Capacitated Arc Routing Problem

CARP can be described as follows: consider an undirected connected graph $G = (V, E)$, with a vertex set V and an edge set E and a set of required edges (tasks) $T \subseteq E$. A fleet of identical vehicles, each of capacity Q, is based at a designated depot vertex $v_0 \in V$. Each edge $e \in E$ incurs a cost $c(e)$ whenever a vehicle travels over it or serves it (if it is a task). Each required edge (task) $\tau \in T$ has a demand $d(\tau) > 0$ associated with it.

The objective of CARP is to determine a set of routes for the vehicles to serve all the tasks with minimal costs while satisfying: a) Each route must start and end at v_0; b) The total demand serviced on each route must not exceed Q; c) Each task must be served exactly once (but the corresponding edge can be traversed more than once)

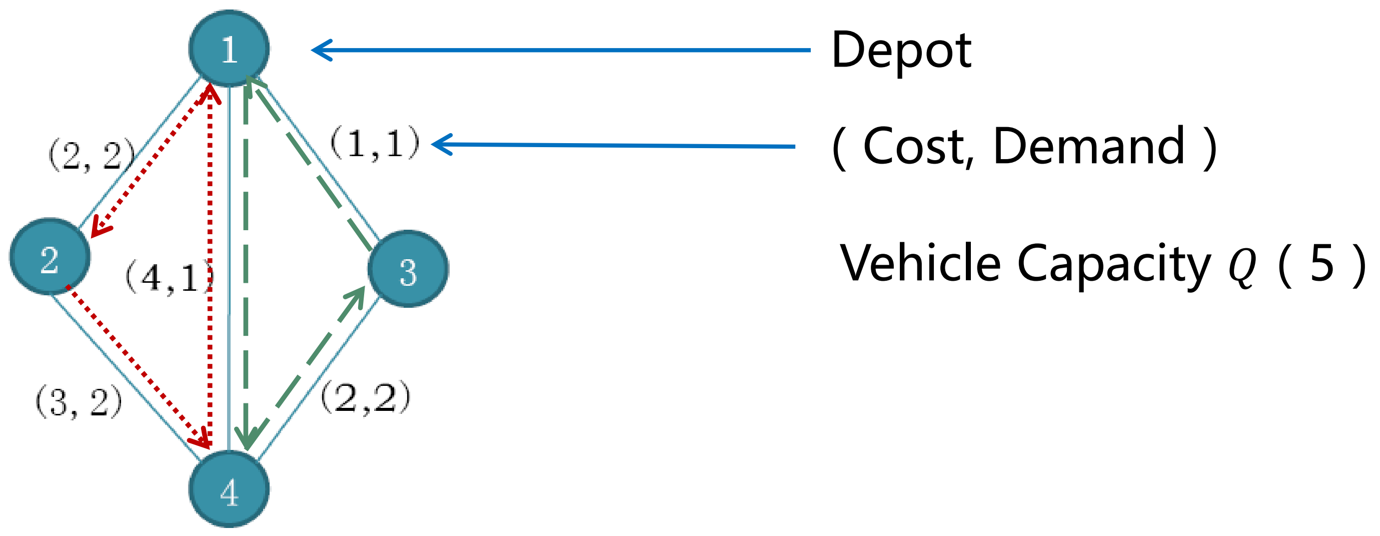
Problem description of CARP

Capacitated Arc Routing Problem (CARP)

- Inputs:
  - An undirected connected graph $G(V, E)$
  - Cost $c(e)$ > 0, and demand $d(e) \geq 0$ for $\forall e \in E$
  - The task set $T = \{\tau \epsilon E | d(\tau) > 0\}$
  - A predefined vertex (depot) $v_0 \in V$, where a set of vehicles are based
  - Vehicle capacity Q

Problem description of CARP

- The goal: to determine a set of routes for the vehicles with minimal costs, which satisfy:
  - Each route starts and ends at v_0
  - Each task is served exactly once
  - The total demand of tasks served in each route $\leq Q$



Problem description of CARP

- Solution representation
  - The route of a vehicle can be represented by
    a) A sequence of vertices which the vehicle visits one by one
    b) A sequence of tasks which the vehicle serves one by one
  - The second representation is more compact
    - The shortest path between the consecutive tasks can be calculated in polynomial time using Dijkstra algorithm

Problem description of CARP

- Solution representation
  - Thus, we have a solution to CARP as:

$$s = (R_1, R_2, \dots, R_m)$$

m is the number of routes (vehicles). The kth route $R_k = (0, \tau_{k1}, \tau_{k2}, \dots, \tau_{kl_k}, 0)$, where τ_{kt} and l_k denote the tth task and the number of tasks served in R_k, and 0 denotes a dummy task which is used to separate different routes. The cost and the demand of the dummy task are both 0 and its two endpoints are both v_0 (the depot).

Moreover, since each task here is an undirected edge and it can be served from either direction, so each task in R_k must be specified from which direction it will be served. Specifically, $\tau_{kt} = \big(head(\tau_{kt}), tail(\tau_{kt})\big)$, where $head(\tau_{kt})$and $tail(\tau_{kt})$ represent the endpoints of τ_{kt}, and τ_{kt} is served from $head(\tau_{kt})$ to $tail(\tau_{kt})$.

Problem formulation of CARP

- Formally, the objective function of CARP is:

$$\textbf{minimize } \boldsymbol{TC(s)} \;\longrightarrow\; \text{Total cost of all routes}$$

$s.t$ (2) $\sum_{k=1}^{m} l_k = |T|$

(3) $\tau_{k_1 i_1} \neq \tau_{k_2 i_2}, \forall (k_1, i_1 \neq k_2, i_2)$

(4) $\tau_{k_1 i_1} \neq inv(\tau_{k_2 i_2}), \forall (k_1, i_1 \neq k_2, i_2)$

→ Each task is served exactly once

(5) $\sum_{i=1}^{l_k} d(\tau_{ki}) \leq Q, \forall k = 1,2, \dots, m$ → Capacity constraint

(6) $\tau_{ki} \in T$ → Definition of τ

$head(\tau)$ and $tail(\tau)$ represent the endpoints of task τ and specifies the direction from which τ is served, and $inv(\tau)$ represent the inverse direction

Problem formulation of CARP

$$TC(s) = \sum_{k=1}^{m} RC(R_k)$$

$RC(R_k)$ is the route cost of route R_k, which can be computed as:

$$RC(R_k) = \sum_{i=1}^{l_k} c(\tau_{ki}) + dc(v_0, head(\tau_{k1})) + \sum_{i=2}^{l_k} dc\left(tail(\tau_{k(i-1)}), head(\tau_{ki})\right) + dc(tail(\tau_{kl_k}), v_0)$$

$dc(v_i, v_j) > 0$ is the cost of the shortest path from v_i to v_j $(i \neq j)$