# A Novel Location-Allocation-Routing Model for Siting Multiple Recharging Points on the Continuous Network Space

by

Yazhu Song

A Dissertation Presented in Partial Fulfillment
of the Requirement for the Degree of
Doctor of Philosophy

Approved January 2020 by the
Graduate Supervisory Committee:

Pitu Mirchandani, Chair
Teresa Wu
Jorge Sefair
Arunabha Sen

ARIZONA STATE UNIVERSITY

May 2020

ABSTRACT

Due to environmental and geopolitical reasons, many countries are embracing electric vehicles (EVs) as an alternative to gasoline powered automobiles. Other alternative-fuel vehicles (AFVs) powered by compressed gas, hydrogen or biodiesel have also been tested for replacing gasoline powered vehicles. However, since the associated refueling infrastructure of AFVs is sparse and is gradually being built, the distance between recharging points (RPs) becomes a crucial prohibitive attribute in attracting drivers to use such vehicles. Optimally locating RPs will both increase demand and help in developing the refueling infrastructure.

The major emphasis in this dissertation is the development of theories and associated algorithms for a new set of location problems defined on continuous network space related to siting multiple RPs for range limited vehicles.

This dissertation covers three optimization problems: locating multiple RPs on a line network, locating multiple RPs on a comb tree network, and locating multiple RPs on a general tree network. For each of the three problems, finding the minimum number of RPs needed to refuel all Origin-Destination (O-D) flows is considered as the first objective. For this minimum number, the location objective is to locate this number of RPs to minimize weighted sum of the travelling distance for all O-D flows. Different exact algorithms are proposed to solve each of the three algorithms.

In the first part of this dissertation, the simplest case of locating RPs on a line network is addressed. Scenarios include single one-way O-D pair, multiple one-way O-D pairs, round trips, etc. A mixed integer program with linear constraints and quartic objective function is formulated. A finite dominating set (FDS) is

identified, and based on the existence of FDS, the problem is formulated as a shortest path problem. In the second part, the problem is extended to comb tree networks. Finally, the problem is extended to general tree networks. The extension to a probabilistic version of the location problem is also addressed.

DEDICATION

*I dedicate this dissertation to my parents,*

*who have loved me so well for all these years.*

*Hi, Mom and Dad!*

*Sorry about that you cannot read English. Ha-ha.*

*But seriously, I love you.*

ACKNOWLEDGMENTS

I wish to express my sincere gratitude to the members of my graduate supervisory committee, Professor Pitu Mirchandani, Professor Jorge Sefair, Professor Teresa Wu and Professor Arunabha Sen, for their time and guidance.

I am deeply grateful for the guidance and mentorship provided by my committee chair, Professor Pitu Mirchandani. From him I have learned much about optimization, transportation science, and what is the right attitude towards challenges in research as well as in life. To be honest, initially I hesitated to work on this challenging research topic, since there was barely related work that has been done and getting every new tiny result took me days, weeks or even months. But Dr. Mirchandani kept encouraging me, "This is research, no wonder that you will experience desperate times during your Ph.D. life. Hang in there!" Thank you for your frank and powerful words. As I delved deeper into it, I have realized he provided me with such a great interesting topic for my dissertation, which not only contributes to location theory but also gives mathematics aesthetic pleasure. Thank you! Your concern for my well-being was apparent all the way through.

Thank you to my friends Xiushuang Li, Guanqi Fang, Yinlin Fu, Congzhe Su, Xiaonan Liu, Kerem Demirtas, Gina Dumkrieger, Gita Ketut and Viswananth Potluri.

Finally, I am very grateful to my family for always supporting me, encouraging me, and believing in me. My parents deserve the utmost appreciation. Their constant and unwavering support helped me overcome all the difficulties I had in my life so far. They have blessed me in more ways than I can tell.

TABLE OF CONTENTS

Page

CHAPTER Page

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

Figure Page

# CHAPTER 1

# INTRODUCTION

## 1.1. Overview

The environmental, geopolitical and financial implications of the global dependence on oil are well known, and much is being done to lessen our use of fossil fuels. Over the last few years, vehicles that are powered by electricity or other alternative clean fuels have received increasing attention as an alternative to traditional gasoline powered automobiles, since they could potentially to help reduce the world's consumption of non-renewable energy resources as well as decrease consumers' transportation costs. Whereas both automobile companies and governments have been trying to incentivize the use of such vehicles, they are still not widely accepted by the public. One of the primary reasons is that the associated recharging/refueling infrastructure is sparse and is gradually being built, and, hence, a driver has to deal with the "range anxiety", that is, the fear of his/her vehicle will run out of charge or fuel before reaching the destination. In this sense, optimally locating recharging/refueling stations will both help market penetration of such vehicles and help in developing the recharging/refueling infrastructure. Hereon, we simply refer to recharging/refueling as 'refueling' for brevity.

Suppose we wish to locate $n$ refueling points (RP) on a transportation network where there are none currently. The problem of optimally locating such RPs has been initially investigated by Kuby and his collaborators, e.g., Kuby and Lim (2005), Kuby and Lim (2007), Upchurch et al. (2009), Lim and Kuby (2010), and

Capar et al. (2012). Typically, they use modifications of flow capturing or flow interception models [Hodgson (1990), Berman et al. (1992), Rebello et al. (1995)], to locate a given number of RPs chosen from a given discrete set of potential sites to "cover" as many origin-destination (O-D) routes as possible, where covering an O-D pair means there are paths between O and D so that a vehicle will not run out of fuel.

Taking a trip, especially one through sparsely populated areas, requires the driver to plan where the vehicle will need to be refueled. Given the abundance of gasoline stations for standard vehicles, a driver usually considers refueling only when the fuel tank is low. In the case of range-limited vehicles (RLVs), planning when to refuel is important, since there are few places to refuel because, at least initially, RPs would be few and far in between. Therefore, one needs to develop models which consider routes that include detouring to RPs if necessary. Objectives for such models could be to (a) minimize the total detouring distance for all O-D pairs and (b) minimize the total number of refueling stops. It is surprising that detouring is not a consideration in most of the reported models. In fact, detouring plays a major role in the problems that have been analyzed in this research. Finding routes in a network considering refueling detours have been studied by, among others, Ichimori (1981), Laporte and Pascoal (2011), Smith et al. (2012), and Adler and Mirchandani (2014).

In this research, minimization of the number of RPs to refuel all O-D flows is considered as the first objective, since building refueling infrastructure is costly. For this minimum number, the location objective is to locate this number of stations to minimize weighted sum of the distance traveled for all O-D flows. The research

starts with the simplest case, locating RPs on a real line network, then this is extended to tree networks, and finally to general networks.

### 1.2. Literature review

Bapna et al. (2002) present a study on locating gas station facilities in developing countries like India which are in the midst of conversion from leaded gasoline to unleaded fuel. They attempt to maximize the population that will be within a given distance from the new gas stations and simultaneously minimize the cost locating the stations, while making all inter-city travel possible. Their method is based on the notion of "enabling arcs", which means to locate a necessary number of stations on the arc such that a vehicle can use it for travel. They propose a heuristic procedure to solve the problem, where at each step an arc of the highest population coverage per unit cost is enabled until a strongly connected spanning subgraph is achieved. However, their model double counts the amount of coverage. Also, this method does not address the problem of where exactly to locate stations, since it is still possible for a vehicle not to be able to travel from one arc to another along a path even though each comprising arc is enabled.

Hodgson (1990) propose a Flow-Capturing Location-Allocation Model (FCLM), which locates a specified number of facilities at nodes in a network to maximize (O-D) traffic flow capture. The term "capture" describes a form of covering the flow: a facility at a node "captures" all the flow which passes through that node, where, for example, in advertisement applications it captures the eyeballs of the vehicles' travelers, and in retailing it captures potential demand for searching any

services/products. In this sense, location at a node is at least as good as location on any adjacent link. Using the following notation one can formulate FCLM:

**Table 1.1**

*Notation for FCLM*

| | |
|---|---|
| $q$ | a particular O-D pair |
| $Q$ | the set of all O-D pairs |
| $f_q$ | the flow between O-D pair $q$ |
| $y_q$ | $= \begin{cases} 1, & \text{if } f_q \text{ is captured} \\ 0, & \text{otherwise} \end{cases}$ |
| $k$ | a potential facility location |
| $\boldsymbol{K}$ | the set of all potential facility locations |
| $x_k$ | $= \begin{cases} 1, & \text{if there is a facility at location } k \\ 0, & \text{otherwise} \end{cases}$ |
| $\boldsymbol{N}_q$ | the set of nodes capable of capturing $f_q$ (the set of nodes on path $q$ between $O_i$ and $D_j$) |
| $p$ | the number of facilities to be located |

FCLM:

$$\text{Maximize} \quad Z = \sum_{q \in \boldsymbol{Q}} f_q y_q \tag{1.1}$$

$$\text{Subject to} \quad \sum_{k \in \boldsymbol{N}_q} x_k \geq y_q \quad for\ all\ q \in \boldsymbol{Q} \tag{1.2}$$

$$\sum_{k \in \boldsymbol{K}} x_k = p \tag{1.3}$$

Objective (1.1) maximizes flow captured and where constraint (1.2) ensures that flow on a path $q$ can be captured only if a facility is located on the path, and constraint (1.3) limits the specified number $p$ facilities being located.

Based on the FCLM, Kuby and Lim (2005) develop the Flow-Refueling Location Model (FRLM), which seeks to locate a pre-specified number of refueling points at network nodes to refuel as many O-D flows as possible. For each O-D pair,

their model initially generates the shortest path and assumes that the O-D is covered if RPs are located on these paths such that RLVs can travel on their shortest paths; it determines a set of minimal RP combinations that can refuel each path. A combination $H$ is said to be minimal, if station $k$ is in $H$, then $H - k$ is not able to refuel such paths. Then the problem is formulated as a mixed-integer program and solved exactly by using the branch-and-bound algorithm. However, there are two drawbacks to this model: (a) the solution can be suboptimal when facility placement is only allowed at nodes, and (b) in reality, people might not always be able to choose the shortest path, any may be able to detour to refuel.

FRLM:

$$\text{Maximize} \quad Z = \sum_{q \in \boldsymbol{Q}} f_q y_q \tag{1.4}$$

$$\text{Subject to} \quad \sum_{h \in \boldsymbol{H}} b_{qh} v_h \geq y_q \quad for\ all\ q \in Q \tag{1.5}$$

$$x_k \geq v_h \quad for\ all\ h \in \boldsymbol{H}; for\ all\ k\ where\ a_{hk} = 1 \tag{1.6}$$

$$\sum_{k \in \boldsymbol{K}} x_k = p \tag{1.7}$$

where,

**Table 1.2**

*More notation for FRLM*

| | |
|---|---|
| $h$ | a particular facility combination |
| $\boldsymbol{H}$ | the set of all potential facility combinations |
| $a_{hk}$ | a coefficient, $= \begin{cases} 1, & \text{if facility } k \text{ is in combination } h \\ 0, & \text{otherwise} \end{cases}$ |
| $b_{qh}$ | a coefficient, $= \begin{cases} 1, & \text{if combination } h \text{ can refuel OD pair } q \\ 0, & \text{otherwise} \end{cases}$ |
| $v_h$ | $= \begin{cases} 1, & \text{if all facilities in combination } h \text{ are established} \\ 0, & \text{otherwise} \end{cases}$ |

For a network location problem where researchers do not know whether a finite dominating set (FDS) exists or not, adding additional potential facility sites along the arcs may lead to better solutions compared to merely restricting placement at given network nodes. Kuby et al. (2005) introduce the Added-Node Dispersion Problem (ANDP). By adding a node to an arc, the arc will be subdivided into sub-arcs. ANDP is to add a given number of nodes along arcs to optimize some criterion function. Based on minimax and maximin distance criteria, they studied heuristics for adding nodes and sub-arcs. The minimax objective aims to minimize the maximum sub-arc length, and the maximin objective aims to maximize the minimum sub-arc length. Then on the basis of minimax (maximin), a secondary objective is considered, which aims to minimize the sum of maximum sub-arc length.

Kuby and Lim (2007) improve the FRLM by adding additional discrete candidate sites along network arcs. They propose three methods: the mid-path segment method, the minimax added-node dispersion problem (ANDP) method, and the maximin ANDP method. For each path that can be refueled with one single facility (except origin and destination), i.e., $d_q \leq r < 2d_q$, where $d_q$ denotes the one-way shortest path distance and $r$ denotes the vehicle range, the mid-path segment method identifies a line segment in which any point can refuel the path by itself. Once all line segments are generated, the method breaks up the overlapping segments, only retains the dominating ones, and adds additional candidate sites, each of which is a midpoint of the segment. The ANDP method does not depend on the vehicle range, which simply disperses additional candidate sites along arcs. The minimax version aims to minimize the longest sub-arc length to prevent any long stretches of network without any candidate sites; and the maximin version aims to

maximize the shortest sub-arc length to not wasting candidate sites by locating them too close together. However, none of these methods generate a finite dominating set. The mid-path segment method does not consider how segments may coordinate with each other to refuel O-D flows, and it will just ignore any path with $d_q > r$. The computation cost explodes for ANDP method as more candidate sites are added.

Kim and Kuby (2011) develop the Deviation-Flow Refueling Location Model (DFRLM), in which they relax the assumption of FRLM --- that each O-D flow sticks with the shortest path between the two nodes. They use a modified $k$ shortest path algorithm to generate deviation paths within a certain upper distance limit for each O-D pair, compute the fraction of flow volume on deviation paths, and come up with combinations of stations to refuel deviation paths, then solve the problem as FRLM.

Cabral et al. (2007) consider a network design problem with relays (NDPR) in the context of telecommunication network design and proposed a column generation scheme and four algorithms. Konak (2012) also studies NDPR and propose a set covering formulation with a meta-heuristic algorithm. However, these models only choose nodes to locate relays, which may not be optimal in our transportation application.

# CHAPTER 2

# THE LINE PROBLEM

Now we address some continuous versions of the detouring-flow location problem on a line network. The set of potential RP locations is relaxed to include not just the network nodes and pre-determined points on the arcs, but all points on the network. That is, every point on an arc is an allowable site for establishing an RP. The objective of this first problem is to (a) first determine the minimum number of RPs that are necessary and sufficient to refuel all O-D traffic flow; and (b) then determine the optimal locations for RPs that minimize the total travelling distance for all vehicles.

## 2.1. Problem with only one-way trips

Suppose the road network is represented as an undirected line graph $G = (V, A)$, where the node set $V = \{v_0, v_2, \cdots, v_n\}$ represents a set of $n + 1$ origin and/or destination nodes, connected by arc set $A = \{(v_j, v_{j+1}), j = 0, \cdots, n-1\}$. See figure 2.1 as an illustration. Associated with each arc $(v_j, v_{j+1})$ is a nonnegative weight $b(v_j, v_{j+1})$ representing its length, and we have $b(v_j, v_{j+1}) = b(v_{j+1}, v_j)$. The length $b(x, y)$ of the portion of arc between points $x$ and $y$ on $(v_j, v_{j+1})$ is defined to be $b(x, y) = |b(v_j, x) - b(v_j, y)|$. The length function $b$ yields a distance function $d$ for the line network, where $d(x, y)$ is defined to be the shortest path length from $x$ to $y$ for any two points $x$, $y$ on $G$. However, in a line network, there is a unique path between them and let $P(x, y)$ denote this path. Furthermore, let $l(x)$ denote the coordinate of $x$ on $G$, which is taken to be the distance from $v_0$ to $x$, i.e. $l(x) =$

$d(v_0, x) + l(v_0)$. Specifically, $l(v_0) = 0$. Hereon, the two expressions $x \prec y$ and $l(x) < l(y)$ will be used interchangeably to indicate that point $x$ is on the left-hand side of $y$ on $G$.

We denote a pair $(v_i, v_j)$ as the one-way transportation need for flow from $v_i$ to $v_j$, for all $v_i, v_j \in V$. The average traffic flow volume on $P(v_i, v_j)$ is denoted as $f(v_i, v_j)$. As a prototypical application, many drivers of electric vehicles will charge their vehicles overnight at their origin (home), and will have proper charging facilities at their destination (workplace) as well. In this problem, an electric vehicle is assumed to depart from its origin with a fully charged battery and needs to reach its destination. If a set of RP locations is given, then we say that trip $(v_i, v_j)$ can be served if the path $P(v_i, v_j)$ has no segment without refueling with a length greater than the range limit. We let $r$ denote the range limit, i.e., the maximum distance that our fully charged electric vehicle can travel before refueling.

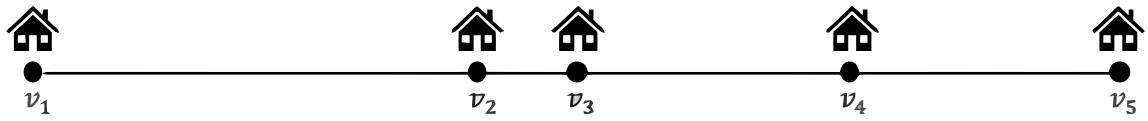



**Figure 2.1** A line road network with 5 nodes

### 2.1.1. Set of candidate sites

We begin to formalize the discussion of our model development by way of a simple example.

EXAMPLE 2.1 Consider the following line network in Figure 2.2, which consists of two nodes $O$ and $D$, and a single arc $(O, D)$ with length $2r < b(O, D) < 3r$. Suppose that two RPs are necessary and sufficient to serve flows from $O$ to $D$ and flows from $D$ to $O$.

