Route optimization for last mile delivery

Finding the best possible route between multiple touchpoints using AI optimization

Table of Contents

Executive Summary

In a product's journey from warehouse shelf to customer doorstep, the "last mile" of delivery is the final step of the process — the point at which the package finally arrives at the buyer's door. Besides customer satisfaction, last mile delivery is both the most expensive and time-consuming part of the shipping process.

Finding the minimum distance between two places is simple. But, when we extend this problem where we need to visit multiple places (say 500+), finding the best route can be complex. There can be additional complexities such as time windows of visit, customer preferences, the type of vehicles available and vehicles' capacity. These constraints make this a hard problem to solve in operational research. Vehicle routing problem falls under the class of a problem called NP-Hard (Computationally infeasible to find a solution and impossible to evaluate a given solution).

To elaborate this point, consider the following: There are 9.33e+157 (100 factorial) ways in which 100 points can be covered without considering any constraints. A system that evaluates 1000 routes per second will take 2.95E+147 years to complete all the combinations.

Thus, it is computationally impossible to try out all possible solutions and to determine the best possible solution. Heuristic/rule-based and AI algorithms are used to solve such problems. These methods do not promise the best possible solution but give a good solution over a reasonable, computational period.

1. Understanding the challenge

1.1. Route optimization for last mile delivery

Vehicle routing problem (VRP) is generalization of the famous Travelling Salesman Problem (TSP). The goal of VRP is to find an optimal set of routes for a fleet of vehicles to deliver goods to a set of locations. Each vehicle needs to return to a central depot after delivering the goods.

The last mile delivery problem is common across many business domains, like:

- A logistics company that delivers parcels to multiple customers in a city/region from a central location
- Large grocery stores, restaurants, fast food chains, and so on that offers home deliveries
- A warehouse supplying inventory to multiple retail outlets

Vehicle routing problems can have multiple objectives like minimizing the total distance, the total cost or the number of vehicles used while maximizing service quality. In this paper, we will consider reducing the total distance, however the proposed solution is flexible to other objectives as well.

1. Understanding the challenge

1.2 Types of vehicle routing problems

Based on the different constraints that are applicable to a delivery problem, one can classify the vehicle routing problem in different types. Some of the common variants are listed below:

- VRP: Vehicle routing problem without constraints
- CVRP: Vehicle routing problem with maximum capacity constraint of each vehicles
- VRPTW: Vehicle routing problem considering customers' promised time window
- CVRPTW: Vehicle routing problem considering customers' promised time window and maximum capacity of each vehicle

The problem we are trying to solve is of CVRPTW type: The demand must be fulfilled for multiple customers at their preferred time. After deliveries are completed, the vehicles must return to the depot within the working hours.

Keeping a real-world scenario in mind, we have added two additional constraints:

- Each customer has a different demand quantity and a preferred time window for accepting the delivery
- The limitation on the maximum load capacity of each vehicle i.e. a single vehicle cannot be used to deliver all the orders

2. Applying heuristic algorithms and AI-based optimization

2.1. A viable solution

We use a combination of heuristic algorithms and AI-based optimization to come up with a viable solution to the problem. Figure 2 is a pictorial overview of the solution.

Step 1
CVRPTW: A fleet of vehicles at a hub with a given capacity needs to serve multiple customers across different locations and abide by specific time windows while covering minimum distance.

Step 2
Sequence of deliveries: We employ a heuristic algorithm to find a possible sequence of deliveries which satisfy all the constraints. The sequencing provides an initial kickstart to the algorithm that can then be further optimized. Furthermore, it will help in reducing the converging time of the algorithm and provide an optimum solution in lesser time.

Step 3

AI-based optimization: The sequence of the customers proposed by the heuristic algorithm is optimized by using an AI-based optimization algorithm. The algorithm tries to minimize the total distance travelled as well as works under the demand and time constraints. Map box API is used for calculating the distance between different delivery points.

The algorithm considers each customer location as a 'node' and the route between two customer locations as an 'edge'. It builds a relation between two nodes by assigning probability to all the edges from this node. The probability assigned is inversely proportional to the distance of the edge. Based on these probabilities, the algorithm considers the next node that can be visited.

