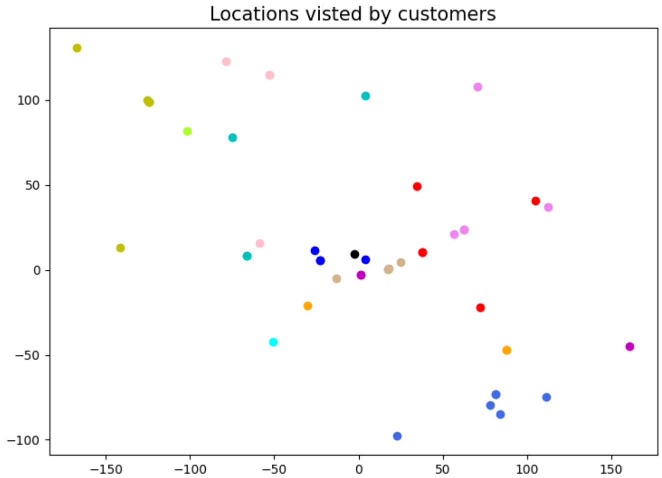

Figure 2: Locations visited by customers

The first column is the index for customers, and the second column is their corresponding demand. Each row has the index of locations traveled by each customer, and the time windows. We preprocessed the data into 5 different matrices.

- **Vc** is a 17 element vector, which defines the set of locations traveled by customer c & depot location (at index 1).
- **Kc** is a 17 element vector, which is the number of locations traveled by customer c.
- **aic** & **bi** are both 17×6 Matrix, which represents the starting & ending time of time window for customer C at location i respectively.
- **t** is a 51×51 Matrix, which is the cost between location i and j. We calculate the cost based on the Euclidean distance between location i and j, and use a 0.5 proportion.

We achieve the optimal cost of 2128 in 1406.62 seconds. Figure 3 illustrates the 4 optimized routes of delivery for 15 customers and 50 locations.

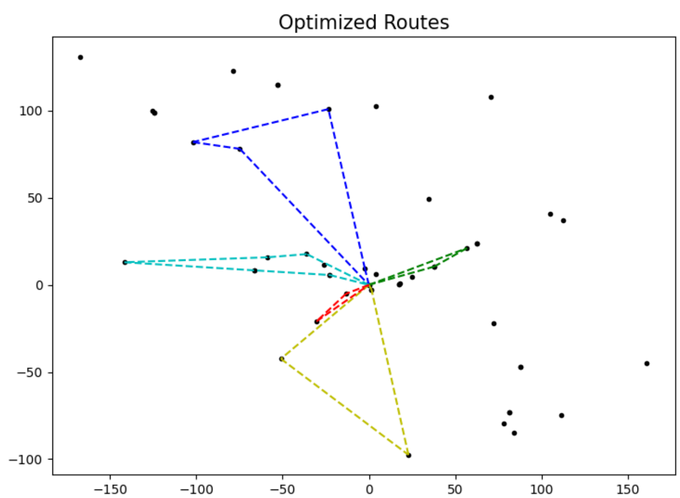

Figure 3: Optimized Routes Visualization

The common strategy of a traditional delivering company would be using 15 cars to deliver to each customer respectively. The solution generated by our model saves 587.63 (26.00%) compared to the mean, and 3181.87 (59.92%) compared to the worst case. Therefore, we would recommend companies to deploy the strategy for it not only saves costs, but also adds personalization options for customers, which can also generate additional revenue.

References

Augustin, L., Simon, T., Frédéric, F., "Vehicle Routing Problem with Roaming Delivery Locations and Stochastic Travel Times (VRPRDL-S)", Transportation Research Procedia, vol. 30, pp 167-177, 2018.

Reyes, D., Savelsbergh, M., Toriello, A., "Vehicle routing with roaming delivery locations," Transp. Res. Part C Emerg. Technol., vol. 80, pp. 71–91, 2017.