To refuel flows from $O$ to $D$: The first RP should be located within $r$ distance from node $O$ and within $2r$ distance to node $D$, and the second RP should be located within $2r$ distance from $O$ and within $r$ distance to $D$. Conversely, to refuel flows from $D$ to $O$, the second RP should be located within $r$ distance from node $D$ and within $2r$ distance to node $O$, and the first RP should be located within $2r$ distance from $D$ and within $r$ distance to $O$. Thus, the set of potential sites for the first RP includes all points on the line segment from point $d(= l(D) - 2r)$ to point $a(= r)$, and the set of potential sites for the second RP includes all points on the line segment from point $c(= l(D) - r)$ to point $b(= 2r)$, as indicated in Figure 2.2. Last but not least, the distance between the two RPs should not exceed $r$.

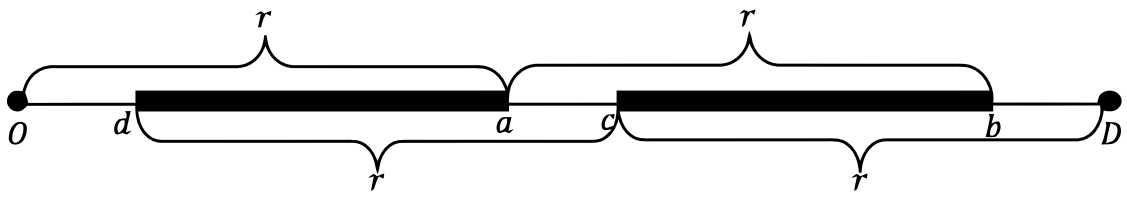



**Figure 2.2** A simple line network for example 2.1

Before proceeding with the discussion, let us state two facts, which are useful and easy to be verified.

***FACT 2.1*** If flows on $(v_i, v_j)$ can be served by the RPs established on the line network, then flows on $(v_j, v_i)$ can be served as well.

***FACT 2.2*** No refueling detouring will occur for electric vehicles on their one-way trips.

### 2.1.2. Minimum number of RPs needed

We now consider the problem: Given a line network $G$ with ordered nodes $v_0, v_1, \dots, v_n$ and a vehicle range limit $r$, what is the minimum number of RPs that should be located so that all O-D flows can be served?

***Theorem 2.1*** The minimum number of RPs that are necessary and sufficient to serve all O-D transportation needs is

$$m = \begin{cases} \left\lfloor \frac{l(v_n)}{r} \right\rfloor, & if\ mod(l(v_n), r) \neq 0 \\ \frac{l(v_n)}{r} - 1, & otherwise \end{cases}, \tag{2.1}$$

equivalently,

$$m = \left\lceil \frac{l(v_n)}{r} \right\rceil - 1. \tag{2.2}$$

Let $\mathcal{P} = \{\mathscr{p}_1, \mathscr{p}_2, \cdots, \mathscr{p}_m\}$ represent the set of RPs to be established on $G$, and let $l(\mathscr{p}_1) < l(\mathscr{p}_2) < \cdots < l(\mathscr{p}_m)$ without loss of generality, i.e., $\mathscr{p}_1 \prec \mathscr{p}_2 \prec \cdots \prec \mathscr{p}_m$. For notational convenience, we introduce $\tau_i$ to represent the position of $\mathscr{p}_i$ in the sequence of RPs in a left-to-right order. Then $\tau_i := i$, where $i$ is the subscript of $\mathscr{p}_i$. $\tau_i$ and $\tau(\mathscr{p}_i)$ will be used interchangeably. Let $\mathscr{p}_{v_j}^-$ and $\mathscr{p}_{v_j}^+$ represent the closest RP established on the left-hand side and on the right-hand side of $v_j$, respectively. However, we should note that for some node $v$ on $G$, it is possible that (a) $v$ does not have a RP established on its left-hand side (for example, node $v_0$) or on its right-hand side (for example, node $v_n$); and that (b) a RP has been established just at the position of node $v_j$, and we say that $\mathscr{p}_v^- = \mathscr{p}_v^+$.

Before proceeding with the proof of theorem 2.1, let us first state the proposition 2.1, based on which we shall prove theorem 2.1.

***Proposition 2.1*** Given a set of RP locations, if flows on O-D pair $(v_0, v_n)$ can be served, then all O-D pairs can be served.

*Proof of Proposition 2.1.* Suppose that the assertion fails. Then we may assume that there exists some O-D pair $(v_i, v_j)$ that cannot be refueled, and without loss of generality we let $v_i \prec v_j$. Then the length of one or more of the following intervals would be greater than the range limit $r$:

- the interval between node $v_i$ and RP $\wp_{v_j}^+$,
- the interval between any two consecutive RPs $\wp_k$ and $\wp_{k+1}$ where $\tau(\wp_{v_i}^+) \leq k < \tau(\wp_{v_j}^-)$,
- the interval between RP $\wp_{v_j}^-$ and node $v_j$.

However, if that is true, the flows on pair $(v_0, v_n)$ cannot be served either. ■

We can now prove theorem 2.1 using this proposition.

*Proof of Theorem 2.1* By proposition 2.1, we know that to compute the minimum number of RPs $m$, it is sufficient to just consider a single O-D pair $(v_0, v_n)$. Now suppose that we are going to locate the RPs in the following fashion:

- If $mod(l(v_n), r) \neq 0$, we locate RPs at distances $r, 2r, \cdots,$ and $\left\lfloor \frac{l(v_n)}{r} \right\rfloor \times r$ from node $v_0$;
- Otherwise, we locate RPs at distances $r, 2r, \cdots,$ and $\left(\frac{l(v_n)}{r} - 1\right) \times r$ from node $v_0$.

It is easy to see that the flows on pair $(v_0, v_n)$ can be refueled. Hence,

$$m \leq \begin{cases} \left\lfloor \frac{l(v_n)}{r} \right\rfloor, & if\ mod(l(v_n), r) \neq 0 \\ \frac{l(v_n)}{r} - 1, & otherwise \end{cases},$$

which gives us an upper bound of $m$.

When $mod(l(v_n), r) \neq 0$, suppose that we locate $\left\lfloor \frac{l(v_n)}{r} \right\rfloor - 1$ RPs, then the average length of the following intervals:

- the interval between node $v_0$ and RP $\wp_1$,
- the interval between any two consecutive RPs $\wp_k$ and $\wp_{k+1}$ where $1 \leq k < \left\lfloor \frac{l(v_n)}{r} \right\rfloor - 2$,
- the interval between RP $\wp_{\left\lfloor \frac{l(v_n)}{r} \right\rfloor - 1}$ and node $v_n$,

would be $\bar{d} = \frac{l(v_n)}{\left\lfloor \frac{l(v_n)}{r} \right\rfloor - 1} \geq \frac{l(v_n)}{\frac{l(v_n)}{r} - 1} > r$, which implies that flows on pair $(v_0, v_n)$ cannot be refueled. When $mod(l(v_n), r) \equiv 0$, we could deduce the same conclusion. Thus, a lower bound of $m$ has been found

$$m \geq \begin{cases} \left\lfloor \frac{l(v_n)}{r} \right\rfloor, & if\ mod(l(v_n), r) \neq 0 \\ \frac{l(v_n)}{r} - 1, & otherwise \end{cases}.$$

We have proved equation (2.1), and it is trivial to see that equations (2.1) and (2.2) are equivalent. ■

As a by-product of Theorem 2.1, we can derive a localization segment for each RP in $\mathcal{P}$, which contains all the allowable sites for that RP. We let $S_k = [\alpha_k, \beta_k]$ denote the localization segment of $\wp_k$.

***Corollary 2.1*** Given that $m$ RPs are to be established on the line, then

$$S_k = [l(v_n) - (m - k + 1)r, kr], \text{ for } 1 \leq k \leq m. \tag{2.3}$$

***Theorem 2.2*** If the objective aims to minimize the total travelling distance, then each feasible solution is an optimal solution.

*Proof.* By FACT 2.2, we know that no refueling detouring will occur for electric vehicles on their one-way trips. Hence, any feasible solution is also optimal. ■

## 2.2. Round trip problem

We now formally consider the case where electric vehicles go for round trips between any two nodes along a road. Let triple $(v_i, v_j, v_i)$ represent a round trip demand, that is, a vehicle starts at $v_i$, arrives at its intended destination $v_j$ and then goes back to $v_i$. The subtrip from origin $v_i$ to destination $v_j$ is called an outbound trip, and the subtrip from $v_j$ back to $v_i$ is called an inbound trip. If a vehicle is able to reach $\wp_{v_j}^-$ (the RP that is on the left-hand side of $v_j$ and closest to $v_j$) during its outbound trip $v_i \to v_j$, and $\wp_{v_j}^-$ is within $\frac{r}{2}$ distance to $v_j$, then the vehicle is able to reach $v_j$ with at least a half full fuel cell and return to $\wp_{v_j}^-$ without running out of fuel. Otherwise, after arriving at $v_j$, the vehicle has to make a detour to visit $\wp_{v_j}^+$ for refueling and then returns to $v_j$.

### 2.2.1. Minimum number of RPs needed

Recall the FACT 2.1, which claims that if flows on $(v_i, v_j)$ can be served by the RPs established on the line, then flows on $(v_j, v_i)$ can be served as well. Here, unlike the one-way case, the fact that a round trip $(v_i, v_j, v_i)$ can be served by the RPs established on the line does not necessarily imply that the round trip $(v_j, v_i, v_j)$ can also be served. Using the following example, let us examine that why FACT 2.1 cannot be generalized to the round-trip case, and also get a rough idea about how to identify a set of RPs to serve all round trips, with minimum cardinality.

EXAMPLE 2.2 Consider the line network of figure 2.3, which is same to the one that was used in example 2.1. Again, we assume that two RPs are to be established on the line. First, we consider the round-trip $(O, D, O)$. To refuel the outbound trip

$O \rightarrow D$, $\wp_1$ should be located within $r$ distance from node $O$ and $2r$ distance to node $D$, and $\wp_2$ should be located within $2r$ distance from $O$ and $r$ distance to $D$. To refuel the inbound trip $D \rightarrow O$, $\wp_2$ should be within $\frac{r}{2}$ distance from $D$ and $2r$ distance to $O$, and $\wp_1$ should be within $\frac{3r}{2}$ distance from $D$ and $r$ distance to $O$. Thus, the set of allowable sites for $\wp_1$ includes all points on the line segment from point $d\left(= l(v_n) - \frac{3r}{2}\right)$ to point $a(= r)$, and the set of allowable sites for $\wp_2$ includes all points on the line segment from point $c\left(= l(v_n) - \frac{r}{2}\right)$ to point $b(= 2r)$, as indicated by two black bold segments in figure 2.3. Moreover, the distance between the two RPs should be less than or equal to $r$.

Follow the same fashion, another two segments, $[b', c']$ and $[a', d']$, for serving the round-trip $(D, O, D)$ can be identified as well, where $b' = l(v_n) - 2r$, $c' = \frac{r}{2}$, $a' = l(v_n) - r$, and $d' = \frac{3r}{2}$. $[b', c']$ and $[a', d']$ are indicated by two red bold segments in figure 2.3.

From the figure, we can see that the black segments and red segments are nonoverlapped, implying that two RPs are not enough to serve both $(O, D, O)$ and $(D, O, D)$.

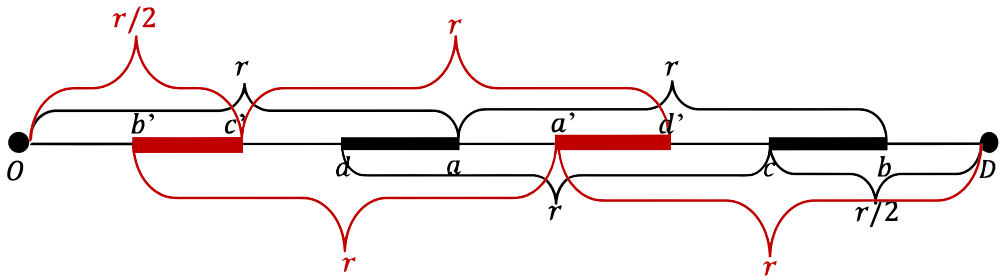



**Figure 2.3** A line network for example 2.2, where 2 RPs are not enough

***Proposition 2.2*** Given a set of RP locations, if flows on round trips $(v_0, v_n, v_0)$ and $(v_n, v_0, v_n)$ can be refueled, then flows on any round trip $(v_i, v_j, v_i)$ can be refueled, where $0 \leq i, j \leq n$. Furthermore, refueling detouring may occur for electric vehicles on their round trips.

*Proof of Proposition 2.2* Let $\mathcal{P}$ be a set of RPs established on the line such that round trips $(v_0, v_n, v_0)$ and $(v_n, v_0, v_n)$ can be served. Then we can safely conclude that

- $\mathscr{p}_1$ is located within $\frac{r}{2}$ distance from node $v_0$;
- the interval between any two consecutive RPs $\mathscr{p}_k$ and $\mathscr{p}_{k+1}$ is not more than $r$, where $1 \leq k < |\mathcal{P}| - 1$ ($|\mathcal{P}|$ is the cardinality of $\mathcal{P}$);
- $\mathscr{p}_{|\mathcal{P}|}$ is located within $\frac{r}{2}$ distance to node $v_n$.

Consider any round trip $(v_i, v_j, v_i)$, and without loss of generality we assume that $v_i \prec v_j$. Then:

(a) The outbound trip $v_i \rightarrow v_j$ can be refueled, since the length of each of the following intervals would be not more than $r$: the interval between node $v_i$ and $\mathscr{p}_{v_j}^+$, the interval between any two consecutive RPs $\mathscr{p}_k$ and $\mathscr{p}_{k+1}$ where $\tau(\mathscr{p}_{v_i}^+) \leq k < \tau\left(\mathscr{p}_{v_j}^-\right)$, and the interval between $\mathscr{p}_{v_j}^-$ and $v_j$.

(b) The inbound trip $v_j \rightarrow v_i$ can be refueled as well. If $\mathscr{p}_{v_j}^-$ is located within $\frac{r}{2}$ distance from $v_j$, after arriving at node $v_j$, the vehicle is able to return to $\mathscr{p}_{v_j}^-$ without running out of fuel. Otherwise if $\mathscr{p}_{v_j}^-$ is located beyond $\frac{r}{2}$ distance from $v_j$, the vehicle is able to make a detour to visit $\mathscr{p}_{v_j}^+$ for refueling and goes back to $v_i$, since the distance between $\mathscr{p}_{v_j}^-$ and $\mathscr{p}_{v_j}^+$ is not more than $r$. ■

***Theorem 2.2*** The minimum number of RPs that are necessary and sufficient to serve all round-trip triples is

$$m = \left\lceil \frac{l(v_n)}{r} \right\rceil. \tag{2.4}$$

*Proof of Theorem 2.2* By proposition 2.2, we know that to compute the minimum number of RPs $m$, it is sufficient to only consider the round trips $(v_0, v_n, v_0)$ and $(v_n, v_0, v_n)$. Now suppose that we are going to locate the RPs in the following fashion:

- If $mod(l(v_n), r) \neq 0$, we locate RPs at distances $\frac{r}{2}, \frac{3r}{2}, \cdots, \frac{r}{2} + \left\lfloor \frac{l(v_n)-r}{r} \right\rfloor \times r$, $l(v_n) - \frac{r}{2}$ from node $v_0$;
- Otherwise, we locate RPs at distances $\frac{r}{2}, \frac{3r}{2}, \cdots, \left\lfloor \frac{l(v_n)-r/2}{r} \right\rfloor \times r$ from node $v_0$.

Then, it is easy to see that flows on round trip triple $(v_0, v_n, v_0)$ and flows on $(v_n, v_0, v_n)$ can be refueled. Hence,

$$m \leq \begin{cases} \left\lfloor \frac{l(v_n)}{r} \right\rfloor + 1, & if\ mod(l(v_n), r) \neq 0 \\ \left\lfloor \frac{l(v_n)}{r} \right\rfloor, & otherwise \end{cases},$$

which gives us an upper bound of $m$.

We then find a lower bound of $m$. Consider the following two cases:

a) To refuel flows on $(v_0, v_n, v_0)$.

For the outbound trip, $\wp_k$ should be located within $kr$ distance from $v_0$ and within $(m - k + 1)r$ to $v_n$. For the inbound trip, $\wp_m$ should be located within $\frac{r}{2}$ distance from $v_n$ and within $mr$ to $v_0$. Consequently, $\wp_k$ should be located within $\frac{r}{2} + (m - k)r$ distance from $v_n$ and $kr$ distance to $v_0$. Therefore,

$$S_1 = \left[ \max\left\{ l(v_n) - \frac{r}{2} - (m-1)r, 0 \right\}, r \right],$$

$S_k = \left[l(v_n) - \frac{r}{2} - (m-k)r, kr\right]$, for $1 < k < m$, and

$S_m = \left[l(v_n) - \frac{r}{2}, \min\{mr, l(v_n)\}\right]$.

b) To refuel flows on $(v_n, v_0, v_n)$.

For the outbound trip, $\wp_k$ should be located within $\big(m-(k-1)\big)r$ distance from $v_n$ and within $kr$ to $v_0$. For the inbound trip, $\wp_1$ should be located within $\frac{r}{2}$ from $v_0$ and within $mr$ to $v_n$. Consequently, $\wp_k$ should be located within $\frac{r}{2} + (k-1)r$ from $v_0$ and within $(m-k+1)r$ distance to $v_n$. Therefore,

$S_1 = \left[\max\{l(v_n) - mr, 0\}, \frac{r}{2}\right]$,

$S_k = \left[l(v_n) - \big(m-(k-1)\big)r, \frac{r}{2} + (k-1)r\right]$, for $1 < k < m$, and

$S_m = \left[l(v_n) - r, \min\left\{\frac{r}{2} + (m-1)r, l(v_n)\right\}\right]$.