In each iteration, the probabilities achieved during the previous iteration assist in reaching more favorable solutions and to disregard the 'not so good' solutions. The probabilities are, therefore, refined incrementally and finally converge to their maximum value that represents the route of vehicles. At last, we apply a genetic algorithm to further optimize the proposed solution.

The constraint handling is done as follows:

- Capacity constraints: From each point, consider the next best point, based on the algorithm and check if the demand can be fulfilled with the available vehicle capacity.

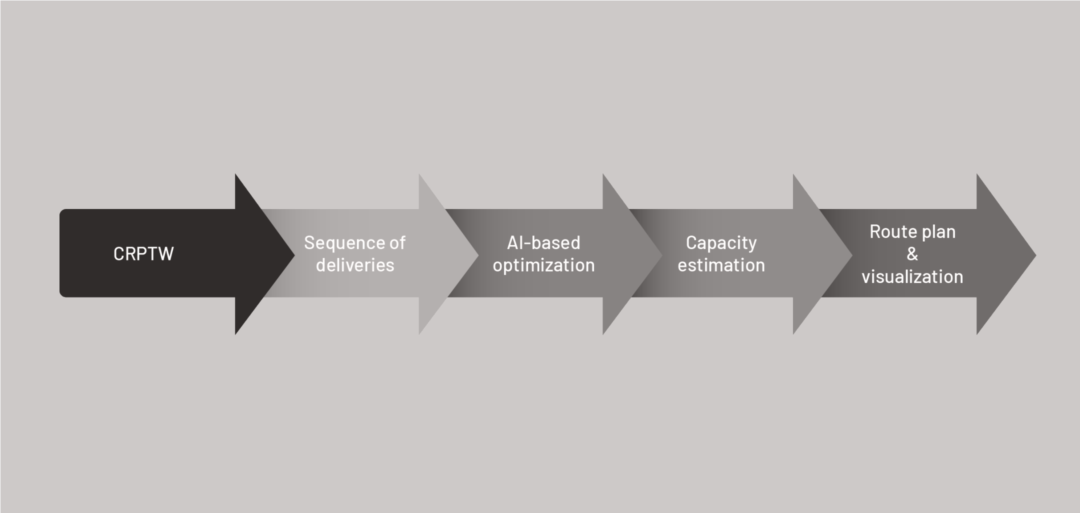
  If yes, then cover the customer; and if not, disregard the node and chose the next best node and repeat the process.

- Time window constraints: From each point, consider the next best point based on the algorithm and check if the vehicle can reach the point within the specified time window and can return back to the hub within the time limit of operations after serving the particular node.

  If yes, then cover the customer; and if not, then recompute the point.

- One can add more constraints as per the business context.

*Figure 2:
High level overview
of the solution*



Step 4
Capacity estimation: In an ideal scenario, we want to achieve a 100% service level. But we can also encounter some scenarios where we cannot fulfil all the customer demand on time with the resources currently available. In such cases, we need to find the minimum number of additional resources (vehicles, drivers, etc.) that will be required to achieve a 100% service level. This is calculated by the algorithm and the logistics team can plan accordingly.

Service level optimization: If, in the above case, we are not able to add additional resources all the demand cannot be served on time. In that case, the algorithm fulfills all the demand with the available vehicle with some delay in deliveries for few customers. This minimizes the total delay to achieve good service levels.

Step 5
Route plan & visualization: The finalized route can be made available as a download for easy execution. A visualization of the detailed route plan for every truck/delivery vehicle is also provided, to quickly understand the results.

2.2. Other considerations

Multi-hub scenarios: The above case describes a route plan for single hub serving multiple customers. We can imagine there can be multiple such hubs belonging to the same company serving multiple customers. So, while planning the route individually it is good to bring synergy and optimize the routes. For example, let us consider 2 hubs A and B. There could be a route that a vehicle paired with hub A could cover few nodes paired with hub B to minimize the overall distance.

