The Vehicle Routing problem

Given a set of vehicles with a certain capacity located at a depot and a set of customers with different demands at various locations, the vehicle routing problem (VRP) is how to satisfy the demand of the customers in the cheapest way possible. This is done by making a vehicle serve a subset of the customers, i.e. travel a route from the depot, to a number of customers and back to the depot. The accumulated demand of the customers cannot exceed the capacity of the vehicle. When routes have been created so that all customers are served, the problem has a feasible solution.

The set covering formulation

The VRP can be formulated as a set covering (SC) problem.

VRP_{SC}:

\[ H_j = \text{a feasible route } j \text{ with the optimal cost } c_j, j = 1, \ldots, M \]

\[ a_{ij} = 1 \text{ if customer } i \text{ is served in route } j, 0 \text{ if not} \]

\[ x_j = 1 \text{ if route } j \text{ is in the optimal solution, 0 if not} \]

\[ \min \sum_{j=1}^{M} c_j x_j \]

s.t. \[ \sum_{j=1}^{M} a_{ij} x_j \geq 1 \quad i = 1, \ldots, n \]

\[ x_j = 0 / 1 \]

If the cost matrix satisfies the triangle inequality, the SC formulation can be transformed to an equivalent set partitioning (SP) formulation by changing the \( \geq \) sign to an = sign. Since a shorter path never can be found by visiting the same customer twice, the optimal solution will be the same in both cases.

The formulation is very general but can easily be extended with other constraints, such as time windows. Most of these constraints will be considered in the problem of finding feasible routes to the VRP_{SC}. As the number of feasible routes can be very large, a column generation approach is necessary, i.e. the VRP_{SC} is solved for a small subset of routes and a new feasible route with a low reduced cost is found in a sub problem and added to the VRP_{SC}. The reduced cost for the columns are given by:

\[ c_r = \min_{j=1 \ldots M} \left\{ c_j - \sum_{i=1}^{n} \pi_i a_{ij} \right\} \]

where \( \pi_i = \text{the dual variables from VRP}_{SC} \)
SUB:

\(d_i = \text{the demand of customer } i\)

\(C = \text{the capacity of a vehicle}\)

\(y_i = 1 \text{ if customer } i \text{ is served in the optimal route, } 0 \text{ if not}\)

\[
\min f(y) - \sum_{i=1}^{n} \pi_i y_i,
\]

\[
s.t. \quad \sum_{i=1}^{n} d_i y_i \leq C
\]

\[
y_i = 0 / 1
\]

\(f(y) = c_j\) is the optimal value for the route consisting of the customers in the optimal solution and can be determined by solving the Travelling Salesman Problem (TSP) for these customers along with the depot. This means that the SUB problem is non-linear and quite hard to solve. Fortunately it is not necessary to solve the SUB problem to optimum. Any column with a negative reduced cost can be used in the VRPSC. One way is to approximate \(f(y) = p^T y\), and reduce SUB to a knapsack problem.

Often the linear relaxation of the VRPSC is solved first and then a feasible solution is found by using a branch-and-bound algorithm on the fractional solutions. It can be shown that under certain general circumstances, when the number of customers increases the value of the optimal fractional solution \(Z^{LP}\) is close to the value of the optimal integer solution \(Z^*\):

\[
\lim_{n \to \infty} \frac{Z^{LP}}{n} = \lim_{n \to \infty} \frac{Z^*}{n}
\]

This means that the linear relaxation of the SP formulation of the VRP gives a good lower bound to the integer problem, which implies that the branch-and-bound phase should be able to find the optimal solution quite fast.

**VRP and cuts**

**Formulation**

Let \(G = (V, E)\) be an undirected and complete graph with node set \(V\) containing \(n+1\) nodes numbered 0,1,2,...,\(n\), where 0 is the depot and 1 through \(n\) are the customers. \(V_0\) will denote the set of customers \(V = V_0 \cup \{0\}\). The demand at the \(i\)th customer is called \(d_i\) and the cost of using an edge \(e \in E\) is called \(c_e\). The number of vehicles used is \(k\) and the capacities of the vehicles are identical and denoted \(C\). For given subsets of nodes \(S, S' \subseteq V\), we define

\[
\delta(S) = \{e = (i, j) \in E : i \in S, j \in V - S\}
\]

\[
(S : S') = \{e = (i, j) \in E : i \in S, j \in S'\}
\]

