A Trilevel $r$-Interdiction Selective Multi-Depot Vehicle Routing Problem

Supplementary Material (B)

Iterative Marginal Cost Analysis

1-Node iMCA:

The marginal cost of a given customer $i$ is defined as in Fig. A1 where $k$ and $l$ are the predecessor and successor of customer $i$, respectively. If $i$ is not worth visiting, then it can be dropped from the route. In other words, customer $i$ should be outsourced. Otherwise, the cost of visiting customer $i$ turns out to be less than its outsourcing cost. Therefore, visiting customer $i$ is desirable, and it should stay between customers $k$ and $l$. The pseudo code of 1-Node iMCA is provided in Table A1. The pseudo code of 2-Node iMCA can be derived in a straightforward way from there.

![Fig. A1. 1-Node Marginal Cost Analysis](image)

Cost of keeping customer $i = d_{ki} + d_{il}$

Cost of outsourcing customer $i = d_{kl} + c^o q_i$

Marginal cost of customer $i = d_{ki} + d_{il} - d_{kl} - c^o q_i$
2-Node iMCA:

The marginal cost of a given chain of two customers $i$ and $j$ is defined as in Fig. A2 where $k$ and $l$ are the predecessor and successor of customer $i$ and customer $j$, respectively. If the chain of customers $i$ and $j$ is not worth visiting, then they can be dropped from the route. In other words, $i$ and $j$ should be outsourced. Otherwise, the cost of visiting this chain turns out to be less than its outsourcing cost. Therefore, visiting customers $i$ and $j$ is desirable, and they should stay as a chain between customers $k$ and $l$.

![Diagram of 2-Node Marginal Cost Analysis](image.jpg)

Cost of keeping customers $i$ and $j$:
$$= d_{kl} + d_{ij} + d_{jl}$$

Cost of outsourcing customers $i$ and $j$:
$$= d_{kl} + o^c (q_i + q_j)$$

Marginal cost of customers $i$ and $j$:
$$= d_{kl} + d_{ij} + d_{jl} - d_{kl} - o^c (q_i + q_j)$$

Fig. A2. 2-Node Marginal Cost Analysis
Table A1. The pseudo code of 1-Node iMCA

Notation
- $R$ : Current set of routes.
- $N_r$ : Subset of customers on route $r \in R$.
- $MC_i$ : Marginal cost of customer $i$.
- $\text{Index}[MC[1]]$ : Index of customer with the highest marginal cost $MC[1]$.
- $\text{succ}(i), \text{pred}(i)$ : Successor and predecessor of customer $i$, respectively.

1: For every route $r \in R$
2:     For every customer $i \in N_r$ on route $r$
3:         Set $MC_i = d_{ki} + d_il - d_{kl} - c_{ilq_i}$ ; // Compute marginal cost of each customer on route $r \in R$.
4:     End For
5: Sort $MC_i$ values in nondecreasing order and create a sorted stack $S$ ;
6: While ($|N_r| > 0$ )
7:     Retrieve $MC[1] = \text{Pop}(S)$ ; // Return and remove the highest marginal cost.
8:     If $MC[1] < 0$ // Marginal cost of all customers are negative.
9:         Break While loop; // Stop the marginal cost analysis on the current route $r$ .
10: Else
11:     Set $i[1] = \text{Index}[MC[1]]$ ; // Customer $i$ with highest marginal cost.
12:     Remove $i[1]$ from the route $r \in R$ ; // Remove the customer with the most positive $MC$.
13:     Update the $MC$ values of $\text{succ}(i[1])$ and $\text{pred}(i[1])$ ;
14:     Restore the nondecreasing order of $MC_i$ in the sorted stack $S$ ;
15:     Update $r \in R$ ;
16:     Update $N_r$ ;
17:     If $|N_r| = 0$ // Route $r$ does not visit any customers.
18:         Discard route $r \in R$ ;
19:     End If
20: End If
21: End While
22: End For