The optimization is done in two stages:

- Compute best routes for each hub individually.
- Consider all the routes of all the hubs and swap edges between different nodes. Apply a genetic algorithm for this swapping process with an objective to minimize the overall distance.

Solution scalability: Combinatorial optimization problems by nature are computationally expensive. For quick algorithm convergence, it is necessary to introduce scalability. We use multi-threading concepts and just in time complier code to make the algorithm much faster. We achieved seven times the computational efficiency for a sufficiently large data set due to these techniques.

2.3. Understanding the solution with sample input data

As an input, we use two datasets: Customer data and Vehicle data, as shown in the samples displayed below in Figure 3 and Figure 4.

Figure 3:
Customer data

Customer No.	Longitude	Latitude	Demand Quantity	Time Window (24hr)	Time to service (min)
1	45.92	68.53	10	20:00-21:00	10
2	45.34	70.53	30	18:30-19:30	30
3	40.34	50.42	20	09:00-10:30	15
...	...	...	...	...	...

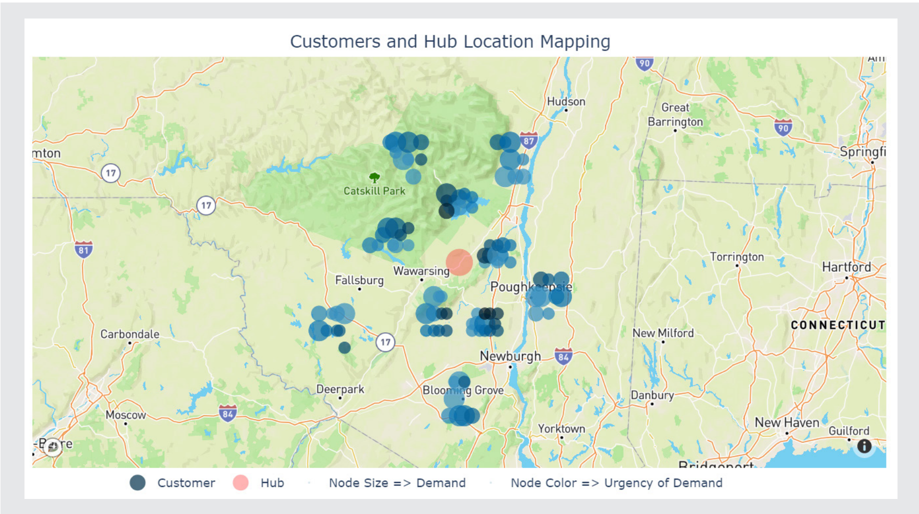
Figure 4:
Vehicle data

Vehicle No.	Capacity	Availability
1	200	Yes
2	100	Yes
3	150	No
...	...	...

Note: The Demand Quantity and Capacity are in same units, like the number of boxes, kilograms (weight), cubic meters (volume), etc.

Figure 5 is a plot that shows a graphical representation of delivery requirments. Each circle represents a customer at a specific longitude and latitude. The red circle represents the hub from where all goods are delivered. Each blue-shaded circle represents a customer. The customer demand is represented by the size of the circle and the urgency of delivery is represented by the circle shade.

Figure 5:
Proactive customer experience



2.4. Map and distance API

The three main APIs used from Mapbox (alternatively Google API can be used) to enable the solution include:

1. Distance Matrix API: Used to fetch distance and trip duration between two locations
2. Direction API: Used to fetch the route (for navigation) to reach from one location to the other
3. Map load for web: Used to display the real map in user interface

To get the traffic data (and therefore the duration of trip) at certain departure or arrival times, we need to use the API several times for a single algorithm run. To minimize the cost associated with API calls, we follow a few measures:

- Caching the API results for directions, distance, and time.
- Building regression models to predict trip duration.
- Building improvements on the algorithm to reduce the number of API calls without impacting the result

3. Evaluating the results with two comparative methods

To establish the efficiency of our AI-based algorithm, we tried to compare it with some heuristic methods. This is done because we cannot directly measure efficiency in any absolute terms, as the global best solution is unknown. Therefore, we resort to comparative methods to check the efficiency of our algorithms.