By combining a) and b), we obtain $S_k = \left[l(v_n) - \frac{r}{2} - (m-k)r, \frac{r}{2} + (k-1)r\right]$.

By letting $l(v_n) - \frac{r}{2} - (m-k)r \le \frac{r}{2} + (k-1)r$, we derive $m \ge \frac{l(v_n)}{r}$, i.e., integer $m \ge \left\lceil\frac{l(v_n)}{r}\right\rceil$, which gives us a lower bound of $m$.

Therefore, we have

$$\left\lceil\frac{l(v_n)}{r}\right\rceil \le m \le \begin{cases} \left\lfloor\frac{l(v_n)}{r}\right\rfloor + 1, & if\ mod(l(v_n), r) \neq 0 \\ \left\lfloor\frac{l(v_n)}{r}\right\rfloor, & otherwise \end{cases}.$$

If $mod(l(v_n), r) \neq 0$, $\left\lceil\frac{l(v_n)}{r}\right\rceil = \left\lfloor\frac{l(v_n)}{r}\right\rfloor + 1$, otherwise if $mod(l(v_n), r) \equiv 0$, $\left\lceil\frac{l(v_n)}{r}\right\rceil = \left\lfloor\frac{l(v_n)}{r}\right\rfloor$.

Equation (2.4) $m = \left\lceil\frac{l(v_n)}{r}\right\rceil$ has been proved. ■

As a by-product of Theorem 2.2, we can derive the localization segment $S_k$ for each $\wp_k$.

***Corollary 2.2*** Given that $m$ RPs are to be established on the line, then

$$S_1 = \left[\max\left\{l(v_n) - \frac{r}{2} - (m-1)r, 0\right\}, \frac{r}{2}\right],$$

$$S_k = \left[l(v_n) - \frac{r}{2} - (m-k)r, \frac{r}{2} + (k-1)r\right], 1 < k < m, \text{ and}$$

$$S_m = \left[l(v_n) - \frac{r}{2}, \min\left\{\frac{r}{2} + (m-1)r, l(v_n)\right\}\right]. \quad (2.5)$$

***Observation 2.1*** Let $\hat{d}$ denote the distance between the right endpoint of localization segment $S_k$ and the left endpoint of localization segment $S_{k+1}$, where $1 \leq k < m$. Then by Corollary 2.2, we have

$$\begin{aligned}\hat{d} &= \alpha_{k+1} - \beta_k \\ &= \left(l(v_n) - \frac{r}{2} - \big(m - (k+1)\big)r\right) - \left(\frac{r}{2} + (k-1)r\right) \\ &= l(v_n) - (m-1)r \\ &= r - \left(\left\lceil\frac{l(v_n)}{r}\right\rceil - \frac{l(v_n)}{r}\right)r.\end{aligned}$$

Therefore, $0 < \hat{d} \leq r$.

***Observation 2.2*** We can observe and refer to a node $v \in V$ an *internal node* if it is either an interior point or a boundary point of a localization segment, otherwise, we will call $v$ is an *external node*.

Furthermore, we can decompose the node set $V$ into two disjoint subsets $V_{in}$ and $V_{ex}$, where $V_{in} = \bigcup_{v \in V: v \text{ is an internal node}}\{v\}$ and $V_{ex} = \bigcup_{v \in V: v \text{ is an external node}}\{v\} = V \backslash V_{in}$.

***Observation 2.3*** Let $v \in V_{ex}$ be an external node, then both $\tau(\wp_v^-)$ and $\tau(\wp_v^+)$ are known to us, given the localization segments. However, let $u \in V_{in}$ be an internal node, then without knowing the exact RP locations, $\tau(\wp_u^-)$ and $\tau(\wp_u^+)$ cannot be determined.

Observations 2 and 3 will shortly be illustrated in Figure 2.4 to follow.

Consider an internal node $u$ and assume that $u$ is an interior point of localization segment $S_k$, then we have

$$\tau(\wp_u^-), \tau(\wp_u^+) = \begin{cases} k, k+1, \ if \ \wp_k \prec u \\ k-1, k, \ if \ u \prec \wp_k \\ k, k, \ if \ l(u) = l(\wp_k) \end{cases}.$$

For notational convenience, for such an internal node $u$, we let $\wp_u^=$ denote the RP of which the localization segment "covers" node $u$. By using the word "cover", we mean that $u$ is either an interior point or a boundary point of the segment $S_{\tau(\wp_u^=)}$.

EXAMPLE 2.3 Consider the 4-node line network in figure 2.4, where $b(v_0, v_1) = 13$, $b(v_1, v_2) = 3$, $b(v_2, v_3) = 8$, $b(v_3, v_4) = 8$ and $r = 7$.

By equation (2.4), we can compute the minimum number of RPs that are needed, $m = \left\lceil \frac{l(v_4)}{r} \right\rceil = \left\lceil \frac{32}{7} \right\rceil = 5$. By equations in (2.5), we can compute the localization segments:

$$S_1 = \left[\max\left\{l(v_4) - \frac{7}{2} - (5-1) \times 7, 0\right\}, \frac{7}{2}\right] = [0.5,3.5],$$

$$S_2 = \left[l(v_4) - \frac{7}{2} - (5-2) \times 7, \frac{7}{2} + (2-1) \times 7\right] = [7.5,10.5],$$

$$S_3 = \left[l(v_4) - \frac{7}{2} - (5-3) \times 7, \frac{7}{2} + (3-1) \times 7\right] = [14.5,17.5],$$

$$S_4 = \left[l(v_4) - \frac{7}{2} - (5-4) \times 7, \frac{7}{2} + (4-1) \times 7\right] = [21.5,24.5],$$

$$S_5 = \left[l(v_4) - \frac{7}{2}, \min\left\{\frac{7}{2} + (5-1) \times 7, l(v_4)\right\}\right] = [28.5,31.5].$$

As shown in figure 2.4, the localization segments are indicated by five black bold segments. Also, we know that $V_{ex} = \{v_0, v_1, v_4\}$ and $V_{in} = \{v_2, v_3\}$. To illustrate

observation 2.3, consider node $v_1$, then $\tau(\wp_{v_1}^-) = 2$ and $\tau(\wp_{v_1}^+) = 3$. However, for node $v_2$, we are not able to tell whether $\tau(\wp_{v_2}^-)$ is equal to 2 or 3 at this moment.

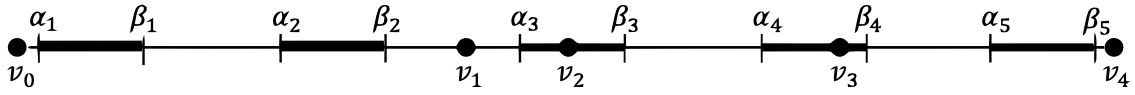



**Figure 2.4** Line network for example 2.3

### 2.2.2. Math programming formulation

Given a line network, given the traffic flow volume of each round trip, and given the vehicle range limit, in order to minimize the total travel distance (i.e., the total refueling detouring distance), by fixing the total number of RPs to be located equal to the minimum number $m$, we can formulate this line problem as a mixed-integer program with linear constraints and quadratic objective function, and solve the problem using the OPTI toolbox in MATLAB.

Define the continuous decision variable $x_k$ for $\wp_k \in \mathcal{P}$ and $k \in \{1, 2, \cdots, m\}$ as the position on the line at which $\wp_k$ is to be established. Let $\boldsymbol{x} = (x_1, x_2, \cdots, x_m)$ represent these $m$ locations. Then the objective function can be written as

$$\min_{\boldsymbol{x}} Z(\boldsymbol{x}) = \sum_{v_j \in V} \left( \sum_{\substack{v_i : v_i < v_j \\ d(v_i, v_j) > r/2}} f_{iji} * \left(1 - I_{\mathbb{L}_j}\left(x_{\tau(\wp_{v_j}^-)}\right)\right) * \left(2d\left(v_j, \wp_{v_j}^+\right)\right) + \right.$$

$$\left. \sum_{\substack{v_k : v_k > v_j \\ d(v_j, v_k) > r/2}} f_{jkj} * \left(1 - I_{\mathbb{R}_j}\left(x_{\tau(\wp_{v_j}^+)}\right)\right) * \left(2d\left(v_j, \wp_{v_j}^-\right)\right) \right), \tag{2.6}$$

where:

$f_{iji} = f(v_i, v_j, v_i)$ denotes the flow volume of round trip $(v_i, v_j, v_i)$;

$\mathbb{L}_j = \left[l(v_j) - \frac{r}{2}, l(v_j)\right]$ denotes a segment on the line network that contains all points on the left-hand side of node $v_j$ and within $\frac{r}{2}$ distance from $v_j$;

$\mathbb{R}_j = \left[l(v_j), l(v_j) + \frac{r}{2}\right]$ denotes another segment on the line network that contains all points on the right-hand side of node $v_j$ and within $\frac{r}{2}$ distance from $v_j$; and

$I_A(x) = \begin{cases} 1, & if\ x \in A \\ 0, & otherwise \end{cases}$ is an indicator function.

Consider any round trip $(v_i, v_j, v_i)$ with $v_i \prec v_j$, the function $1 - I_{\mathbb{L}_j}\left(x_{\tau\left(\not{p}_{v_j}^-\right)}\right)$ can be interpreted as whether or not the electric vehicle will need to make a refueling detour to $\not{p}_{v_j}^+$ after arriving at $v_j$. Then, $1 - I_{\mathbb{L}_j}\left(x_{\tau\left(\not{p}_{v_j}^-\right)}\right) =$ $\begin{cases} 0, & if\ \not{p}_{v_j}^- \in \left[l(v_j) - \frac{r}{2}, l(v_j)\right] \\ 1, & otherwise \end{cases}$. If this refueling detouring occurs, the corresponding detouring distance is two times the distance between $v_j$ and $\not{p}_{v_j}^+$, i.e., $2d\left(v_j, \not{p}_{v_j}^+\right)$.

Likewise, consider any round trip $(v_k, v_j, v_k)$ with $v_j \prec v_k$, the function $1 - I_{\mathbb{R}_j}\left(x_{\tau\left(\not{p}_{v_j}^+\right)}\right)$ can be interpreted as whether or not the electric vehicle will need to make a refueling detour to $\not{p}_{v_j}^-$ after arriving at $v_j$. Then, $1 - I_{\mathbb{R}_j}\left(x_{\tau\left(\not{p}_{v_j}^+\right)}\right) =$ $\begin{cases} 0, & if\ \not{p}_{v_j}^+ \in \left[l(v_j), l(v_j) + \frac{r}{2}\right] \\ 1, & otherwise \end{cases}$, and the associated detouring distance is $2d\left(v_j, \not{p}_{v_j}^-\right)$.

From $\left(1 - I_{\mathbb{L}_j}\left(x_{\tau\left(\not{p}_{v_j}^-\right)}\right)\right) * \left(2d\left(v_j, \not{p}_{v_j}^+\right)\right)$ and $\left(1 - I_{\mathbb{R}_j}\left(x_{\tau\left(\not{p}_{v_j}^+\right)}\right)\right) * \left(2d\left(v_j, \not{p}_{v_j}^-\right)\right)$, we can easily see that whether or not the electric vehicle will need to make a refueling detour, and what the associated detouring distance exactly is, are independent of the origin node and the given destination node $v_j$. Then for notational convenience, we

let $f_j^- = \sum_{\substack{v_i : v_i < v_j \\ d(v_i, v_j) > r/2}} f_{iji}$ and $f_j^+ = \sum_{\substack{v_k : v_k > v_j \\ d(v_j, v_k) > r/2}} f_{kjk}$. We can rewrite the objective (2.6) as

$$\min_{\boldsymbol{x}} Z(\boldsymbol{x}) = 2 * \sum_{v_j \in V} \left( f_j^- * \left( 1 - I_{\mathbb{L}_j}\left( x_{\tau\left(\wp_{v_j}^-\right)} \right) \right) * d\left( v_j, \wp_{v_j}^+ \right) + f_j^+ * \right.$$

$$\left. \left( 1 - I_{\mathbb{R}_j}\left( x_{\tau\left(\wp_{v_j}^+\right)} \right) \right) * d\left( v_j, \wp_{v_j}^- \right) \right). \tag{2.7}$$

Now we replace the two indicator functions in (2.7). Define two binary decision variables $y_{v_j}^-$ and $y_{v_j}^+$ for $v_j \in V$ as whether or not the electric vehicle will need to make a refueling detour during its round trip with origin on the left-hand side of $v_j$ and with origin on the right-hand side of $v_j$, respectively. That is,

$$y_{v_j}^- = \begin{cases} 1, \text{ if refueling detouring occurs for any round trip } \left(v_i, v_j, v_i\right), \text{where } v_i \prec v_j \\ \qquad\qquad \left( \Leftrightarrow x_{\tau\left(\wp_{v_j}^-\right)} < l\left(v_j\right) - \frac{r}{2} \right) \\ 0, \text{ otherwise} \end{cases} \tag{2.8}$$

$$y_{v_j}^+ = \begin{cases} 1, \text{ if refueling detouring occurs for any round trip } \left(v_k, v_j, v_k\right), \text{where } v_k \succ v_j \\ \qquad\qquad \left( \Leftrightarrow x_{\tau\left(\wp_{v_j}^+\right)} > l\left(v_j\right) + \frac{r}{2} \right) \\ 0, \text{ otherwise} \end{cases} \tag{2.9}$$

The mixed integer program to locate the RPs is now the following:

Minimize $Z(\boldsymbol{x})$

Subject to

$$\alpha_k \le x_k \le \beta_k \qquad 1 \le k \le m \tag{2.10}$$

$$x_{k+1} - x_k \le r \qquad 1 \le k < m \tag{2.11}$$

$$y_{v_j}^- + y_{v_j}^+ \le 1 \qquad \forall v_j \in V \tag{2.12}$$

$$x_{\tau(\wp_{v_j}^-)} \ge l\left(v_j\right) - \frac{r}{2} - M y_{v_j}^- \qquad \forall v_j \in V \tag{2.13}$$

$$x_{\tau(\wp_{v_j}^-)} \le l\left(v_j\right) - \frac{r}{2} + M\left(1 - y_{v_j}^-\right) \qquad \forall v_j \in V \tag{2.14}$$

$$x_{\tau(\wp_{v_j}^+)} \le l(v_j) + \frac{r}{2} + M y_{v_j}^+ \qquad \forall v_j \in V \tag{2.15}$$

$$x_{\tau(\wp_{v_j}^+)} \ge l(v_j) + \frac{r}{2} - M\left(1 - y_{v_j}^+\right) \qquad \forall v_j \in V \tag{2.16}$$

$$y_{v_j}^-, y_{v_j}^+ \in \{0,1\} \qquad \forall v_j \in V \tag{2.17}$$

Constraint (2.11) ensures the distance between every two consecutive RPs on the line is not more than the range limit $r$. Constraint (2.12) ensures that given any node $v_j$, refueling detouring will occur for either round trip $(v_i, v_j, v_i)$ or round trip $(v_k, v_j, v_k)$, but not both, where $v_i \prec v_j \prec v_k$. Suppose that we have $y_{v_j}^- + y_{v_j}^+ > 1$, then the distance between these two consecutive RPs, $\wp_{v_j}^-$ and $\wp_{v_j}^+$, would be greater than $r$. Constraints (2.13) and (2.14) ensure that if no refueling detouring will occur for a round trip $(v_i, v_j, v_i)$ where $v_i \prec v_j$, i.e., $y_{v_j}^- = 0$, then $\wp_{v_j}^-$ should be established within $\frac{r}{2}$ distance from node $v_j$. Constraints (2.15) and (2.16) ensure that if no refueling detouring will occur for a round trip $(v_k, v_j, v_k)$ where $v_k \succ v_j$, i.e., $y_{v_j}^+ = 0$, then $\wp_{v_j}^+$ should be established within $\frac{r}{2}$ distance from node $v_j$.

For any external node $u \in V_{ex}$, the indices $\tau(\wp_u^-)$ and $\tau(\wp_u^+)$ are determined, that is, they will stay the same no matter where $\wp_u^-$ and $\wp_u^+$ are to be established within their corresponding localization segments. Then, we can rewrite the total refueling detouring distance associated with external nodes, denoted $Z_{ex}(\boldsymbol{x})$, by replacing $d(u, \wp_u^+)$ with $x_{\tau(\wp_u^+)} - l(u)$ and replacing $d(u, \wp_u^-)$ with $l(u) - x_{\tau(\wp_u^-)}$. Hence, we have

$$Z_{ex}(\boldsymbol{x}) = 2 * \sum_{v_j \in V_{ex} \subseteq V} \left( f_j^- * y_{v_j}^- * \left( x_{\tau(\wp_{v_j}^+)} - l(v_j) \right) + f_j^+ * y_{v_j}^+ * \left( l(v_j) - x_{\tau(\wp_{v_j}^-)} \right) \right) \tag{2.18}$$

However, we should note that there is a problem with this formulation. Recall observation 2.3, we know that for an internal node $u \in V_{in}$ the indices $\tau(\wp_u^-)$ and $\tau(\wp_u^+)$ can only be determined after knowing the exact RP locations, i.e., after deriving the solution. Hence, we shall revise the above formulation for internal nodes.

