IME639A - Analytics in Transport and Telecom

Project report

Vehicle Routing Problem with Time Windows

Course Instructor: **Dr. Faiz Hamid**

Group Members

1. Srajan Jain(190862)
2. Sparsh Sihotiya (190860)
3. Paramveer Singh Choudhary (200665)
4. Sangh Priya Gautam (200859)

Problem Description

The Vehicle Routing Problem with Time Windows (VRPTW) is a well-known logistics and transportation optimization problem. It entails calculating the best route for a group of vehicles to take in order to deliver goods or services to a group of customers within a particular time frame.

In this problem, a set of vehicles is offered, each with a specified beginning point and a given capacity. There is also a set of consumers, each with a known demand, a time frame in which they can be provided, and a defined location. The goal is to establish a set of routes for the vehicles that allows all customers to be visited, all demands to be met, and all time frames to be met while minimising the total distance travelled or the number of vehicles employed.

The challenge is exacerbated by the fact that the vehicles must visit customers within the time frames specified. If a vehicle arrives too early or too late, the customer may be unable to get the goods or services, which may result in a penalty. The time windows may also differ for each customer, adding to the problem's complexity.

Solving the VRPTW entails striking a balance between optimizing routes for the shortest distance traveled or the smallest number of vehicles

used and satisfying the customers' time frames. This is a difficult undertaking since the number of alternative solutions grows exponentially as the number of customers and automobiles grows.

The VRPTW offers a wide range of practical applications in logistics, transportation, and distribution. It's popular in areas including food delivery, courier services, and garbage removal. Finding efficient VRPTW solutions might result in significant cost savings and increased customer service.

IP Formulation

A fleet of vehicles, V, a set of customers, C, and a directed graph, Q, define the VRPTW. The fleet is typically considered homogeneous, meaning that all vehicles are identical. The graph consists of |C| + 2 vertices, where the customers are represented by the vertices 1, 2,..., and n and the depot by the vertices 0 ("the starting depot") and n + 1 ("the returning depot"). The set of all vertices, i.e., 0, 1,..., n+1, is denoted by N. The set of arcs, A, represents direct connections between the depot and customers and between customers. There are no arcs originating from vertex n+1 or terminating at vertex 0. Each arc (i,j), where $i \neq j$, a cost $c_{ij,}$ and a time t_{ij}, which may include service time at customer i, are associated.

Each vehicle has a capacity of q, while every customer has a demand of d_i. Each consumer i has a time window $[a_{i,} b_i]$, and vehicles must arrive before b_i. If it comes before the opening of the time window, it must wait until a_i to serve the customer. It is presumed that the time windows for both depots are identical to $[a_{o,} b_o]$, representing the scheduling horizon. The vehicles cannot depart the depot before a_o and must be returned by b_{n+1} at the latest.

q, a_i, b_i, d_i, and c_{ij} are assumed to be non-negative integers, while t_{ij} are assumed to be positive integers.

The model includes two decision variables, denoted by x and s. For each arc (i, j), where i ≠ j, i ≠ n+1, j ≠ 0, and each vehicle k, we define x_{ijk} as

$$x_{ijk} = \begin{cases} 1, \text{ if vehicle k drives directly from vertex i to vertex j ,} \\ 0, \text{ otherwise} \end{cases}$$

The decision variable s_{ik} is defined for each vertex i and vehicle k and represents the time at which vehicle k begins serving consumer i. If vehicle k does not service customer i, s_{ik} is meaningless, and its value is therefore considered irrelevant. We presume $a_o = 0$ and $s_{ok} = 0$ for all k, assuming $a_o = 0$.