We have considered two heuristic approaches:

- Heuristic 1: Nearest neighbor first approach
- Heuristic 2: Density-based heuristic approach

Problem considered:
The demand of 100 customer nodes with total order of 1810 units were served by a single hub with 30 vehicles having the capacity of 200 units each.

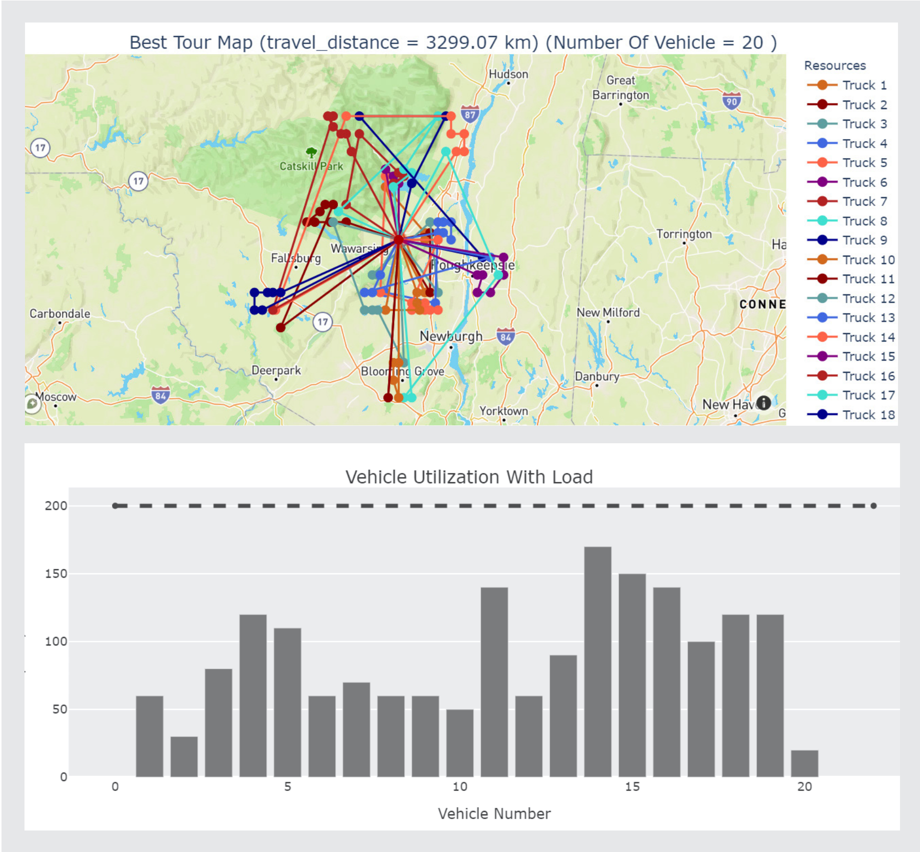
3.1. Heuristic 1: Nearest neighbor first approach

This algorithm uses the following approach:

- Find the nearest customer location from your current location (start from hub)
- Fulfil the demand of this location if constraints are adhered
- Repeat step 1 and 2 until all neighbors are exhausted
- Return to hub
- Repeat steps 1 to 4 until all the demands are fulfilled

The result obtained from this approach on the problem specified is represented in Figure 6:

Figure 6:
Nearest neighbor heuristic results



3.2. Heuristic 2 - Maximum Density First

1. Cluster the customers based on their location data.
2. Represent each customer location in terms of density. A high density indicates that the customer has several other customers near to their location.
3. Choose the cluster with the highest density and a node within this cluster that has the highest density
4. Fulfil the demand of this location if constraints are adhered
5. Follow the nearest neighbor heuristic from this node to cover other locations in the cluster
6. Repeat from step 3 until all customer demands are fulfilled
7. The result obtained from this approach on the problem specified is represented in Figure 7