Given the $m$ localization segments, let us consider two cases: $\frac{r}{2} < \hat{d} \leq r$ and $\hat{d} \leq \frac{r}{2}$, where $\hat{d}$ was defined in observation 2.1, representing the distance between the right endpoint of some localization segment $S_k$ and the left endpoint of $S_{k+1}$.

(a) $\frac{r}{2} < \hat{d} \leq r$

Consider any internal node $v_j \in V_{in}$, which is covered by localization segment $S_{\tau(\wp_{\bar{v}_j})}$. Then we have

$$l(v_j) - \frac{r}{2} \geq \alpha_{\tau(\wp_{\bar{v}_j})} - \frac{r}{2} > \alpha_{\tau(\wp_{\bar{v}_j})} - \hat{d} = \beta_{\tau(\wp_{\bar{v}_j})-1} \geq l\left(\wp_{\tau(\wp_{\bar{v}_j})-1}\right), \text{ and}$$

$$l(v_j) + \frac{r}{2} \leq \beta_{\tau(\wp_{\bar{v}_j})} + \frac{r}{2} < \beta_{\tau(\wp_{\bar{v}_j})} + \hat{d} = \alpha_{\tau(\wp_{\bar{v}_j})+1} \leq l\left(\wp_{\tau(\wp_{\bar{v}_j})+1}\right),$$

that is, no matter where the two RPs, $\wp_{\tau(\wp_{\bar{v}_j})-1}$ and $\wp_{\tau(\wp_{\bar{v}_j})+1}$, are to be established on the line, they are beyond $\frac{r}{2}$ distance from node $v_j$, as illustrated in figure 2.5.

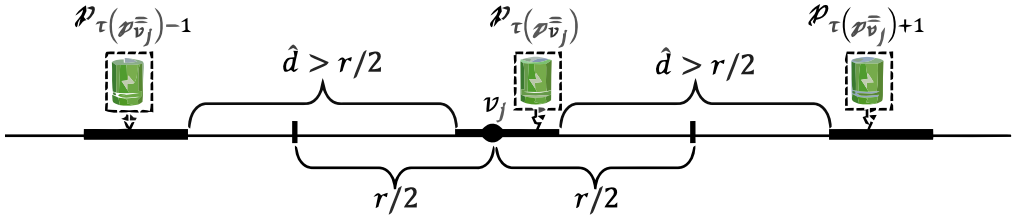



**Figure 2.5** An illustration for case (a) $\frac{r}{2} < \hat{d} \leq r$

Then,

- if $l\left(\wp_{\tau\left(\wp^{=}_{v_j}\right)}\right) = l(v_j)$: $y^{-}_{v_j} = y^{+}_{v_j} = 0$. That is, no refueling detouring will occur for both round trip $(v_i, v_j, v_i)$ where $v_i \prec v_j$ and round trip $(v_k, v_j, v_k)$ where $v_k \succ v_j$;
- if $\wp^{-}_{v_j} = \wp^{=}_{v_j}$: $y^{-}_{v_j} = 0$ and $y^{+}_{v_j} = 1$. That is, refueling detouring will occur for any round trip $(v_k, v_j, v_k)$ where $v_k \succ v_j$, the electric vehicle will have to make a detour to $\wp^{=}_{v_j}$ and the associated detouring distance is $2\left(l(v_j) - x_{\tau(\wp^{=}_{v_j})}\right)$;
- if $\wp^{+}_{v_j} = \wp^{=}_{v_j}$: $y^{-}_{v_j} = 1$ and $y^{+}_{v_j} = 0$. That is, refueling detouring will occur for any round trip $(v_i, v_j, v_i)$ where $v_i \prec v_j$, the electric vehicle will have to make a detour to $\wp^{=}_{v_j}$ and the associated detouring distance is $2\left(x_{\tau(\wp^{=}_{v_j})} - l(v_j)\right)$. See figure 2.5 as an illustration.

Therefore, the total refueling detouring distance associated with all internal nodes is

$$Z_{in}(\boldsymbol{x}) = 2 * \sum_{v_j \in V_{in}} \left( f_j^{-} * y^{-}_{v_j} * \left( x_{\tau(\wp^{=}_{v_j})} - l(v_j) \right) + f_j^{+} * y^{+}_{v_j} * \left( l(v_j) - x_{\tau(\wp^{=}_{v_j})} \right) \right) \quad (2.19)$$

The constrains associated with internal nodes are now the following:

$$y^{-}_{v_j} + y^{+}_{v_j} = 1 \qquad \forall v_j \in V_{in} \qquad (2.20.1)$$

$$x_{\tau(\wp^{=}_{v_j})} \leq l(v_j) + M y^{-}_{v_j} \qquad \forall v_j \in V_{in} \qquad (2.20.2)$$

$$x_{\tau(\wp^{=}_{v_j})} \geq l(v_j) - M\left(1 - y^{-}_{v_j}\right) \qquad \forall v_j \in V_{in} \qquad (2.20.3)$$

$$x_{\tau(\wp^{=}_{v_j})} \geq l(v_j) - M y^{+}_{v_j} \qquad \forall v_j \in V_{in} \qquad (2.20.4)$$

$$x_{\tau(\wp^{=}_{v_j})} \leq l(v_j) + M\left(1 - y^{+}_{v_j}\right) \qquad \forall v_j \in V_{in} \qquad (2.20.5)$$

$$y^{-}_{v_j}, y^{+}_{v_j} \in \{0,1\} \qquad \forall v_j \in V_{in} \qquad (2.20.6)$$

(b) $\hat{d} \leq \frac{\boldsymbol{r}}{\mathbf{2}}$

Case (b) would be a bit more complicated than case (a). In this case, the length of a localization segment is greater than or equal to $\frac{r}{2}$, and it is possible that both $\mathcal{p}_{\tau\left(\mathcal{p}_{v_j}^{-}\right)}$ and $\mathcal{p}_{\tau\left(\mathcal{p}_{v_j}^{+}\right)}$ are established within $\frac{r}{2}$ distance from node $v_j$. As we can see from figure 2.6, where $\mathcal{p}_{v_j}^{-} = \mathcal{p}_{v_j}^{=}$ is within $\frac{r}{2}$ distance from $v_j$ and $\mathcal{p}_{v_j}^{+} = \mathcal{p}_{\tau\left(\mathcal{p}_{v_j}^{=}\right)+1}$ is within $\frac{r}{2}$ distance from $v_j$, then $y_{v_j}^{-} = y_{v_j}^{+} = 0$.

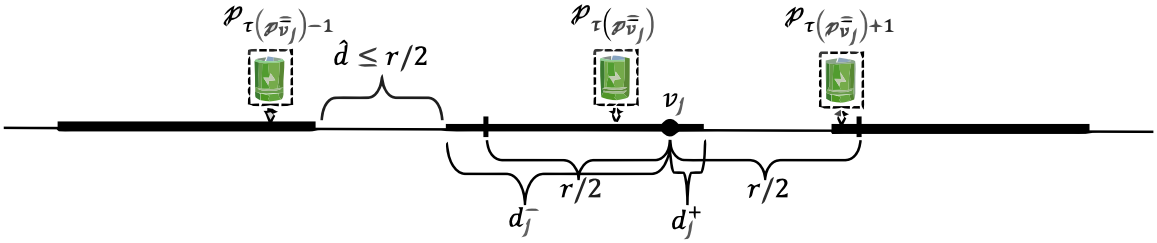



**Figure 2.6** An illustration for case (b)

Let $d_j^{-}$ denote the distance between the internal node $v_j$ and the left endpoint of the localization segment $S_{\tau\left(\mathcal{p}_{v_j}^{=}\right)}$ that covers $v_j$, and let $d_j^{+}$ denote the distance between $v_j$ and the right endpoint of $S_{\tau\left(\mathcal{p}_{v_j}^{=}\right)}$.

Furthermore, let us consider three sub-cases: (b1) $d_j^{-} \geq \frac{r}{2}, d_j^{+} < \frac{r}{2}$; (b2) $d_j^{-} < \frac{r}{2}, d_j^{+} \geq \frac{r}{2}$; and (b3) $d_j^{-} < \frac{r}{2}, d_j^{+} < \frac{r}{2}$, where in each case we are able to avoid the problem of not knowing indices $\tau\left(\mathcal{p}_{v_j}^{-}\right)$ and $\tau\left(\mathcal{p}_{v_j}^{+}\right)$ for an internal node $v_j$.

(b1) $d_j^{-} \geq \frac{r}{2}$ and $d_j^{+} < \frac{r}{2}$

- If $\mathcal{p}_{v_j}^{+} = \mathcal{p}_{v_j}^{=}$, then for any round trip $(v_i, v_j, v_i)$ where $v_i \prec v_j$, the electric vehicle will need to make a refueling detour to $\mathcal{p}_{v_j}^{+}$ since by $d_j^{-} \geq \frac{r}{2}$ we know that $\mathcal{p}_{v_j}^{-} = \mathcal{p}_{\tau\left(\mathcal{p}_{v_j}^{=}\right)-1}$ is established beyond $\frac{r}{2}$ distance from $v_j$. For any round trip $(v_k, v_j, v_k)$

where $v_k \succ v_j$, no refueling detouring will occur since by $d_j^+ < \frac{r}{2}$ we know that $\wp_{v_j}^+ = \wp_{\bar{v}_j}^=$ is established within $\frac{r}{2}$ distance from $v_j$. That is, $y_{v_j}^- = 1$ and $y_{v_j}^+ = 0$. See illustration in figure 2.7(a).

- If $\wp_{v_j}^- = \wp_{\bar{v}_j}^=$ and $\wp_{\bar{v}_j}^=$ is established beyond $\frac{r}{2}$ distance from $v_j$, then $y_{v_j}^- = 1$ and by $y_{v_j}^- + y_{v_j}^+ \leq 1$ we shall have $y_{v_j}^+ = 0$. See illustration in figure 2.7(b).
- If $\wp_{v_j}^- = \wp_{\bar{v}_j}^=$ and $\wp_{\bar{v}_j}^=$ is established within $\frac{r}{2}$ distance from $v_j$, then $y_{v_j}^- = 0$. But $y_{v_j}^+$ can be either 0 or 1, depending on the distance between $v_j$ and $\wp_{v_j}^+$, that is, $y_{v_j}^+ = 1$ if $d\left(v_j, \wp_{v_j}^+\right) > \frac{r}{2}$, otherwise $y_{v_j}^+ = 0$. See illustration in figure 2.6 and figure 2.7(c).

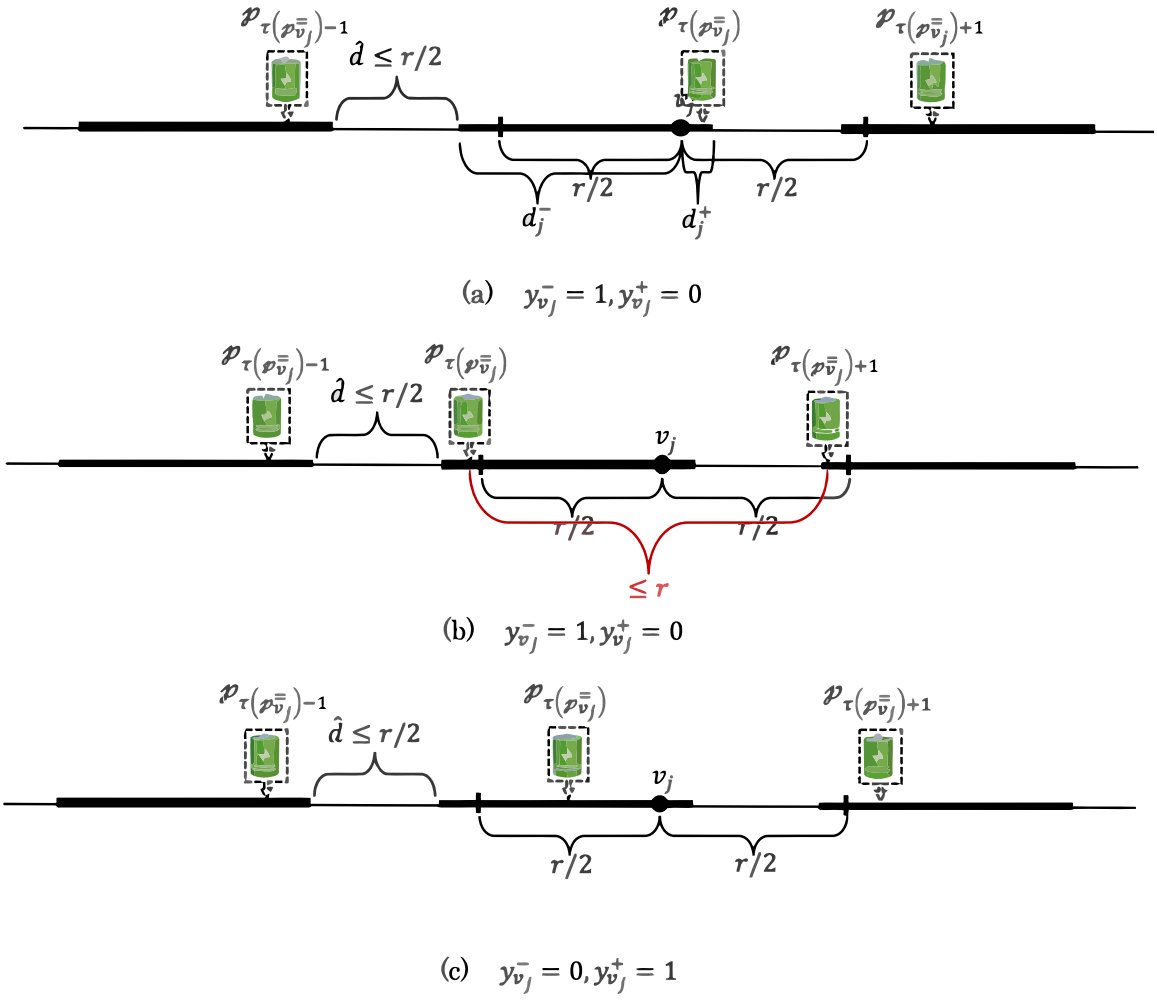



**Figure 2.7** An illustration for sub-cases (b1), (b2) and (b3)

Define three binary variables, $\eta_{v_j}^1$, $\eta_{v_j}^2$ and $\eta_{v_j}^3$:

$\eta_{v_j}^1 = \begin{cases} 1, & if\ x_{\tau(\overline{p}_{v_j})} < l(v_j) - \frac{r}{2} \\ 0, & otherwise \end{cases}$, indicating whether or not $\overline{p}_{v_j}$ is established on the left-hand side of $v_j$ and beyond $\frac{r}{2}$ distance from $v_j$; (2.21.1)

$\eta_{v_j}^2 = \begin{cases} 1, & if\ x_{\tau(\overline{p}_{v_j})} > l(v_j) \\ 0, & otherwise \end{cases}$, indicating whether or not $\overline{p}_{v_j}$ is established on the left-hand side of $v_j$; (2.21.2)

$\eta_{v_j}^3 = \begin{cases} 1, \ if \ x_{\tau\left(\not{p}_{v_j}^{=}\right)+1} < l(v_j) + \frac{r}{2} \\ 0, \ otherwise \end{cases}$, indicating whether or not $\not{p}_{\tau\left(\not{p}_{v_j}^{=}\right)+1}$ is established beyond $\frac{r}{2}$ distance from $v_j$. (2.21.3)

Then with these three variables, we shall be able to see that in case (b1) we could avoid the problem of not knowing indices $\tau\left(\not{p}_{v_j}^{-}\right)$ and $\tau\left(\not{p}_{v_j}^{+}\right)$ for an internal node $v_j$. Since if refueling detouring occurs for a round trip $(v_i, v_j, v_i)$ where $v_i \prec v_j$, when $x_{\tau\left(\not{p}_{v_j}^{=}\right)} < l(v_j) - \frac{r}{2}$ (i.e., $w_{v_j}^1 = 1$), the electric vehicle will need to make a detour to $\not{p}_{\tau\left(\not{p}_{v_j}^{=}\right)+1}$, and when $x_{\tau\left(\not{p}_{v_j}^{=}\right)} > l(v_j)$ (i.e., $w_{v_j}^2 = 1$), the electric vehicle will need to make a detour to $\not{p}_{v_j}^{=}$. And if refueling detouring occurs for a round trip $(v_k, v_j, v_k)$ where $v_k \succ v_j$, then the vehicle will need to make a detour to $\not{p}_{v_j}^{=}$. Let $V'_{in} = \left\{v_j \in V_{in} : d_j^- \geq \frac{r}{2}, d_j^+ < \frac{r}{2}\right\}$. We can rewrite the total refueling detouring distance associated with internal nodes in set $V'_{in}$ as

$$Z'_{in}(\boldsymbol{x}) = 2 * \sum_{v_j \in V'_{in}} \left( f_j^- * \eta_{v_j}^1 * \left( x_{\tau\left(\not{p}_{v_j}^{=}\right)+1} - l(v_j) \right) + f_j^- * \eta_{v_j}^2 * \left( x_{\tau\left(\not{p}_{v_j}^{=}\right)} - l(v_j) \right) + f_j^+ * y_{v_j}^+ * \left( l(v_j) - x_{\tau\left(\not{p}_{v_j}^{=}\right)} \right) \right). \quad (2.22)$$