The goal is to design a set of routes that minimize total cost, such that

- each customer is serviced exactly once
- every route originates at vertex 0 and ends at vertex n + 1, and
- the time windows of the customers and capacity constraints of the vehicles are observed.

$$\min \sum_{k \epsilon V} \sum_{i \epsilon N} \sum_{j \epsilon N} c_{ij} x_{ijk} \text{ s.t.,}$$

The above function minimizes the total travel cost.

$$\sum_{k \epsilon V} \sum_{j \epsilon N} x_{ijk} = 1 \qquad \forall\, i \epsilon C,$$

These constraints ensure that each customer is visited exactly once.

$$\sum_{i \epsilon C} d_i \sum_{j \epsilon N} x_{ijk} \leq q \qquad \forall\, k \epsilon V,$$

These constraints state that a vehicle can only be loaded up to its capacity.

$$\sum_{j \epsilon N} x_{0jk} = 1 \qquad \forall\, k \epsilon V,$$

$$\sum_{i \epsilon N} x_{ihk} - \sum_{j \epsilon N} x_{hjk} = 0 \qquad \forall\, h \epsilon C,\ \forall\, k \epsilon V,$$

$$\sum_{i \epsilon N} x_{i,n+1,k} = 1 \qquad \forall\, k \epsilon V,$$

The above three equations indicate that each vehicle must leave the depot 0; after a vehicle arrives at a customer it has to leave for another destination; and finally, all vehicles must arrive at the depot n + 1.

$$x_{ijk}(s_{ik} + t_{ij} - s_{jk}) \leq 0 \qquad \forall\, i, j \epsilon N,\ \forall\, k \epsilon V,$$

The above inequality establishes the relationship between the vehicle departure time from a customer and its immediate successor.

$$a_i \leq s_{ik} \leq b_i \qquad \forall\, i \epsilon N,\ \forall\, k \epsilon V,$$

Above equation affirms that the time windows are observed.

$$x_{ijk} \epsilon \{0,\ 1\}\ \forall\, i, j \epsilon N, \qquad \forall\, k \epsilon V,$$

Implementation

The code implementation can be found in the following two notebooks.

- Notebook 1
- Notebook 2

Data file description

For data, We have used one of the instances of the given ORTEC data, which can be found here. (K70 and K19 one)
Data files have various sections

1. Time Matrix - As described in the provided link, it is a matrix that indicates the time taken in travelling i^{th} to j^{th} station.
2. Time Windows - These are windows in which delivery is expected to happen.
3. Demands - Demand of a specific customer
4. Service Time
5. Capacity

Methods to Improve

The branch-and-bound technique is used to find the optimal algorithm. The branch-and-bound tree has three sets that correspond to its nodes: $F(\alpha)$, the set of fixed viable routes beginning and ending at the depot: $P(\alpha)$, the collection of partly built routes beginning at the depot, and $C(\alpha)$, the set of customers who cannot be served next on $P(\alpha)$.

To perform a branch, choose a client i that is neither prohibited (i.e., $iC(\alpha)$ not featured (i.e., $iF(\alpha) \cup P(\alpha)$). The allocation of routes and customers is used to decide where to set up branches. Then, two forks are produced: one where the partly built route $P(\alpha)$ is extended by i and another where i is not permitted as the next client on the route that is added to $C(\alpha)$. The lower limit of node an is calculated by selecting customer I as the one whose partial route $P(\alpha)$ was extended. Lower bounds on all possible solutions specified by $F(\alpha)$, $P(\alpha)$, and $C(\alpha)$ are computed using dynamic programming at each branch-and-bound node α.

We begin with the simplest possible node, the root ($F(\alpha) = \phi$, $C(\alpha) = \phi$ and $P(\alpha) =$ depot). For this problem, we build a directed network with vertices $v(i, q, k)$ for $i = 0, 1,..., n$; $q = 0, 1,..., Q$ and $k = 0, 1,..., m$, where n is the number of customers, m is the number of cars, and Q is the total of all consumer requests q_i. Therefore, each branch-and-bound node has a corresponding set of paths.