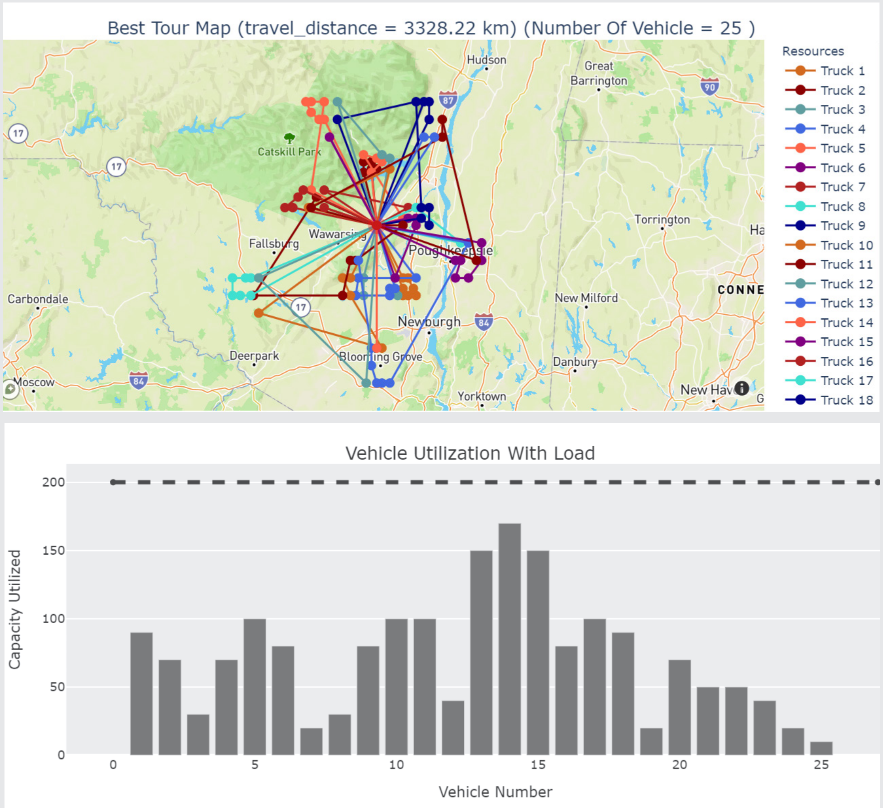


Figure 7:
Density-based heuristic results

3.3. Optimization Algorithm (Our Solution)

This approach uses the solution described in section 2. The result obtained from this approach on the problem specified is represented in Figure 8. The optimization algorithm also achieved 100% service level (criteria - on time in full).

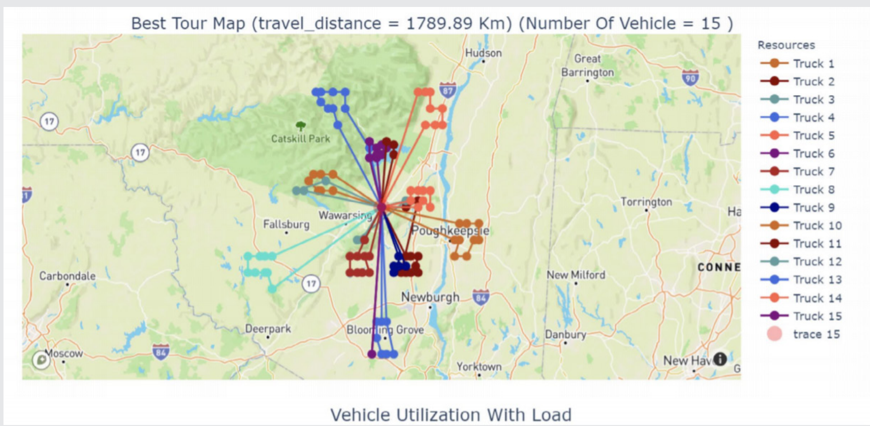
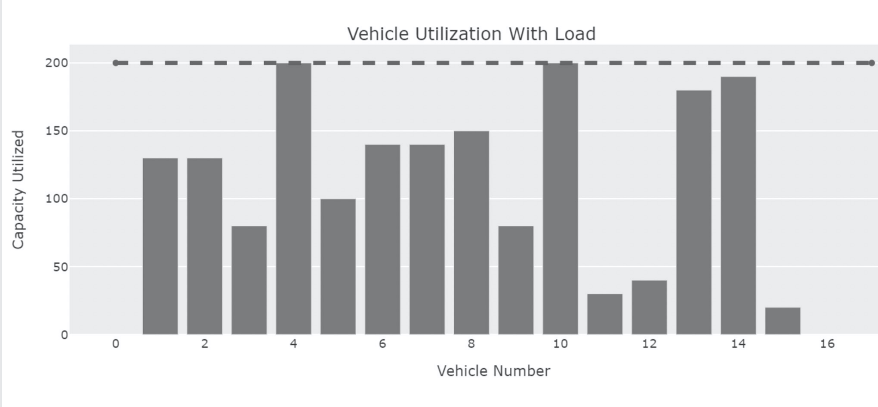