Now we can construct the constraints as:

$$x_{\tau\left(\not{p}_{v_j}^{=}\right)} \geq l(v_j) - \frac{r}{2} - M\eta_{v_j}^1 \qquad \forall v_j \in V'_{in} \quad (2.23.1)$$

$$x_{\tau\left(\not{p}_{v_j}^{=}\right)} \leq l(v_j) - \frac{r}{2} + M\left(1 - \eta_{v_j}^1\right) \qquad \forall v_j \in V'_{in} \quad (2.23.2)$$

$$x_{\tau\left(\not{p}_{v_j}^{=}\right)} \leq l(v_j) + M\eta_{v_j}^2 \qquad \forall v_j \in V'_{in} \quad (2.23.3)$$

$$x_{\tau\left(\not{p}_{v_j}^{=}\right)} \geq l(v_j) - M\left(1 - \eta_{v_j}^2\right) \qquad \forall v_j \in V'_{in} \quad (2.23.4)$$

$$x_{\tau\left(\wp^{=}_{v_j}\right)+1} \geq l(v_j) + \frac{r}{2} - M\eta^3_{v_j} \qquad \forall v_j \in V'_{in} \qquad (2.23.5)$$

$$x_{\tau\left(\wp^{=}_{v_j}\right)+1} \leq l(v_j) + \frac{r}{2} + M\left(1 - \eta^3_{v_j}\right) \qquad \forall v_j \in V'_{in} \qquad (2.23.6)$$

$$\eta^1_{v_j} + \eta^2_{v_j} \leq 1 \qquad \forall v_j \in V'_{in} \qquad (2.23.7)$$

$$\eta^1_{v_j} \leq \eta^3_{v_j} \qquad \forall v_j \in V'_{in} \qquad (2.33.8)$$

$$y^+_{v_j} \leq 1 - \eta^3_{v_j} \qquad \forall v_j \in V'_{in} \qquad (2.23.9)$$

$$y^+_{v_j} \leq 1 - \eta^2_{v_j} \qquad \forall v_j \in V'_{in} \qquad (2.23.10)$$

$$\eta^1_{v_j} + \eta^2_{v_j} + y^+_{v_j} \leq 1 \qquad \forall v_j \in V'_{in} \qquad (2.23.11)$$

$$\eta^1_{v_j}, \eta^2_{v_j}, \eta^3_{v_j} \in \{0,1\} \qquad \forall v_j \in V'_{in} \qquad (2.23.12)$$

Constraint (2.23.7) ensures that we will not have $x_{\tau\left(\wp^{=}_{v_j}\right)} < l(v_j) - \frac{r}{2}$ and $x_{\tau\left(\wp^{=}_{v_j}\right)} > l(v_j)$ existing at the same time. Constraint (2.23.8) ensures that the distance between $\wp^{=}_{v_j}$ and $\wp_{\tau\left(\wp^{=}_{v_j}\right)+1}$ will not be greater than the range limit $r$. Constraint (2.23.9) ensures that if $\wp_{\tau\left(\wp^{=}_{v_j}\right)+1}$ is established within $\frac{r}{2}$ distance from $v_j$, then no refueling detouring will occur for any round trip $(v_k, v_j, v_k)$ where $v_k \succ v_j$. Constraint (2.23.10) ensures that if refueling detouring occurs for a round trip $(v_k, v_j, v_k)$ where $v_k \succ v_j$, then $\wp^{=}_{v_j}$ is established on the left-hand side of $v_j$. Constraint (2.23.11) is equivalent to $y^-_{v_j} + y^+_{v_j} \leq 1$ since it is easy to see that $\eta^1_{v_j} + \eta^2_{v_j} = y^-_{v_j}$.

Now let us consider case (b2).

(b2) $d^-_j < \frac{r}{2}$ and $d^+_j \geq \frac{r}{2}$

This case is symmetric to (b1), then we shall be able to safely skip the details. Again, we define three binary decision variables:

$$\lambda_{v_j}^1 = \begin{cases} 1, \ if \ x_{\tau\left(\not{p}_{\bar{v}_j}\right)} > l\left(v_j\right) \\ 0, \ otherwise \end{cases} \tag{2.24.1}$$

$$\lambda_{v_j}^2 = \begin{cases} 1, \ if \ x_{\tau\left(\not{p}_{\bar{v}_j}\right)} < l\left(v_j\right) + \frac{r}{2} \\ 0, \ otherwise \end{cases} \tag{2.24.2}$$

$$\lambda_{v_j}^3 = \begin{cases} 1, \ if \ x_{\tau\left(\not{p}_{\bar{v}_j}\right)-1} < l\left(v_j\right) - \frac{r}{2} \\ 0, \ otherwise \end{cases} \tag{2.24.3}$$

Let $V_{in}'' = \left\{v_j \in V_{in}: d_j^- < \frac{r}{2}, d_j^+ \geq \frac{r}{2}\right\}$. Then the total refueling detouring distance associated with the internal nodes in set $V_{in}''$ can be rewritten as

$$Z_{in}''(\boldsymbol{x}) = 2 * \sum_{v_j \in V_{in}''} \left( f_j^- * y_{v_j}^- * \left( x_{\tau(\not{p}_{\bar{v}_j})} - l\left(v_j\right) \right) + f_j^+ * \left(1 - \lambda_{v_j}^1\right) * \left( l\left(v_j\right) - x_{\tau\left(\not{p}_{\bar{v}_j}\right)} \right) + f_j^+ * \left(1 - \lambda_{v_j}^2\right) * \left( l\left(v_j\right) - x_{\tau\left(\not{p}_{\bar{v}_j}\right)-1} \right) \right). \tag{2.25}$$

The constraints are the following:

$$x_{\tau(\not{p}_{\bar{v}_j})} \geq l\left(v_j\right) - M\left(1 - \lambda_{v_j}^1\right) \qquad \forall v_j \in V_{in}'' \tag{2.26.1}$$

$$x_{\tau(\not{p}_{\bar{v}_j})} \leq l\left(v_j\right) + M\lambda_{v_j}^1 \qquad \forall v_j \in V_{in}'' \tag{2.26.2}$$

$$x_{\tau(\not{p}_{\bar{v}_j})} \leq l\left(v_j\right) + \frac{r}{2} + M\left(1 - \lambda_{v_j}^2\right) \qquad \forall v_j \in V_{in}'' \tag{2.26.3}$$

$$x_{\tau(\not{p}_{\bar{v}_j})} \geq l\left(v_j\right) + \frac{r}{2} - M\lambda_{v_j}^2 \qquad \forall v_j \in V_{in}'' \tag{2.26.4}$$

$$x_{\tau(\not{p}_{\bar{v}_j})-1} \leq l\left(v_j\right) - \frac{r}{2} + M\left(1 - \lambda_{v_j}^3\right) \qquad \forall v_j \in V_{in}'' \tag{2.26.5}$$

$$x_{\tau(\not{p}_{\bar{v}_j})-1} \geq l\left(v_j\right) - \frac{r}{2} - M\lambda_{v_j}^3 \qquad \forall v_j \in V_{in}'' \tag{2.26.6}$$

$$\lambda_{v_j}^1 + \lambda_{v_j}^2 \geq 1 \qquad \forall v_j \in V_{in}'' \tag{2.26.7}$$

$$\lambda_{v_j}^2 \geq \lambda_{v_j}^3 \qquad \forall v_j \in V_{in}'' \tag{2.26.8}$$

$$y_{v_j}^- \leq \lambda_{v_j}^1 \qquad \forall v_j \in V_{in}'' \tag{2.26.9}$$

$$y_{v_j}^- \leq \lambda_{v_j}^3 \qquad \forall v_j \in V_{in}'' \tag{2.26.10}$$

$$y_{v_j}^- + \left(1 - \lambda_{v_j}^1\right) + \left(1 - \lambda_{v_j}^2\right) \leq 1 \qquad \forall v_j \in V_{in}'' \qquad (2.26.11)$$

$$\lambda_{v_j}^1, \lambda_{v_j}^2, \lambda_{v_j}^3 \in \{0,1\} \qquad \forall v_j \in V_{in}'' \qquad (2.26.12)$$

Then let us consider case (b3).

(b3) $d_j^- < \frac{r}{2}$ and $d_j^+ < \frac{r}{2}$

Since $\left(d_j^- + d_j^-\right) + \hat{d} = r$, we have $d_j^- + \hat{d} > \frac{r}{2}$ and $d_j^+ + \hat{d} > \frac{r}{2}$. Then

- If $l\left(\wp_{\tau\left(\wp_{v_j}^{=}\right)}\right) = l(v_j)$: $y_{v_j}^- = y_{v_j}^+ = 0$;
- If $\wp_{v_j}^- = \wp_{v_j}^{=}$: by $d_j^- < \frac{r}{2}$ we know that $y_{v_j}^- = 0$, and by $d_j^+ + \hat{d} > \frac{r}{2}$ we know that $y_{v_j}^+ = 1$;
- If $\wp_{v_j}^+ = \wp_{v_j}^{=}$: by $d_j^+ < \frac{r}{2}$ we know that $y_{v_j}^+ = 0$, and by $d_j^- + \hat{d} > \frac{r}{2}$ we know that $y_{v_j}^- = 1$.

Let $V_{in}''' = \left\{v_j \in V_{in}: d_j^- < \frac{r}{2}, d_j^+ < \frac{r}{2}\right\}$. Then the total refueling detouring distance associated with the internal nodes in set $V_{in}'''$ can be rewritten as

$$Z_{in}'''(\boldsymbol{x}) = 2 * \sum_{v_j \in V_{in}'''} \left(f_j^- * y_{v_j}^- * \left(x_{\tau(\wp_{v_j}^{=})} - l(v_j)\right) + f_j^+ * y_{v_j}^+ * \left(l(v_j) - x_{\tau\left(\wp_{v_j}^{=}\right)}\right)\right) \qquad (2.27)$$

Recall our discussion in case (a), we shall see that case (b3) is similar to (a), since no matter where $\wp_{\tau\left(\wp_{v_j}^{=}\right)-1}$ and $\wp_{\tau\left(\wp_{v_j}^{=}\right)+1}$ are to be established on the line, they are beyond $\frac{r}{2}$ distance from node $v_j$. Therefore, we can construct the following constraints:

$$y_{v_j}^- + y_{v_j}^+ = 1 \qquad \forall v_j \in V_{in}''' \qquad (2.28.1)$$

$$x_{\tau(\wp_{v_j}^{=})} \leq l(v_j) + M y_{v_j}^- \qquad \forall v_j \in V_{in}''' \qquad (2.28.2)$$

$$x_{\tau(\wp_{v_j}^{=})} \geq l(v_j) - M\left(1 - y_{v_j}^-\right) \qquad \forall v_j \in V_{in}''' \qquad (2.28.3)$$

$$x_{\tau(\wp\bar{\bar{v}}_j)} \geq l(v_j) - My_{v_j}^{+} \qquad \forall v_j \in V_{in}''' \qquad (2.28.4)$$

$$x_{\tau(\wp\bar{\bar{v}}_j)} \leq l(v_j) + M\left(1 - y_{v_j}^{+}\right) \qquad \forall v_j \in V_{in}''' \qquad (2.28.5)$$

The mixed integer program with linear constraints and quadric is now the following:

**Case (a):** $\frac{r}{2} < \hat{d} \leq r$

| | |
|---|---|
| Minimize | $Z(\boldsymbol{x}) = Z_{ex}(\boldsymbol{x}) + Z_{in}(\boldsymbol{x})$ |
| Subject to | constraints (2.10)-(2.11) |
| | constraints (2.12)-(2.17) for external nodes in $V_{ex}$ |
| | constraints (2.20.1)-(2.20.6) for internal nodes in $V_{in}$ |

**Case (b):** $\hat{d} \leq \frac{r}{2}$

| | |
|---|---|
| Minimize | $Z(\boldsymbol{x}) = Z_{ex}(\boldsymbol{x}) + Z_{in}'(\boldsymbol{x}) + Z_{in}''(\boldsymbol{x}) + Z_{in}'''(\boldsymbol{x})$ |
| Subject to | constraints (2.10)-(2.11)` |
| | constraints (2.12)-(2.17) for external nodes in $V_{ex}$ |
| | constraints (2.23.1)-(2.23.12) for external nodes in $V_{in}'$ |
| | constraints (2.26.1)-(2.26.12) for internal nodes in $V_{in}'$ |
| | constraints (2.28.1)-(2.28.5) for internal nodes in $V_{in}'''$ |

Then we are able to solve the line problem using the OPTI toolbox in MATLAB.

***Theorem 2.3.*** $\wp_1$ can always be located at $\beta_1 = \frac{r}{2}$ where $\beta_1$ is the right endpoint of localization segment $S_1$, and $\wp_m$ can always be located at $\alpha_m = l(v_n) - \frac{r}{2}$ where $\alpha_m$ is the left endpoint of localization segment $S_m$.

*Proof of Theorem 2.3.* Given any feasible solution $\boldsymbol{x} = (x_1, \cdots, x_m)$ to the line problem, we obtain $\boldsymbol{x}'$ by repositioning $\wp_1$ at $\frac{r}{2}$ and $\wp_m$ at $l(v_n) - \frac{r}{2}$, then it is easy to

see that $\boldsymbol{x}'$ is feasible to the problem as well. Let $v_j$ be a node within $(x_1, x_1 + \frac{r}{2}]$ if any. Then if refueling detouring will occur for a round trip $(v_k, v_j, v_k)$ with $v_k \succ v_j$, by repositioning $\wp_1$, the detouring distance is $2 \times \max\left\{0, l(v_j) - \frac{r}{2}\right\}$, decreased. Let $v_j$ be a node within $(x_1 + \frac{r}{2}, r]$ if any. Then if refueling detouring will occur for a round trip $(v_i, v_j, v_i)$ with $v_i \prec v_j$, by repositioning $\wp_1$, refueling detouring will be no longer needed. Likewise, by repositioning $\wp_m$, the refueling detouring distance would be non-increasing as well. ■

EXAMPLE 2.4. Consider the same line network used in example 2.3. Assume that

| flow volume | $v_1$ | $v_2$ | $v_3$ |
|---|---|---|---|
| $f_j^-$ | 5 | 10 | 10 |
| $f_j^+$ | 10 | 10 | 5 |

One of the optimal solution given by the OPTI is $x_1 = 3.5, x_2 = 10, x_3 = 16, x_4 = 23, x_5 = 28.5$, and the total refueling detouring distance is 5.

### 2.2.3. Existence of finite dominating set

In this section, we will show that there exists a finite dominating set (FDS) to the line problem, i.e., a finite set of points where an optimal solution must belong.

#### 2.2.3.1. Set of breakpoints

For any node $v_j$, let ${v_j}^-(d)$ represent the point that is on the left-hand side of $v_j$ and at $d$ distance away from $v_j$, i.e., $l\left({v_j}^-(d)\right) = l(v_j) - d$, and let ${v_j}^+(d)$ represent the point that is on the right-hand side of $v_j$ and at $d$ distance away from $v_j$, i.e.,

$\left(v_j{}^+(d)\right) = l(v_j) + d$. Specifically, we say that $v_j{}^-\left(\frac{r}{2}\right)$ and $v_j{}^+\left(\frac{r}{2}\right)$ are two extreme none refueling detouring (XNRD) sites for $\mathscr{p}_{v_j}^-$ and $\mathscr{p}_{v_j}^+$, respectively. By "XNRD" we mean that $v_j{}^-\left(\frac{r}{2}\right)$ is the farthest allowable site on the left-hand side of $v_j$ such that no refueling detouring will occur for a round trip $(v_i, v_j, v_i)$ with $v_i \prec v_j$, and that $v_j{}^+\left(\frac{r}{2}\right)$ is the farthest allowable site on the right-hand side of $v_j$ such that no refueling detouring will occur for a round trip $(v_k, v_j, v_k)$ with $v_k \succ v_j$.

Define $\boldsymbol{\mathcal{B}}$ as the set of breakpoints, and $\boldsymbol{\mathcal{B}}$ is composed of the following four parts:

- $B_1 = \bigcup_{k=1}^{m}\{\alpha_k, \beta_k\}$, the set of endpoints of each localization segment;
- $B_2 = V_{in}$, the set of internal nodes;
- $B_3 = \left\{v_j{}^-\left(\frac{r}{2}\right), v_j{}^+\left(\frac{r}{2}\right) : v_j \in V\right\} \cap \{\bigcup_{k=1}^{m} S_k\}$, the set of XNRD sites that are either interior points or boundary points of some localization segments;
- $B_4 = \bigcup_{x\in B_1\cup B_2\cup B_3}\left\{\{x^-(ir), x^+(ir): i = 1, 2, \cdots, m\} \cap \{\bigcup_{k=1}^{m} S_k\}\right\}$. Suppose that a RP is to be established at some point $x \in B_1 \cup B_2 \cup B_3$, then with $B_4$, we are guaranteed to be able to identify a set of $m$ locations such that the distance between every two consecutive RPs is not more than the range limit $r$.

By identifying the set of breakpoints $\boldsymbol{\mathcal{B}}$, each localization segment can be further divided into several sub-segments. A segment is called *indivisible* if it does not contain any breakpoint as its interior point. Consider a localization segment $S_k$, let $n_k$ denote the number of breakpoints that are either interior points or boundary points of $S_k$. Then, $S_k$ can be decomposed into $n_k - 1$ indivisible sub- segments.

EXAMPLE 2.5 Consider the same line network used in example 2.3, where $b(v_0, v_1) = 13$, $b(v_1, v_2) = 3$, $b(v_2, v_3) = 8$, $b(v_3, v_4) = 8$ and $r = 7$. From example 2.3, we know that the five localization segments are $S_1 = [0.5,3.5]$, $S_2 = [7.5,10.5]$, $S_3 = [14.5,17.5]$, $S_4 = [21.5,24.5]$, and $S_5 = [28.5,31.5]$.