Each of the k routes in the set has a total load of q, and each of the k last-visited customers is located at a different vertex of the graph $\{1, 2,..., i\}$. The duration of the related paths will be used to define the arc lengths in the directed graph. Then the lower limit is the lowest of the lengths of the shortest routes from $v(0, 0, 0)$ to $v(n, Q, k)$. The created routes are not obligated to stop by any certain set of consumers. A lower bound is the minimum that is obtained. We make an effort to dynamically add one more customer to the set of k routes that have load q and last customers $\{1, 2,..., i\}$. There are two options here:

- Routes do not terminate at the home of customer $i + 1$. This causes a zero-length arc to be drawn from $v(i, q, k)$ to $v(i + 1, q, k)$. Keep in mind that a customer $i + 1$ who is not an endpoint may still participate in one of the other routes produced by the function $F(i, q)$ shown below.
- Customer $i + 1$ should be added to the route as the last customer carrying load q'. For all values of q', this produces an arc of length $F(i + 1, q')$ from $v(i, q, k)$ to $v(i + 1, q + q', k + 1)$. $F(i, q)$ is the shortest route that may be taken with a total load of q and a final client i. The paths that go to a certain vertex, denoted by $v(i, q, k)$, are those that result from computing $F(i, q)$ and extending the graph with arcs.

At each given node in the branch-and-bound tree, we may categorise the situation as either $P(\alpha) = \phi$ or $P(\alpha) \neq \phi$. If $P(\alpha) = \phi$, we simply modify the issue to account for the existing routes $\widehat{k}$, their combined load $\widehat{q}$, and the consumers $\widehat{I}$ served by these routes.

For $P(\alpha) \neq \phi$, precisely one of the paths in the lower bound extends $P(\alpha)$. (a). Let us now define the shortest length of this extension to be $F'(i, q)$. $F'(i, q)$ is computed in the same manner as $F(i, q)$. The same holds true for this problem: it can be simplified using $\widehat{k}$, $\widehat{q}$ and $\widehat{I}$. Now we've added arcs connecting $v(i, q, k)$ and $v'(i, q, k)$ to the directed graph.

- Zero-radius arcs connecting $v(i, q, k)$ and $v(i + 1, q, k)$, and $v'(i, q, k)$ and $v'(i + 1, q, k)$. These are the routes when the $(i + 1)^{th}$ customer is not used.
- For any value of q', there exists an arc of length $F(i + 1, q')$ connecting $v(i, q, k)$ and $v(i + 1, q + q', k + 1)$, and $v'(i, q, k)$ and $v'(i + 1, q + q', k + 1)$.

Conclusion

In conclusion, this study has offered a thorough overview of the Vehicle Routing Problem with Time Windows (VRPTW), including a problem description, mathematical formulation as an integer programming (IP) problem, and a heuristic approach to identifying feasible solutions.

The VRPTW problem, which includes identifying the optimal routing of a group of vehicles to deliver goods or services to a set customer within a certain time window, was initially detailed in the project. The goal is to design a set of routes for the vehicles that minimises the total distance travelled or the number of vehicles used while meeting the customers' time constraints.

Following that, the problem was formalised as an IP problem, which entails establishing decision variables, objective functions, and constraints. The IP model was found to be effective for small and medium-sized instances of the VRPTW, but due to the exponential rise of viable solutions, it may be computationally intractable for bigger cases.

To circumvent this constraint, a heuristic technique to discovering feasible solutions to the VRPTW problem was proposed. For large-scale issues, the proposed heuristic method was proved to be effective in finding good quality answers in a reasonable amount of time.

Overall, this project proved the importance of the VRPTW problem in logistics and transportation, as well as how IP and heuristic approaches may be employed to address this difficult problem. Future study in this field could concentrate on establishing more efficient and effective ways for solving the VRPTW, as well as translating these approaches to real-world applications in a variety of industries.

References

- https://euro-neurips-vrp-2022.challenges.ortec.com/

- Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods [Nasser A. El-Sherbeny]
- Column Generation (Gerad 25th Anniversary) 2005th Edition by Guy Desaulniers (Editor), Jacques Desrosiers (Editor), Marius M. Solomon (Editor)
- http://dimacs.rutgers.edu/programs/challenge/vrp/vrptw/
- https://www.shutterstock.com/
- https://developers.google.com/optimization/routing/tsp