Figure 8:
Optimization Algorithm results (1/2)

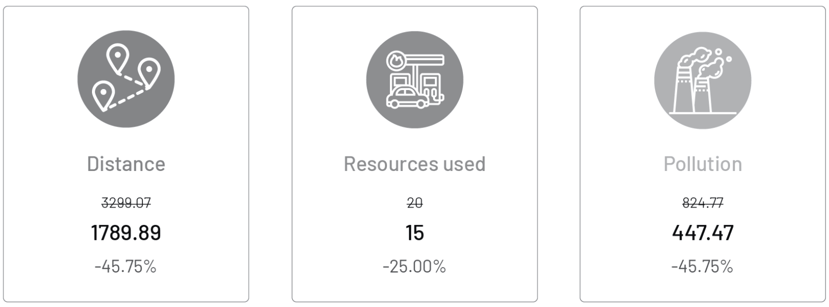
Figure 8:
Optimization Algorithm results (2/2)



Summary table
We see a significant improvement (refer to Figure 9) by using the optimization algorithm

Approach	Number of vehicles to fulfill demand	Total distance covered (km)
Nearest neighbor first	20	3299
Maximum density first	25	3328
AI-based optimization	15	1789

Figure 9:
Optimized use of resources (distance travelled, vehicle usage, and pollution indicators)



Conclusion

Benefits of AI-based optimization solution

The solution has been designed to realize the following benefits:

- Cost optimization with the reduction in the total travel distance with high service level
- Handling multiple types of vehicles to fulfill demand
- Distance and duration prediction between two location using AI model
- Optimum utilization of the resources and reduction of ecological footprint
- Quick route planning with a light weight tool with effective UI
- Flexible and scalable solution that can be adapted to any number of customers, vehicles, and time windows
- Customizable program that can be tailored to different busines needs or scenarios

Further Enhancements: Taking it one step further

The following features can also be added to the solution, if required:

- Including two different service types (pickup and delivery) in the problem, transforms it to the Pickup and Delivery Problem (PDPTW)
- Dynamic route planning with scope to add/remove customers or capacity based on real time events like reschedules by customer, vehicle unavailability due to faults, etc.
- Including new constraints in the model such as orders that can be transported by a special vehicle only, orders that cannot be transported through same vehicle, etc.
- Including feedback mechanism to the algorithm based on real events

About the author

Hariram G
Data Science and Analytics

Hariram has 7 years of experience in Data Science and analytics and is an expert in domains such as Pharma and FMCG. He holds a postgraduate degree in management and has keen interest in solving business problems using Data Science. At Nagarro, Hariram has worked in multiple functions such as supply chain, finance, and strategy.

Marketing team

Shailesh Dhaundiyal
Editor

Deeksha Mamtani
Content Marketer

Nitish Singh
Designer

About Nagarro
In a changing and evolving world, challenges are ever more unique and complex. Nagarro helps to transform, adapt, and build new ways into the future through a forward thinking, agile and caring mindset. We excel at digital product engineering and deliver on our promise of thinking breakthroughs. Today, we are 8,400 experts across 25 countries, forming a Nation of Nagarrians, ready to help our customers succeed. www.nagarro.com