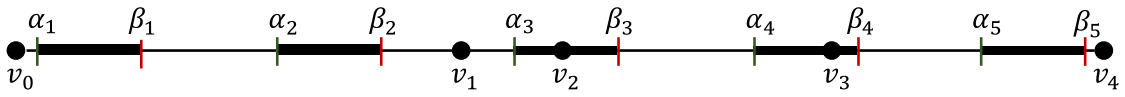



**Figure 2.8** A copy of Figure 2.4

Then,

$B_1 = \cup_{k=1}^{5}\{\alpha_k, \beta_k\} = \{0.5, 3.5\} \cup \{7.5, 10.5\} \cup \{14.5, 17.5\} \cup \{21.5, 24.5\} \cup \{28.5, 31.5\}$;

$B_2 = V_{in} = \{v_2, v_3\} = \{16, 24\}$;

$B_3 = \left\{v_j^{-}\left(\frac{r}{2}\right), v_j^{+}\left(\frac{r}{2}\right) : v_j \in V\right\} \cap \left\{\cup_{k=1}^{5} S_k\right\} = \{3.5, 9.5, 16.5, \cancel{12.5}, \cancel{19.5}, \cancel{20.5}, \cancel{27.5}, 28.5\}$;

$B_4 = \cup_{x \in B_1 \cup B_2 \cup B_3}\left\{\{x^{-}(ir), x^{+}(ir): i = 1, 2, \cdots, m\} \cap \left\{\cup_{k=1}^{5} S_k\right\}\right\} = B_1 \cup \{2, 9, 23, 30\} \cup \{3, 10, 17, 31\} \cup \{2.5, 16.5, 23.5, 30.5\}$.

Hence, we can see that

$S_1$ can be divided into 4 sub-segments: [0.5,2], [2,2.5], [2.5,3] and [3,3.5];

$S_2$ can be divided into 4 sub-segments: [7.5,9], [9,9.5], [9.5,10] and [10,10.5];

$S_3$ can be divided into 4 sub-segments: [14.5,16], [16,16.5], [16.5,17] and [17,17.5];

$S_4$ can be divided into 4 sub-segments: [21.5,23], [23,23.5], [23.5,24] and [24,25.5];

$S_5$ can be divided into 4 sub-segments: [28.5,30], [30,30.5], [30.5,31] and [31,31.5].

Now we claim that there exists an FDS to the line problem.

***Theorem 2.4*** $\mathcal{B}$ is an FDS to the line problem.

### 2.2.3.2. Another perspective on calculating refueling detouring distance

Before proceeding with the proof of theorem 2.4, let us introduce a different perspective on calculating the total refueling detouring distance. Recall in (2.6), the objective function is written as

$$Z(\boldsymbol{x}) = \sum_{v_j \in V} \left( \sum_{\substack{v_i : v_i \prec v_j \\ d(v_i, v_j) > r/2}} f_{iji} * \left( 1 - I_{\mathbb{L}_j} \left( x_{\tau\left(\wp_{v_j}^-\right)} \right) \right) * \left( 2d\left(v_j, \wp_{v_j}^+\right) \right) + \right.$$

$$\left. \sum_{\substack{v_k : v_k \succ v_j \\ d(v_j, v_k) > r/2}} f_{kjk} * \left( 1 - I_{\mathbb{R}_j} \left( x_{\tau\left(\wp_{v_j}^+\right)} \right) \right) * \left( 2d\left(v_j, \wp_{v_j}^-\right) \right) \right),$$

which is a summed over all round trips. Now, let us rewrite $Z(\boldsymbol{x})$ as

$$Z(\boldsymbol{x}) = \sum_{k=1}^{m} Z_k(\boldsymbol{x}), \tag{2.29}$$

where $Z_k(\boldsymbol{x})$ is the total refueling detouring distance associated with RP $\wp_k$. That is, the electric vehicle will need to make a detour to $\wp_k$ to complete its round trip on the line. Then we have,

$$Z_k(\boldsymbol{x}) = \sum_{\substack{v_j \in V : \wp_{v_j}^- = \wp_{k-1}, \\ d(\wp_{k-1}, v_j) > \frac{r}{2}}} \left( \sum_{\substack{v_i : v_i \prec v_j, \\ d(v_i, v_j) > \frac{r}{2}}} f_{iji} * 2d\left(v_j, \wp_k\right) \right) +$$

$$\sum_{\substack{v_j \in V : \wp_{v_j}^+ = \wp_{k+1}, \\ d(v_j, \wp_{k+1}) > \frac{r}{2}}} \left( \sum_{\substack{v_k : v_k \succ v_j, \\ d(v_j, v_k) > \frac{r}{2}, }} f_{kjk} * 2d\left(v_j, \wp_k\right) \right). \tag{2.30}$$

For notational convenience, let $V_{in}^k = \{v_j : v_j \in V_{in} \cap S_k\}$ represent the set of internal nodes that are within localization segment $S_k$ and let $V_{ex}^{k-1,k} = \{v_j : v_j \in V_{ex}, \beta_{k-1} < l(v_j) < \alpha_k\}$ represent the set of external nodes that are between localization segments $S_{k-1}$ and $S_k$, particularly, $V_{ex}^{0,1} = \{v_j : v_j \in V_{ex}, 0 < l(v_j) < \alpha_1\}$. Note that $V_{in} = \bigcup_{k=1}^{m} V_{in}^k$ and $V_{ex} = \bigcup_{k=1}^{m} V_{ex}^{k-1,k}$.

Given a set of RP locations $\boldsymbol{x}$. Consider an external node $v_j$:

- If $v_j \in V_{ex}^{k-1,k}$ and $l(v_j) - x_{k-1} > \frac{r}{2}$, then the refueling detouring distance associated with any round trip $(v_i, v_j, v_i)$ with $v_i \prec v_j$ is $2f_j^-\left(x_k - l(v_j)\right)$;
- If $v_j \in V_{ex}^{k,k+1}$ and $x_{k+1} - l(v_j) > \frac{r}{2}$, then the refueling detouring distance associated with any round trip $(v_k, v_j, v_k)$ with $v_k \succ v_j$ is $2f_j^+\left(l(v_j) - x_k\right)$.

Then,

$$Z_k^{ex}(\boldsymbol{x}) = 2\left(\sum_{\substack{v_j \in V_{ex}: \mathcal{p}_{v_j}^- = \mathcal{p}_{k-1}, \\ d(\mathcal{p}_{k-1}, v_j) > \frac{r}{2}}} f_j^- d(v_j, \mathcal{p}_k) + \sum_{\substack{v_j \in V_{ex}: \mathcal{p}_{v_j}^+ = \mathcal{p}_{k+1}, \\ d(v_j, \mathcal{p}_{k+1}) > \frac{r}{2}}} f_j^+ d(v_j, \mathcal{p}_k)\right)$$

$$= 2\left(\sum_{\substack{v_j \in V_{ex}^{k-1,k}: \\ l(v_j) > x_{k-1} + \frac{r}{2}}} f_j^-\left(x_k - l(v_j)\right) + \sum_{\substack{v_j \in V_{ex}^{k,k+1}: \\ l(v_j) < x_{k+1} - \frac{r}{2}}} f_j^+\left(l(v_j) - x_k\right)\right) \qquad (2.31)$$

Now let us consider an internal node $v_j$. Again, we shall consider two cases:

(a) $\frac{r}{2} < \hat{d} \leq r$

Given any RP locations $\boldsymbol{x}$. Then:

- for any $v_j \in V_{in}^{k-1}$ with $\mathcal{p}_{v_j}^- = \mathcal{p}_{k-1}$, $d(v_j, \mathcal{p}_{k-1}) \leq \beta_k - \alpha_k = r - \hat{d} < \frac{r}{2}$,
- for any $v_j \in V_{in}^{k+1}$ with $\mathcal{p}_{v_j}^+ = \mathcal{p}_{k+1}$, $d(v_j, \mathcal{p}_{k+1}) \leq \beta_{k+1} - \alpha_{k+1} = r - \hat{d} < \frac{r}{2}$,

i.e., we have $\left\{v_j \in V_{in}^{k-1} \middle| \begin{matrix} \mathcal{p}_{v_j}^+ = \mathcal{p}_k \\ d(\mathcal{p}_{k-1}, v_j) > \frac{r}{2} \end{matrix}\right\} = \emptyset$ and $\left\{v_j \in V_{in}^{k+1} \middle| \begin{matrix} \mathcal{p}_{v_j}^- = \mathcal{p}_k, \\ d(v_j, \mathcal{p}_{k+1}) > \frac{r}{2} \end{matrix}\right\} = \emptyset$.

Furthermore, for any $v_j \in V_{in}^k$:

- if $\mathcal{p}_{v_j}^- = \mathcal{p}_{k-1}$ (i.e., $l(v_j) < x_k$), $d(v_j, \mathcal{p}_{k-1}) \geq \alpha_k - \beta_{k-1} = \hat{d} > \frac{r}{2}$,
- if $\mathcal{p}_{v_j}^+ = \mathcal{p}_{k+1}$, (i.e., $l(v_j) > x_k$), $d(v_j, \mathcal{p}_{k+1}) \geq \alpha_{k+1} - \beta_k = \hat{d} > \frac{r}{2}$,

i.e., we have $\left\{v_j \in V_{in}^k \middle| \begin{matrix} \not{p}_{v_j}^+ = \not{p}_k, \\ d(\not{p}_{k-1}, v_j) > \frac{r}{2} \end{matrix}\right\} = \{v_j \in V_{in}^k : l(v_j) < x_k\}$, and

$\left\{v_j \in V_{in}^k \middle| \begin{matrix} \not{p}_{v_j}^- = \not{p}_k, \\ d(v_j, \not{p}_{k+1}) > \frac{r}{2} \end{matrix}\right\} = \{v_j \in V_{in}^k : l(v_j) > x_k\}$.

Then,

$$Z_k^{in}(\boldsymbol{x}) = 2\left(\sum_{\substack{v_j \in V_{in}: \not{p}_{v_j}^- = \not{p}_{k-1}, \\ d(\not{p}_{k-1}, v_j) > \frac{r}{2}}} f_j^- d(v_j, \not{p}_k) + \sum_{\substack{v_j \in V_{in}: \not{p}_{v_j}^+ = \not{p}_{k+1}, \\ d(v_j, \not{p}_{k+1}) > \frac{r}{2}}} f_j^+ d(v_j, \not{p}_k)\right)$$

$$= 2\left(\sum_{v_j \in V_{in}^k : l(v_j) < x_k} f_j^- \left(x_k - l(v_j)\right) + \sum_{v_j \in V_{in}^k : l(v_j) > x_k} f_j^+ \left(l(v_j) - x_k\right)\right) \qquad (2.32)$$

(b) $0 < \hat{d} \leq \frac{r}{2}$

Unlike case (a), here, let us consider an internal node $v_j \in V_{in}^{k-1}$ with $l(v_j) > x_{k-1} + \frac{r}{2}$, then the electric will need to make a refueling detour to $\not{p}_k$ on its round trip $(v_i, v_j, v_i)$ where $v_i \prec v_j$. In case (b), we have the following:

$\left\{v_j \in V_{in} \middle| \begin{matrix} \not{p}_{v_j}^+ = \not{p}_k, \\ d(\not{p}_{k-1}, v_j) > \frac{r}{2} \end{matrix}\right\} =$

$\left\{v_j \in V_{in}^k : x_{k-1} + \frac{r}{2} < l(v_j) < x_k\right\} \cup \left\{v_j \in V_{in}^{k-1} : l(v_j) > x_{k-1} + \frac{r}{2}\right\}$,

$\left\{v_j \in V_{in} \middle| \begin{matrix} \not{p}_{v_j}^- = \not{p}_k, \\ d(v_j, \not{p}_{k+1}) > \frac{r}{2} \end{matrix}\right\} =$

$\left\{v_j \in V_{in}^k : x_k < l(v_j) < x_{k+1} - \frac{r}{2}\right\} \cup \left\{v_j \in V_{in}^{k+1} : l(v_j) < x_{k+1} - \frac{r}{2}\right\}$.

Then,

$$Z_k^{in}(\boldsymbol{x}) = 2\left(\sum_{\substack{v_j\in V_{in}:\, \wp_{v_j}^{-}=\wp_{k-1},\\ d(\wp_{k-1},v_j)>\frac{r}{2}}} f_j^{-} d(v_j,\wp_k) + \sum_{\substack{v_j\in V_{in}:\, \wp_{v_j}^{+}=\wp_{k+1},\\ d(v_j,\wp_{k+1})>\frac{r}{2}}} f_j^{+} d(v_j,\wp_k)\right)$$

$$= 2\left(\sum_{\substack{v_j\in V_{in}^{k}\\ x_{k-1}+\frac{r}{2}<l(v_j)<x_k}} f_j^{-}\left(x_k - l(v_j)\right) + \sum_{\substack{v_j\in V_{in}^{k-1}\\ l(v_j)>x_{k-1}+\frac{r}{2}}} f_j^{-}\left(x_k - l(v_j)\right)\right.$$

$$\left. + \sum_{\substack{v_j\in V_{in}^{k}\\ x_k<l(v_j)<x_{k+1}-\frac{r}{2}}} f_j^{+}\left(l(v_j) - x_k\right) + \sum_{\substack{v_j\in V_{in}^{k+1}\\ l(v_j)<x_{k+1}-\frac{r}{2}}} f_j^{+}\left(l(v_j) - x_k\right)\right) \quad (2.33)$$

Now, we are able to rewrite the objective function uniformly as

$$Z(\boldsymbol{x}) = \sum_{k=1}^{m}\left(Z_k^{ex}(\boldsymbol{x}) + Z_k^{in}(\boldsymbol{x})\right) \quad (2.34)$$

where $Z_k^{in}(\boldsymbol{x})$ can be computed using (2.32) in case (a) and using (2.33) in case (b).

#### 2.2.3.3. Restricted problem

Now, we shall continue to prove theorem 2.4, which claims that the set of breakpoints $\boldsymbol{\mathcal{B}}$ is an FDS to the line problem. Let us consider a set of restricted problems by requiring each RP to be established within an indivisible sub-segment of its original localization segment. Then an optimal solution of at least one of these restricted problems is optimal to the original problem. Thus, it suffices to show that the set of breakpoints $\boldsymbol{\mathcal{B}}$ is an FDS for each restricted problem. A restricted problem is formulated as following:

Minimize $\quad Z(\boldsymbol{x}) = \sum_{k=1}^{m}\left(Z_k^{ex}(\boldsymbol{x}) + Z_k^{in}(\boldsymbol{x})\right)$ (2.35)

Subject to $\quad x_{k+1} - x_k \le r \qquad 1 \le k < m$ (2.36)

$\alpha_k^{\boldsymbol{h}} \le x_k \le \beta_k^{\boldsymbol{h}} \qquad 1 \le k \le m$ (2.37)

where $\boldsymbol{h} = (h_1, h_2, \cdots, h_m)$ is a particular combination of indivisible sub-segments and $h_k \in \{1, 2, \cdots, n_k - 1\}$ (recall that $n_k$ is defined as the number of breakpoints in $S_k$), and where $\left[\alpha_k^{\boldsymbol{h}}, \beta_k^{\boldsymbol{h}}\right]$ is the ${h_k}^{th}$ indivisible sub-segment between two consecutive breakpoints $\alpha_k^{\boldsymbol{h}}$ (the ${h_k}^{th}$ breakpoint in $S_k$) and $\beta_k^{\boldsymbol{h}}$ (the $(h_k + 1)^{th}$ breakpoint in $S_k$).

Let $\boldsymbol{C}$ represent the solution space. It is easy to see that $\boldsymbol{C}$ is a convex set. Hereon, we proceed with proving that the objective function $Z(\boldsymbol{x})$ is concave on $\boldsymbol{C}$. Let $\boldsymbol{x}, \boldsymbol{x}' \in \boldsymbol{C}$ and $0 < \theta < 1$, that is, we wish to prove $Z(\boldsymbol{x}'') = Z\big((1-\theta)\boldsymbol{x} + \theta\boldsymbol{x}'\big) \geq (1-\theta)Z(\boldsymbol{x}) + \theta Z(\boldsymbol{x}')$. Not surprisingly, we would discuss the following two cases: (a) $\frac{r}{2} < \hat{d} \leq r$ and (b) $0 < \hat{d} \leq \frac{r}{2}$.

