Vehicle Routing

Sidorets Kirill, Ieva Petrulionyte, Bennaceur Zerhouni, Antonio, Ayoub Ouhadi

December 20, 2022

1 Elementary Shortest Path with Single Slot

We want to find the minimum cost tour that starts and ends at the depot and visits n clients at most once. A single visit interval with starting time s_i and ending time e_i is given for each client i.

We represent the locations by an ordered set $\mathcal{T} = \{1, 2, \ldots, n-1, n\}$ where n represents the depot and $1, 2, \ldots, n-1$ represent the clients. The cost of traveling from i to j, $i, j \in \mathcal{T}$ is given by the entry $C_{i,j}$ of an $n \times n$ cost matrix C. The ordering $\mathcal{T}$ is obtained by sorting $n-1$ clients by the starting time of their visit interval (illustrated in figure 1) and appending the depot n at the end of $\mathcal{T}$.

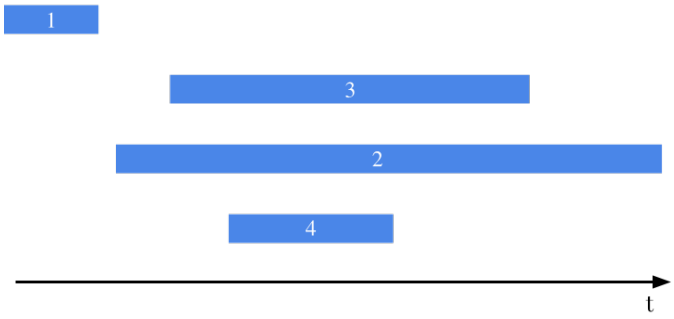

Figure 1

We use dynamic programming to fill two $1 \times n$ tables T and L where T_i is the cost of the optimal tour starting at the depot and ending at location i, $i \in \mathcal{T}$ and L_i is the location visited before i in this tour.

1.1 Recursive relation

Let $t_{i,j}$ the time of traveling from i to j.
The recursive relation for $i \in \mathcal{T}$ is given by:

$$T_i = \begin{cases} +\infty & \text{if } e_i + t_{i,n} > s_n \\ \min \begin{cases} C_{n,i} \\ T_j + C_{j,i} \quad \text{for } j \text{ s.t. } s_i \geq e_j + t_{j,i} \end{cases} & \text{otherwise} \end{cases}$$

if $T_i = C_{n,i}$, then $L_i = n$, otherwise $L_i = j$ where j is the index with minimum feasible $T_j + C_{j,i}$. The cost of the optimal tour is the last entry in the table T_n.

1.2 Pseudo Code

Algorithm 1 Find the cost of the optimal tour given an ordered set of locations $\mathcal{T}$

```
T ← table of size n filled with +∞
L ← table of size n filled with 0
for i ∈ T = 1, 2, ..., n − 1, n do
    m ← C_{n,i}
    m_id ← 0
    for j ∈ 1, 2, ..., i − 1, i do
        m_j = T_j + C_{j,i}
        if s_i ≥ e_j + t_{j,i} and m_j ≤ m then
            m ← m_j
            m_id ← j
        end if
    end for
    T_i ← m
    L_i ← m_id
end for
return T_n
```

1.3 Backtracking

Backtracking is done using the L table since for every $i \in \mathcal{T}$, L_i is the index of the location visited before i in the optimal tour.

2 Elementary Shortest Path with Multiple Slots

We want to find the minimum cost tour that starts and ends at the depot and visits n clients at most once. Multiple visit slots are given for each client c_j.

We consider a tree that enumerates all possible tours where (c_i, s_{ij}) corresponds to visiting client c_i at one of his given time slots s_{ij}. Here we illustrate part of such a tree with 3 clients where client 1 has three intervals, client 2 has two intervals, and client 3 has one interval:

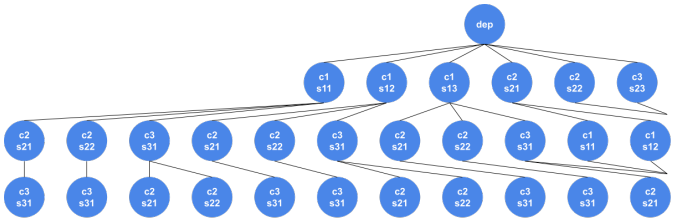

Figure 2

any path in this tree is a tour. We want to cut branches that are infeasible or non-optimal w.r.t. some heuristic or lower/upper bound.

2.1 Infeasibility Pruning

While enumerating the tree, whenever a path (sub-tour) reaches a client c_j, we check if visiting the client creates a cycle in the sub-tour or violates the time window constraint. If any of the two cases are true, the branch can be pruned because all tours starting at this node are infeasible.

2.2 Lower Bound Pruning

If from one client c_i we can reach multiple time slots of another client c_j, we take the earliest time slot of c_j because taking any later time slot could only increase the chance of getting an infeasible solution or would not affect the value of the objective function.