\[
\gamma(S) = \{e = (i, j) \in E : i \in S, j \in S\}
\]
The VRP formulated as a linear integer program:

\[
\begin{align*}
\text{minimize} & \quad \sum_{e \in E} c_e x_e \\
\text{subject to} & \quad \sum_{e \in \delta(i)} x_e = 2 \quad \forall i \in V_0 \quad c_1 \\
& \quad \sum_{e \in \delta(0)} x_e = 2k \quad c_2 \\
& \quad \sum_{e \in \delta(S)} x_e \geq 2\left[\frac{d(S)}{C}\right] \quad \forall S \subseteq V_0, S \neq \emptyset \quad c_3 \\
& \quad 0 \leq x_e \leq 1 \quad \forall e \in \gamma(V_0) \quad c_4 \\
& \quad 0 \leq x_e \leq 2 \quad \forall e \in \delta(\{0\}) \quad c_5 \\
\end{align*}
\]

\(x_e\) integer \(\forall e \in E\)

Here, the variable \(x_e\), associated to each edge \(e\) of \(E\), represents the number of times edge \(e\) is used by the solution. We also have \(d(S)\) for the total demand in set \(S\) and \(D_T\) for the total demand of all customers.

The first constraints \((c_1)\) assure that each customer is visited exactly once. \(c_2\) states that each of the \(k\) vehicles has to leave and go back to the depot. Constraints \(c_3\) are the capacity constraints (also called generalised subtour elimination constraints). Their validity becomes clear noting that, for a given subset \(S\) of customers, at least \(\left\lceil\frac{d(S)}{C}\right\rceil\) vehicles are needed to satisfy the demand in \(S\) and, since the depot is outside \(S\), each of these vehicles must necessarily enter and leave \(S\).

Since all constraints to VRP are linear and all \(x_e\) are bounded, the solution space to the problem is a bounded polyhedron. This means that the convex hull of the feasible solutions can be described by a finite number of linear inequalities. Hence, theoretically, it would be possible to solve the VRP as an LP problem. In practice, however, this is not possible since

1. All inequalities describing the convex hull are not known.
2. The number of such inequalities is a very large number.

The capacity separation problem

Given a solution satisfying constraints \(c_1, c_2, c_4, c_5\) and the integer constraint, find a generalised subtour elimination constraint, \(c_3\), violated by \(x\) or prove that none exists.

Given a subset \(S \subseteq V_0, S \neq \emptyset\), and an LP solution \(x\), define

\[
f(S) = \sum_{e \in \delta(S)} x_e - 2\left[\frac{d(S)}{C}\right]
\]

Then, the capacity constraints, \(c_3\), corresponding to \(S\) is only violated when \(f(S) < 0\). On the other hand, if \(f(S) \geq 0\) for all \(S \subseteq V_0, S \neq \emptyset\), then no capacity constraint is violated by \(x\). This means that the capacity separation problem is equivalent to compute

\[
\min \{ f(S) : S \subseteq V_0, S \neq \emptyset \}
\]

This is a very hard problem to solve, and it is usually done by heuristics.
Generalised capacity inequalities

Let $\Omega = \{S_i, i \in I\}$ be a partition of $V_0$ and let $I' \subseteq I$ denote the set of indices of those subsets $S_i$ with more than one element. Let us now define $r(\Omega)$ as the solution to the bin packing problem where the capacity of the bins are $C$ and the sets of items and their weights are defined as follows

- If $d(S_i) \leq C$, then include an item of weight $d(S_i)$
- Otherwise, include $\lceil (d(S_i)/C) \rceil - 1$ items of weight $C$ and one more item of weight $d(S_i) - (\lceil (d(S_i)/C) \rceil - 1)C$.

If this is done, it can be shown that the following equality is valid

$$\sum_{\in I'} \sum_{x \in S_i} x_c \geq 2 \sum_{\in I'} \left\lfloor \frac{d(S_i)}{C} \right\rfloor + 2(r(\Omega) - k)$$

It is also possible to apply this idea to only a subset $H$ of $V_0$. If $H \subseteq V_0$ and $\Omega = \{S_i, i \in I\}$ be a partition of $H$ and let $I'$ be defined as before. If $r(H, \Omega)$ is the solution to the bin packing problem on $\Omega$ we have the bin inequality

$$\sum_{x \in \delta(H)} + \sum_{j=1}^{l} \sum_{x \in \delta(S_j)} x_c \geq 2 \sum_{j=1}^{l} \left\lfloor \frac{d(S_j)}{C} \right\rfloor + 2r(H, \Omega)$$