(a) $\frac{r}{2} < \hat{d} \leq r$

(a.1) Let us first show that $Z_k^{ex}(\boldsymbol{x})$ is concave on $\boldsymbol{C}$. Recall equation (2.31):

$$Z_k^{ex}(\boldsymbol{x}) = 2\left(\sum_{\substack{v_j \in V_{ex}^{k-1,k}: \\ l(v_j) > x_{k-1} + \frac{r}{2}}} f_j^-\left(x_k - l(v_j)\right) + \sum_{\substack{v_j \in V_{ex}^{k,k+1}: \\ l(v_j) < x_{k+1} - \frac{r}{2}}} f_j^+\left(l(v_j) - x_k\right)\right).$$

Consider any external node $v_j \in V_{ex}^{k-1,k}$. Then ${v_j}^-\left(\frac{r}{2}\right) \notin \left(\alpha_{k-1}^{\boldsymbol{h}}, \beta_{k-1}^{\boldsymbol{h}}\right)$, since $\left[\alpha_{k-1}^{\boldsymbol{h}}, \beta_{k-1}^{\boldsymbol{h}}\right]$ is indivisible. That is, we have either ${v_j}^-\left(\frac{r}{2}\right) \leq \alpha_k^{\boldsymbol{h}}$ or ${v_j}^-\left(\frac{r}{2}\right) \geq \beta_k^{\boldsymbol{h}}$. Then for any $x_{k-1} \in \left[\alpha_{k-1}^{\boldsymbol{h}}, \beta_{k-1}^{\boldsymbol{h}}\right]$, we have

$$\left\{v_j : v_j \in V_{ex}^{k-1,k}, l(v_j) > x_{k-1} + \frac{r}{2}\right\}$$

$$= \left\{v_j : v_j \in V_{ex}^{k-1,k}, {v_j}^-\left(\frac{r}{2}\right) > x_{k-1}\right\}$$

$$= \begin{cases} \left\{v_j : v_j \in V_{ex}^{k-1,k}, x_{k-1} < {v_j}^-\left(\frac{r}{2}\right) < \beta_{k-1}^{\boldsymbol{h}}\right\} \cup \left\{v_j : v_j \in V_{ex}^{k-1,k}, {v_j}^-\left(\frac{r}{2}\right) \geq \beta_{k-1}^{\boldsymbol{h}}\right\}, & if\ x_{k-1} < \beta_{k-1}^{\boldsymbol{h}} \\ \left\{v_j : v_j \in V_{ex}^{k-1,k}, {v_j}^-\left(\frac{r}{2}\right) > \beta_{k-1}^{\boldsymbol{h}}\right\}, & if\ x_{k-1} = \beta_{k-1}^{\boldsymbol{h}} \end{cases}$$

$$= \begin{cases} \left\{v_j : v_j \in V_{ex}^{k-1,k}, {v_j}^-\left(\frac{r}{2}\right) \geq \beta_{k-1}^{\boldsymbol{h}}\right\}, & if\ x_{k-1} < \beta_{k-1}^{\boldsymbol{h}} \\ \left\{v_j : v_j \in V_{ex}^{k-1,k}, {v_j}^-\left(\frac{r}{2}\right) > \beta_{k-1}^{\boldsymbol{h}}\right\}, & if\ x_{k-1} = \beta_{k-1}^{\boldsymbol{h}} \end{cases} \left(by\ {v_j}^-\left(\frac{r}{2}\right) \notin \left(\alpha_{k-1}^{\boldsymbol{h}}, \beta_{k-1}^{\boldsymbol{h}}\right)\right) \quad (2.38)$$

which implies that if there exists a node $v^* \in V_{ex}^{k-1,k}$ such that ${v^*}^-\left(\frac{r}{2}\right) = \beta_{k-1}^{\boldsymbol{h}}$,

$$\left\{v_j : v_j \in V_{ex}^{k-1,k}, {v_j}^-\left(\frac{r}{2}\right) \geq \beta_{k-1}^{\boldsymbol{h}}\right\} \backslash \{v^*\} = \left\{v_j : v_j \in V_{ex}^{k-1,k}, {v_j}^-\left(\frac{r}{2}\right) > \beta_{k-1}^{\boldsymbol{h}}\right\}.$$

Without loss of generality we assume that $x_{k-1} < x'_{k-1}$. Then,

$$\sum_{\substack{v_j \in V_{ex}^{k-1,k} \\ l(v_j) > x''_{k-1} + \frac{r}{2}}} f_j^-\left(x''_k - l(v_j)\right)$$

$$= \sum_{\substack{v_j \in V_{ex}^{k-1,k} \\ l(v_j) > x''_{k-1} + \frac{r}{2}}} f_j^-\left((1-\theta)x_k + \theta x'_k) - l(v_j)\right)$$

$$= \sum_{\substack{v_j \in V_{ex}^{k-1,k} \\ l(v_j) > x''_{k-1} + \frac{r}{2}}} (1-\theta) f_j^-\left(x_k - l(v_j)\right) + \theta f_j^-\left(x'_k - l(v_j)\right)$$

$$= (1-\theta) \sum_{\substack{v_j \in V_{ex}^{k-1,k} \\ l(v_j) > x''_{k-1} + \frac{r}{2}}} f_j^-\left(x_k - l(v_j)\right) + \theta \sum_{\substack{v_j \in V_{ex}^{k-1,k} \\ l(v_j) > x''_{k-1} + \frac{r}{2}}} f_j^-\left(x'_k - l(v_j)\right)$$

$$\geq (1-\theta) \sum_{\substack{v_j \in V_{ex}^{k-1,k} \\ l(v_j) > x_{k-1} + \frac{r}{2}}} f_j^-\left(x_k - l(v_j)\right) + \theta \sum_{\substack{v_j \in V_{ex}^{k-1,k} \\ l(v_j) > x'_{k-1} + \frac{r}{2}}} f_j^-\left(x'_k - l(v_j)\right) \qquad (by\ (2.38))$$

where strict inequality is achieved when $x'_{k-1} = \beta_{k-1}^{\boldsymbol{h}}$ and there exists a node $v^* \in V_{ex}^{k-1,k}$ such that ${v^*}^-\left(\frac{r}{2}\right) = \beta_{k-1}^{\boldsymbol{h}}$, since $\left\{v_j : v_j \in V_{ex}^{k-1,k}, l(v_j) > x''_{k-1} + \frac{r}{2}\right\} \backslash \{v^*\} = \left\{v_j : v_j \in V_{ex}^{k-1,k}, l(v_j) > x''_{k-1} + \frac{r}{2}\right\}$. Hence, the first component $\sum_{\substack{v_j \in V_{ex}^{k-1,k}: \\ l(v_j) > x_{k-1} + \frac{r}{2}}} f_j^-\left(x_k - l(v_j)\right)$ of $Z_k^{ex}(\boldsymbol{x})$ is concave on $\boldsymbol{C}$. Likewise, we are able to show that the second component $\sum_{\substack{v_j \in V_{ex}^{k,k+1}: \\ l(v_j) < x_{k+1} - \frac{r}{2}}} f_j^+\left(l(v_j) - x_k\right)$ of $Z_k^{ex}(\boldsymbol{x})$ is concave on $\boldsymbol{C}$.

(a.2) Then, let us show that $Z_k^{in}(\boldsymbol{x})$ is concave on $\boldsymbol{C}$ as well. Recall equation (2.32):

$$Z_k^{in}(\boldsymbol{x}) = 2\left(\sum_{v_j \in V_{in}^k:\, l(v_j) < x_k} f_j^-\left(x_k - l(v_j)\right) + \sum_{v_j \in V_{in}^k:\, l(v_j) > x_k} f_j^+\left(l(v_j) - x_k\right)\right).$$

Note that we can rewrite $Z_k^{in}(\boldsymbol{x})$ as

$$2\left(\sum_{v_j \in V_{in}^k:\, l(v_j) \le x_k} f_j^-\left(x_k - l(v_j)\right) + \sum_{v_j \in V_{in}^k:\, l(v_j) \ge x_k} f_j^+\left(l(v_j) - x_k\right)\right),$$

Consider any internal node $v_j \in V_{in}^k$. Then $v_j \notin \left(\alpha_k^{\boldsymbol{h}}, \beta_k^{\boldsymbol{h}}\right)$, since $\left[\alpha_k^{\boldsymbol{h}}, \beta_k^{\boldsymbol{h}}\right]$ is indivisible. That is, we have either $v_j \le \alpha_k^{\boldsymbol{h}}$ or $v_j \ge \beta_k^{\boldsymbol{h}}$. Then for any $x_k \in \left[\alpha_k^{\boldsymbol{h}}, \beta_k^{\boldsymbol{h}}\right]$, we have

$$\left\{v_j : v_j \in V_{in}^k, l(v_j) \le x_k\right\} = \begin{cases} \left\{v_j : v_j \in V_{in}^k, l(v_j) \le \alpha_k^{\boldsymbol{h}}\right\},\ if\ x_k < \beta_k^{\boldsymbol{h}} \\ \left\{v_j : v_j \in V_{in}^k, l(v_j) \le \alpha_k^{\boldsymbol{h}}\right\} \cup \left\{\beta_k^{\boldsymbol{h}}\right\},\ if\ x_k = \beta_k^{\boldsymbol{h}}\ and\ if\ \beta_k^{\boldsymbol{h}} \in V_{in}^k \end{cases},$$

and

$$\left\{v_j : v_j \in V_{in}^k, l(v_j) \ge x_k\right\} = \begin{cases} \left\{v_j : v_j \in V_{in}^k, l(v_j) \ge \beta_k^{\boldsymbol{h}}\right\} \cup \left\{\alpha_k^{\boldsymbol{h}}\right\},\ if\ x_k = \alpha_k^{\boldsymbol{h}}\ and\ if\ \alpha_k^{\boldsymbol{h}} \in V_{in}^k \\ \left\{v_j : v_j \in V_{in}^k, l(v_j) \ge \beta_k^{\boldsymbol{h}}\right\},\ if\ x_k > \alpha_k^{\boldsymbol{h}} \end{cases}.$$

Then,

$$\begin{aligned} &\sum_{v_j \in V_{in}^k:\, l(v_j) \le x_k''} f_j^-\left(x_k'' - l(v_j)\right) \\ &= (1-\theta)\sum\nolimits_{v_j \in V_{in}^k:\, l(v_j) \le x_k''} f_j^-\left(x_k - l(v_j)\right) + \theta \sum\nolimits_{v_j \in V_{in}^k:\, l(v_j) \le x_k''} f_j^-\left(x_k' - l(v_j)\right) \\ &= (1-\theta)\sum\nolimits_{v_j \in V_{in}^k:\, l(v_j) \le x_k} f_j^-\left(x_k - l(v_j)\right) + \theta \sum\nolimits_{v_j \in V_{in}^k:\, l(v_j) \le x_k'} f_j^-\left(x_k' - l(v_j)\right), \end{aligned}$$

it is worth to mention that when $x_k' = \beta_k^{\boldsymbol{h}}$ the equality between $\sum_{v_j \in V_{in}^k:\, l(v_j) \le x_k'} f_j^-\left(x_k' - l(v_j)\right)$ and $\sum_{v_j \in V_{in}^k:\, l(v_j) \le x_k''} f_j^-\left(x_k' - l(v_j)\right)$ still holds, since $\left(x_k' - \beta_k^{\boldsymbol{h}}\right)\big|_{x_k' = \beta_k^{\boldsymbol{h}}} = 0$. Hence, the first component of $Z_k^{in}(\boldsymbol{x})$ is concave on $\boldsymbol{C}$. Follow the same fashion, we can show that the second component of $Z_k^{in}(\boldsymbol{x})$ is also concave on $\boldsymbol{C}$.

Therefore, we shall claim that for case (a) the objective function $Z(x)$ is concave on $\boldsymbol{C}$.

(b) $0 < \hat{d} \le \frac{r}{2}$

In case (b), follow the same fashion in (a), we are able to show that the following four components are concave on $\boldsymbol{C}$:

- $\sum_{\substack{v_j \in V_{ex}^{k-1,k}: \\ l(v_j) > x_{k-1} + \frac{r}{2}}} f_j^-\left(x_k - l(v_j)\right) + \sum_{\substack{v_j \in V_{in}^{k-1} \\ l(v_j) > x_{k-1} + \frac{r}{2}}} f_j^-\left(x_k - l(v_j)\right)$,
- $\sum_{\substack{v_j \in V_{ex}^{k,k+1}: \\ l(v_j) < x_{k+1} - \frac{r}{2}}} f_j^+\left(l(v_j) - x_k\right) + \sum_{\substack{v_j \in V_{in}^{k+1} \\ l(v_j) < x_{k+1} - \frac{r}{2}}} f_j^+\left(l(v_j) - x_k\right)$,
- $\sum_{\substack{v_j \in V_{in}^{k} \\ x_{k-1} + \frac{r}{2} < l(v_j) < x_k}} f_j^-\left(x_k - l(v_j)\right)$,
- $\sum_{\substack{v_j \in V_{in}^{k} \\ x_k < l(v_j) < x_{k+1} - \frac{r}{2}}} f_j^+\left(l(v_j) - x_k\right)$.

Therefore, at least one of the extreme points will be the optimal solution for each restricted problem. Let $\mathcal{E}(\boldsymbol{C})$ denote the set of extreme points of the solution space $\boldsymbol{C}$. Then we shall prove that $\mathcal{E}(\boldsymbol{C}) \subseteq \{(b_1, b_2, \cdots, b_m): b_k \in \boldsymbol{\mathcal{B}}, k = 1, \dots, m\}$. The extreme points of a polyhedron are defined algebraically as: Let $\bar{\boldsymbol{x}} \in P = \{x \in \mathbb{R}^n \mid \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}\}$, where $rank(\boldsymbol{A}) = n$ and $\boldsymbol{b} \in \mathbb{R}^m$. Further, let $\bar{\boldsymbol{A}}\bar{\boldsymbol{x}} = \bar{\boldsymbol{b}}$ be the equality subsystem of $\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}$. Then $\bar{\boldsymbol{x}}$ is an extreme point of $P$ if and only if $rank(\bar{\boldsymbol{A}}) = n$. That the equality subsystem has rank $n$ basically means that there should be at least $n$ linearly independent half-spaces going through the point $\bar{\boldsymbol{x}}$.

Recall that $\boldsymbol{C} = \begin{Bmatrix} \boldsymbol{x}: & x_{k+1} - x_k \leq r, 1 \leq k < m \\ & \alpha_k^{\boldsymbol{h}} \leq x_k \leq \beta_k^{\boldsymbol{h}}, for\ 1 \leq k \leq m \end{Bmatrix}$. Then for any $\bar{\boldsymbol{x}} \in \mathcal{E}(\boldsymbol{C})$, there should be at least $m$ linear independent half-spaces going through it. Hence, at least one half-space in $\{x_k \geq \alpha_k^{\boldsymbol{h}}, for\ 1 \leq k \leq m\} \cup \{x_k \leq \beta_k^{\boldsymbol{h}}, for\ 1 \leq k \leq m\}$ goes through $\bar{\boldsymbol{x}}$. Without loss of generality, suppose that the half-space $x_k \leq \beta_k^{\boldsymbol{h}}$ together with other $m - 1$ half-spaces in $\{x_{k+1} - x_k \leq r, for\ 1 \leq k < m\}$ go through $\bar{\boldsymbol{x}}$. Hence,

$\overline{\boldsymbol{x}} = \left(\beta_k^{\boldsymbol{h}} - (k-1)r, \cdots, \beta_k^{\boldsymbol{h}} - r, \beta_k^{\boldsymbol{h}}, \beta_k^{\boldsymbol{h}} + r, \cdots, \beta_k^{\boldsymbol{h}} + (m-k)r\right)$. Clearly, $\overline{\boldsymbol{x}} \in$ $\{(b_1, b_2, \cdots, b_m)\colon b_k \in \boldsymbol{\mathcal{B}}, k = 1, \ldots, m\}$.

Thus, theorem 2.4 has been proved.

### 2.2.4. Solution Method

#### 2.2.4.1. Network construction

In this section, we formulate our problem as a shortest path problem on an acyclic network. The network has $m$ layers of nodes: It has one layer corresponding to each RP $\mathcal{p}_k \in \{\mathcal{p}_1, \mathcal{p}_2, \cdots, \mathcal{p}_m\}$. The layer $k$ has $n_k$ nodes, $\{n_1^k, n_2^k, \cdots, n_{n_k}^k\}$, where $n_i^k$ denotes the $i^{th}$ breakpoint (in the left to right order) in localization segment $S_k$, and signifies that $\mathcal{p}_k$ is established at that breakpoint. Connect nodes $n_i^k$ and $n_j^{k+1}$ if the distance between the two corresponding breakpoints is not more than $r$. Denote $w\left(n_i^k, n_j^{k+1}\right)$ the cost of edge $\left(n_i^k, n_j^{k+1}\right)$. Recall equation (2.30), the total refueling detouring distance associated with RP $\mathcal{p}_k$ is $Z_k(\boldsymbol{x})$:

$$Z_k(\boldsymbol{x}) = 2\left(\sum_{\substack{v_j \in V:\, \mathcal{p}_{v_j}^- = \mathcal{p}_{k-1}, \\ d(\mathcal{p}_{k-1}, v_j) > \frac{r}{2}}} \left(f_j^- d\left(v_j, \mathcal{p}_k\right)\right) + \sum_{\substack{v_j \in V:\, \mathcal{p}_{v_j}^+ = \mathcal{p}_{k+1}, \\ d(v_j, \mathcal{p}_{k+1}) > \frac{r}{2}}} \left(f_j^- d\left(v_j, \mathcal{p}_k\right)\right)\right),$$

Here the first sum depends on the positions of $\mathcal{p}_{k-1}$ and $\mathcal{p}_k$ and the second one depends on the positions of $\mathcal{p}_k$ and $\mathcal{p}_{k+1}$. Then, for notational convenience, we let

$$D_k^-(x_{k-1}, x_k) = \sum_{\substack{v_j:\, \mathcal{p}_{v_j}^- = \mathcal{p}_{k-1}, \\ d(\mathcal{p}_{k-1}, v_j) > \frac{r}{2}}} f_j^- d\left(v_j, \mathcal{p}_k\right) \tag{2.39}$$

$$D_k^+(x_k, x_{k+1}) = \sum_{\substack{v_j:\, \mathcal{p}_{v_j}^+ = \mathcal{p}_{k+1}, \\ d(\mathcal{p}_{k+1}, v_j) > \frac{r}{2}}} f_j^+ d\left(v_j, \mathcal{p}_k\right) \tag{2.40}$$

Particularly, $D_1^-(x_1) = 0$ and $D_m^+(x_m) = 0$. In case (a) $\frac{r}{2} < \hat{d} \leq r$, we have

$$D_k^-(x_{k-1}, x_k) = \sum_{\substack{v_j \in V_{ex}^{k-1,k} \\ l(v_j) > x_{k-1} + \frac{r}{2}}} f_j^- \left(x_k - l(v_j)\right) + \sum_{\substack{v_j \in V_{in}^k \\ l(v_j) < x_k}} f_j^- \left(x_k - l(v_j)\right),$$

$$D_k^+(x_k, x_{k+1}) = \sum_{\substack{v_j \in V_{in}^k \\ l(v_j) > x_k}} f_j^+ \left(l(v_j) - x_k\right) + \sum_{\substack{v_j \in V_{ex}^{k,k+1} \\ l(v_j) < x_{k+1} - \frac{r}{2}}} f_j^+ \left(l(v_j) - x_k\right).$$

In case (b) $0 < \hat{d} \leq \frac{r}{2}$, we have

$$D_k^-(x_{k-1}, x_k) = \sum_{\substack{v_j \in V_{in}^{k-1} \\ l(v_j) > x_{k-1} + \frac{r}{2}}} f_j^- \left(x_k - l(v_j)\right) + \sum_{\substack{v_j \in V_{ex}^{k-1,k} \\ l(v_j) > x_{k-1} + \frac{r}{2}}} f_j^- \left(x_k - l(v_j)\right)$$

$$+ \sum_{\substack{v_j \in V_{in}^k \\ x_{k-1} + \frac{r}{2} < l(v_j) < x_k}} f_j^- \left(x_k - l(v_j)\right),$$

$$D_k^+(x_k, x_{k+1}) = \sum_{\substack{v_j \in V_{in}^k \\ x_k < l(v_j) < x_{k+1} - \frac{r}{2}}} f_j^+ \left(l(v_j) - x_k\right) + \sum_{\substack{v_j \in V_{ex}^{k,k+1} \\ l(v_j) < x_{k+1} - \frac{r}{2}}} f_j^+ \left(l(v_j) - x_k\right)$$

$$+ \sum_{\substack{v_j \in V_{in}^{k+1} \\ l(v_j) < x_{k+1} - \frac{r}{2}}} f_j^+ \left(l(v_j) - x_k\right).$$

Let $w\left(n_i^k, n_j^{k+1}\right) = D_k^+\left(n_i^k, n_j^{k+1}\right) + D_{k+1}^-\left(n_i^k, n_j^{k+1}\right)$.