2.3 Dominance Rule

Here we compare solutions that end at the same location and visit the same set of locations. Then node i with cost c_i and tour end time e_i dominates node j if and only if:

$$c_i \leq c_j \text{ and } e_i \leq e_j. \tag{1}$$

3 Vehicle Routing with Single Slot

We have multiple vehicles and want to find a minimum distance set of tours that start and end at the depot and visit n clients exactly once. A single visit interval with starting time s_i and ending time e_i is given for each client i.

3.1 Exponential Formulation

Let $G = (V, E)$ be the fully connected graph with vertices representing all N locations and routes between them. We introduce binary variables x_{ijk} indicating whether tour $k \in 1, \ldots, K$ uses the arc from i to j.

$$\min \sum_{k=1}^{K} \sum_{(i,j)\in E} d(i,j) x_{ijk} \tag{2}$$

$$\sum_{j\in V} x_{0jk} = 1, \quad \forall k \in \{1, \ldots, K\} \tag{3}$$

$$\sum_{i\in V} x_{ijk} - \sum_{i\in V} x_{jik} = 0, \quad \forall k,\ \forall j \in \{1, \ldots, N\} \tag{4}$$

$$\sum_{k=1}^{K} \sum_{j\in V} x_{ijk} = 1, \quad \forall i \in \{1, \ldots, N\} \tag{5}$$

$$x_{ijk}\,(e_i + t_{ij}) \le x_{ijk} s_j, \quad \forall k,\ \forall (i,j) \in E \tag{6}$$

$$x_{ijk} \in \{0, 1\} \tag{7}$$

Constraint (4) ensures that each client is visited by exactly one tour. Constraint (5) defines feasible time slots. We now introduce a model that we can use for column generation.

3.2 Set Covering Reformulation

Let $\mathcal{T}$ be the set of feasible tours, such that each tour starts and ends at the depot and satisfies time slot constraints. Let c_k be the distance of the tour $t_k \in \mathcal{T}$. Let $a_{ik} = 1$ if tour t_k visits customer i and 0 otherwise. Let $b_{ijk} = 1$ if tour t_k uses arc (i, j) and 0 otherwise.

$$c_k = \sum_{(i,j)\in E} b_{ijk} d(i,j) \tag{8}$$

$$a_{ik} = \sum_{\{j\in V | (i,j)\in E\}} b_{ijk} \tag{9}$$

The decision variable $\theta_k = 1$ if tour t_k is selected in the solution and 0 otherwise. We have the set covering model:

$$\min \sum_{t_k\in\mathcal{T}} c_k \theta_k \tag{10}$$

$$\sum_{t_k\in\mathcal{T}} a_{ik} \theta_k = 1, \quad \forall i \in \{1, \ldots, N\} \tag{11}$$

$$\theta_k \in \{0, 1\} \tag{12}$$

We solve the linear relaxation of this problem using the Column Generation procedure.

Algorithm 2 Column Generation

```
Generate an initial set of columns T_{i=0} ⊂ T
while Γ ≠ ∅ do
    s ← master(T_i)   ▷ solve on a subset of columns
    Γ ← subproblem(s)   ▷ find new columns
    T_i ← T_i ∪ Γ   ▷ update the current subset
    i ← i + 1
end while
```

3.3 Master Program

The master problem $\texttt{master}(\mathcal{T}_i)$ is the linear relaxation of the original problem (9-11) where only a subset of tours $\mathcal{T}_i \subset \mathcal{T}$ is being considered:

$$\min \sum_{t_k\in\mathcal{T}_i} c_k \theta_k \tag{13}$$

$$\sum_{t_k\in\mathcal{T}_i} a_{ik} \theta_k \ge 1, \quad \forall i \in \{1, \ldots, N\} \tag{14}$$

$$\theta_k \ge 0 \tag{15}$$

Let λ_i^* the dual value for $i \in \{1, \ldots, N\}$.

3.4 Subproblem

The subproblem $\texttt{subproblem}(s)$ identifies an improving column that can improve the objective function of the master problem w.r.t. current dual values λ_i. We want to find a column that would decrease the objective function if we included it in the set of selected tours. This corresponds to finding a feasible tour $t_k \in \mathcal{T}$ with a negative reduced cost:

$$c_k - \sum_{i=1}^{N} a_{ik} \lambda_i^* < 0 \tag{16}$$

This can also be expressed as:

$$\sum_{(i,j)\in E} b_{ijk} \left(d(i,j) - \lambda_i^*\right) < 0. \tag{17}$$

Tours satisfying this constraint are tours with a negative reduced cost. Finding a tour that satisfies single time slot constraints of the smallest reduced cost corresponds to the problem we solved in 1 with the cost of traveling from i to j given by $C_{i,j} = d(i,j) - \lambda_i^*$.

4 Vehicule Routing with Multiple Slots

We have multiple vehicles and want to find a minimum distance set of tours that start and end at the depot and visit n clients exactly once. Multiple visit intervals are given for each client i.

4.1 Column Generation

To solve the given problem we use the column generation procedure given in algorithm 2. The master program `master` $(\mathcal{T}_i)$ is equivalent to the one discussed in the previous section 3.3. However, the subproblem `subproblem`(s) is different. Here we want the subproblem to find a tour with multiple time slots constraint of the smallest reduced cost, which corresponds to the problem we solved in section 2.