Comb inequalities

A special kind of valid inequalities, called comb inequalities, arises in the TSP. A comb consists of a so called handle, denoted $H$, and an odd number $l, l \geq 3$, of teeth, denoted $T_i$, $T_2, \ldots, T_l$. The handle and teeth are sets of nodes. All teeth are disjunct and have at least one node in common with the handle. All teeth also have nodes, which are not part of the handle. The arcs contained in the comb are $\gamma(H)$ and $\gamma(T_i), i = 1, 2, \ldots, l$. Note that arcs are counted twice if they have both ends in the handle and in the same teeth. The comb inequality for the TSP is then given by

$$\sum_{x \in \delta(H)} + \sum_{i=1}^{l} \sum_{x \in \delta(T_i)} x_c \geq |H| + \sum_{i=1}^{l} (|T_i| - 1) - \frac{l + 1}{2}$$

This inequality has been proven to be valid for the VRP as well. By using the constraints

$$\sum_{x \in \delta(H)} = 2 \quad \forall i \in V_0$$

$$\sum_{x \in \delta(T_i)} = 2k$$

we can see that the comb inequality is equivalent to

$$\sum_{x \in \delta(H)} + \sum_{i=1}^{l} \sum_{x \in \delta(T_i)} x_c \geq 3l + 1$$

Hypotour inequalities

These inequalities are similar to the hypo-hamiltonian inequalities for the TSP. Let $H$ be a set of edges and $v \in V_0$. If $H$ intersects all the routes including $v$ then the following inequality is a valid hypotour inequality:

$$\sum_{x \in \delta} x_c \geq 1$$

It is possible to generalise this inequality, this is not done here.
Algorithm
The method used in practice can be a cutting plane algorithm that proceeds in the following way. At each iteration, an LP relaxation of the original problem including \( c_1 \) and \( c_2 \) and a set of valid inequalities is solved. Three cases may occur:

1. If the solution is integer and feasible we are done. We have found the optimal solution.
2. If the solution is integer but not feasible is it very easy to identify a violated capacity constraint. We add this constraint and solve the LP once more.
3. If the solution is not integer we strengthen the LP by adding a set of valid inequalities violated by the current solution.

The algorithm stops when we have a solution or, more likely, when we can not find any more valid inequalities violated by the current solution. Since every solution to the LP is a positive estimation of the VRP, the latter case will hopefully give us a good lower bound.

Every time during the solving process that we get a non-integer solution, we have to use algorithms to find the valid inequalities we want to add. Depending on which of the constraints are violated, different algorithms are needed.

Identify capacity constraints
One way of identifying violated capacity constraints is to start shrinking the graph. Starting with the LP solution, we can construct the support graph \( G(x) \) of \( x \). \( G(x) \) is the weighted graph induced by the edges with \( x_e > 0 \). It has been pointed out that any edge \( e = (i,j) \) of \( G(x) \) not incident with the depot and with value 1 can be shrunk in the following way. We create a supernode, \( q \), with demand \( d_q = d_i + d_j \), and redefine the value of any edge \( (q,v) \) as \( x_{qv} = x_{iv} + x_{qv} \). This procedure is repeated until there are no more edges with value 1. Then, if in any step of the shrinking procedure, a supernode with demand greater than \( C \) arises, the capacity constraint associated to the set of its corresponding original nodes is violated. In the early stages of the cutting plane algorithm, many violated capacity constraints are found by this shrinking procedure, but later more sophisticated methods are needed.

Identifying combs
The identification of violated combs has two stages; in the first one, we determine the handle, while in the second, the comb is completed by finding appropriate teeth and is checked for violation.

Identifying hypotours
We start by defining \( E(x) \) as the edge set of the support graph \( G(x) \). The algorithm for finding violated hypotours is also divided in two steps. In the first step, we check if \( x(E - E(x)) \geq 1 \) is a valid inequality (and thus violated). We look for a node \( v \) such that the set of edges \( E - E(x) \) intersects every route containing \( v \). If such node \( v \) is found, the second step of the algorithm strengthens the inequality by looking for a minimal set of edges \( F \) such that \( F \subset E - E(x) \) and \( x(F) \geq 1 \) is valid.
References