#### 2.2.4.2. Correctness

Each path of the network corresponds to a feasible solution of our original problem, as each path contains one node from every layer, and each node represents a breakpoint which is a candidate location site for a refueling station, and two nodes can be connected only if the distance between the two corresponding breakpoints is not more than $r$. Let $P = \left(n_{r_1}^1, n_{r_2}^2, \cdots, n_{r_m}^m\right)$ be a path of the network, where $r_k$ denote the rank of the node in its layer. Denote $W(P)$ the cost of path $P$, then

$$
\begin{aligned}
W(P) &= \sum_{k=1}^{m-1} w\left(n_{r_k}^k, n_{r_{k+1}}^{k+1}\right) \\
&= \sum_{k=1}^{m-1} \left(D_k^+\left(n_{r_k}^k, n_{r_{k+1}}^{k+1}\right) + D_{k+1}^-\left(n_{r_k}^k, n_{r_{k+1}}^{k+1}\right)\right) \\
&= D_1^+\left(n_{r_1}^1\right) + \sum_{k=2}^{m-1} \left(D_k^-\left(n_{r_{k-1}}^{k-1}, n_{r_k}^k\right) + D_k^+\left(n_{r_k}^k, n_{r_{k+1}}^{k+1}\right)\right) + D_m^-\left(n_{r_m}^m\right) \\
&= \sum_{k=1}^{m} Z_k(\boldsymbol{x}).
\end{aligned}
$$

Therefore, the path cost is equal to the total refueling detouring distance associated with such an infrastructure layout. Conversely, a feasible solution to our problem defines a path from layer 1 to layer $m$ with a cost equal to the total detour distance.

This correspondence implies that we are able find the optimal solution to our original problem by finding the shortest path of the constructed network.

**Remark** Recall theorem 2.3 which claims that $\wp_1$ can always be located at $\beta_1$, and $\wp_m$ can always be located at $\alpha_m$. Therefore, we can only keep the breakpoint $\beta_1$ for localization segment $S_1$ and $\alpha_m$ for segment $S_m$.

EXAMPLE 2.5 Let us consider the same example for illustrating how to construct the network. There are five layers of nodes, each layer corresponding to a station $\wp_k$, $k \in \{1, \cdots, 5\}$. Layer 1 contains only one node $n_1^1$, i.e., the breakpoint $b_1 (= 3.5)$, and layer 5 contains one node $n_1^5 (= a_5 = 28.5)$ as well. Layer 2 contains 5 nodes: $n_1^2 = 7.5$, $n_2^2 = 9$, $n_3^2 = 9.5$, $n_4^2 = 10$, and $n_5^2 = 10.5$. Layer 3 contains 5 nodes: $n_1^3 = 14.5$, $n_2^3 = 16$, $n_3^3 = 16.5$, $n_4^3 = 17$, and $n_5^3 = 17.5$. Layer 4 contains 5 nodes as well: $n_1^4 = 28.5$, $n_2^4 = 30$, $n_3^4 = 30.5$, $n_4^4 = 31$, and $n_5^4 = 31.5$.

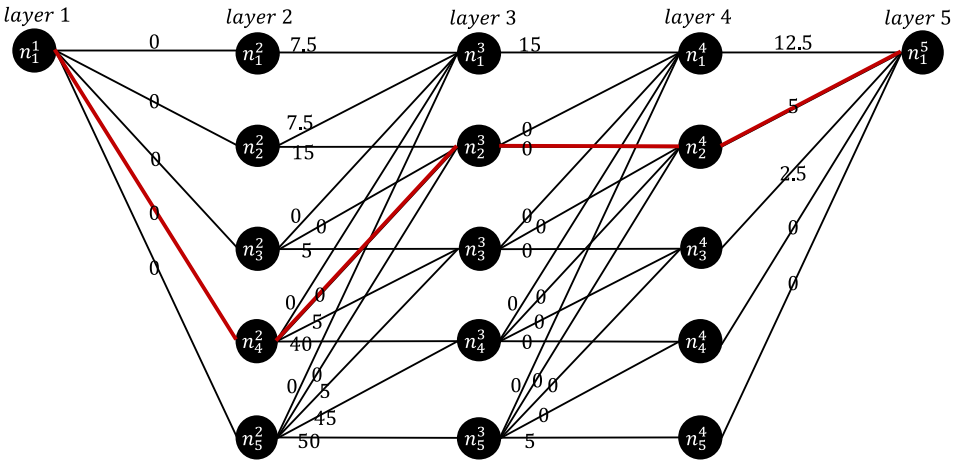



**Figure 2.9** The network constructed for example 2.5

To illustrate how to derive edge costs, we take $w(n_4^2, n_3^3)$ for example. Note that $\hat{d} = 4 > \frac{r}{2}$ in this example.

$$w(n_4^2, n_3^3) = D_2^+(n_4^2, n_3^3) + D_3^-(n_4^2, n_3^3)$$

$$= \sum_{\substack{v_j \in V_{in}^2 \\ l(v_j) > 10}} f_j^+\big(l(v_j) - 10\big) + \sum_{\substack{v_j \in V_{ex}^{2,3} \\ l(v_j) < 16.5 - 3.5}} f_j^+\big(l(v_j) - 10\big)$$

$$+ \sum_{\substack{v_j \in V_{ex}^{2,3} \\ l(v_j) > 10 + 3.5}} f_j^-\Big(16.5 - l\big(v_j\big)\Big) + \sum_{\substack{v_j \in V_{in}^3 \\ l(v_j) < 16.5}} f_j^-\Big(16.5 - l\big(v_j\big)\Big),$$

where $V_{in}^2 = \emptyset$, $V_{ex}^{2,3} = \{v_1\}$, and $V_{in}^3 = \{v_2\}$, and $l(v_1) = 13$, $l(v_2) = 16$. Therefore,

$\sum_{\substack{v_j \in V_{in}^2 \\ l(v_j) > 10}} f_j^+\big(l(v_j) - 10\big) = 0$, $\sum_{\substack{v_j \in V_{ex}^{2,3} \\ l(v_j) < 13}} f_j^+\big(l(v_j) - 10\big) = 0$, $\sum_{\substack{v_j \in V_{ex}^{2,3} \\ l(v_j) > 13.5}} f_j^-\Big(16.5 - l\big(v_j\big)\Big) =$ 0, and $\sum_{\substack{v_j \in V_{in}^3 \\ l(v_j) < 16.5}} f_j^-\Big(16.5 - l\big(v_j\big)\Big) = f_{v_2}^- \cdot (0.5)$. Therefore, $w(n_4^2, n_3^3) = f_{v_2}^- \cdot (0.5)$.

The shortest path of the network is $n_1^1 - n_4^2 - n_2^3 - n_2^4 - n_1^5$, with length = 5.

Same result as using math programming: $x_1 = 3.5, x_2 = 10, x_3 = 16, x_4 = 23, x_5 = 28.5$, total detouring = 5.

**Remark** The network we constructed can be viewed as multistage graph.

A multistage graph is a directed graph in which the nodes can be divided into a set of stages such that all edges are from a stage to next stage only. There are multiple strategies we can apply to find the shortest path. For example, the Dijkstra's algorithm of single source shortest paths, but which does not use the special feature that a multistage graph has. The best strategy is using dynamic programming, and the time complexity is $\mathcal{O}(n^2)$, where $n$ is the number of nodes in the graph.

***Theorem 2.5*** The line problem can be solved in $\mathcal{O}(mn^3 + m^2n^2)$.

*Proof of Theorem 2.5* Recall from Theorem 2.2, the minimum number of RPs that are necessary and sufficient to serve all round-trip triples is $m = \left\lceil \frac{l(v_n)}{r} \right\rceil$. Let $T_1$ represents the time required to construct the network and let $T_2$ represent the time required to find the shortest path in the constructed network.

(1) $T_1 = \mathcal{O}(mn^3)$. By the nature of the line network problem and by the construction of the multistage network note that the number of nodes in each node layer in our constructed multistage network is at most $n + 1$. Also, note that we only need to keep one breakpoint in the first layer and last layer. By the calculation of edge weights, we know that the time required to find the weight of an edge is $\mathcal{O}(n + 1)$. Thus, $T_1 = \mathcal{O}\big((m - 3)(n + 1)^3 + 2(n + 1)^2\big) = \mathcal{O}(mn^3)$.

(2) $T_2 = \mathcal{O}(m^2n^2)$. Note that the total number of nodes in our multistage network is at most $(m - 2) * (n + 1) + 2$. Thus, finding the shortest path in the constructed multistage network will take $\mathcal{O}\left(\big((m - 2) * (n + 1) + 2\big)^2\right) = \mathcal{O}(m^2n^2)$ time.

The overall run time T of the line problem is $T_1 + T_2 = \mathcal{O}(mn^3 + m^2n^2)$. ■

## 2.3. Conclusion

In this chapter, we studied the continuous location problem related to locating RPs on line networks, where finding the minimum number of RPs needed to refuel all O-D flows is considered as the first objective. Given this minimum number, our goal is to locate this number of RPs to minimize weighted sum of the travelling distance for all O-D flows. The one-way scenario is rather simple. For the round-trip scenario, an integer program with linear constraints and quartic objective function is formulated, and the problem can be solved using OPTI toolbox in Matlab. We have also identified a finite dominating set to the problem, and based on the existence of finite dominating set, the problem is formulated as a shortest path problem.

# CHAPTER 3

# THE COMB TREE PROBLEM

## 3.1. Overview

In chapter 2 we discussed the continuous version of detouring-flow location problem on a real line. In this chapter we address the location problem on a comb tree. Again, this problem is to (a) first determine the minimum number of RPs that are necessary and sufficient to refuel all O-D traffic flow, and (b) then determine the optimal locations for RPs that minimize the total travelling distance.

Let us begin our discussion by way of a simple example. Consider the small tree in Figure 3.1. By a "small tree" we mean that a single RP is sufficient to serve all O-D transportation needs. We are ignoring the possibility of queue formation for battery swapping/recharging services. In Figure 3.1, there are 6 ordered O-D pairs in total: $(A,B)$, $(A,C)$, $(B,A)$, $(B,C)$, $(C,A)$ and $(C,B)$. We wish to find the optimal location that minimizes the total traveling distance by using a single RP.

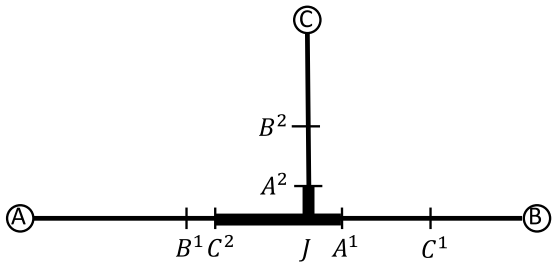



**Figure 3.1** A simple example with a small tree

Points $A^1$, $A^2$, $B^1$, $B^2$, $C^1$ and $C^2$ are such that the distances $AA^1$, $AA^2$, $BB^1$, $BB^2$, $CC^1$ and $CC^2$ are all equal to the range limit $r$. The bold segments, which constitute a subtree, represent the intersection of all paths between these points. Note that if we

locate the RP beyond this subtree, then one RP would be insufficient to serval all O-D pairs. For example, suppose that we've located the RP at point $C^1$, then trip $(A, B)$ and trip $(A, C)$ cannot be satisfied. Therefore, by using a single RP, it must be located on this subtree, and all O-D pairs can be served then, with or without the need of detouring. To minimize the total traveling distance, junction node $J$ would be the only optimal location such that all trips can be served without detouring, since $J$ lies on any shortest path between a given pair of O-D nodes.

We now formally consider a general comb tree.

Let $G = (V, A)$ be an undirected comb tree with node set $V$ and arc set $A$. See Figure 3.2 for illustration. The node set $V$ can be further partitioned into two subsets: a subset of leaf nodes (nodes with degree 1), and a subset of junction nodes. End node $j$ is denoted $v_j$, $j = 0, 1, \cdots, n$, and each end node serves as an origin and/or a destination of a trip by the electric vehicle. Junction node $j$ is denoted $J_j$, $j = 1, \cdots, n-1$. Hence, the cardinality of set $V$ is $|V| = 2n$. The arc set $A = \{(v_0, J_1), (v_n, J_{n-1})\} \cup \{(v_k, J_k), k = 1, \cdots, n-1\} \cup \{(J_{k-1}, J_k), k = 2, \cdots, n-1\}$. An arc connecting nodes $v_k$ and $J_k$ is denoted $a_k$, $k = 1, \cdots, n-1$, and an arc connecting nodes $J_{k-1}$ and $J_k$ is denoted $a_{n+k-1}$, $k = 2, \cdots, n-1$. Additionally, let $a_0$ represent the arc connecting $v_0$ and $J_1$, and $a_n$ represent the arc connecting $v_n$ and $J_{n-1}$. Hence, the cardinality of set $A$ is $|A| = 2n - 1$. We say that $a_k$ is a comb tooth of $G$ if $a_k$ connects a leaf node and a junction node. Associate with each arc $a_k \in A$ is a nonnegative weight $b_k$ representing its length, and the length $b(x, y)$ of the portion of arc between points $x$ and $y$ on $a_k$ is defined to be $b(x, y) = \left|b\left(e_k^1, x\right) - b\left(e_k^1, y\right)\right|$, where $e_k^1$ and $e_k^2$ are the two endpoints of arc $a_k$, one is a leaf node and the other is a junction node, precisely, let $e_k^1$ represent the leaf node. The length function $b$ yields a distance

function $d$ for the comb, where $d(x, y)$ is defined to be the shortest path length from $x$ to $y$ for any two points $x$, $y$ on $G$. However, there is exactly one path between them since $G$ is a tree and let $P(x, y)$ denote this unique path. We say that $P(v_0, v_n)$ is the comb span of $G$.

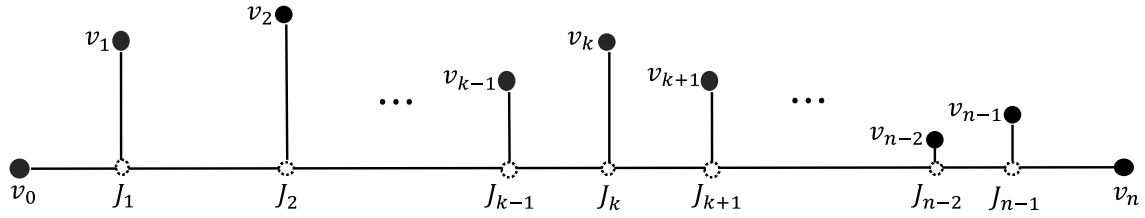



**Figure 3.2** A comb tree

Further, we use the cartesian coordinate system to uniquely determine the position of the points on $G$, without loss of generality we assume that each comb tooth is perpendicular to the comb span.

Then the line goes through comb span is chosen as the horizontal axis, and the line that is perpendicular to the comb span and goes through node $v_0$ is chosen as the vertical axis. Let $\big(l_1(x), l_2(x)\big)$ denote the coordinates of $x$ on $G$, where $l_1(x)$ and $l_2(x)$ are taken to be the distances to the axes. Specifically, $\big(l_1(v_0), l_2(v_0)\big) = (0, 0)$. For points $x$, $y$ on $G$, we say that $x$ is on the left-hand side of $y$ on $G$ if $l_1(x) < l_1(y)$, and that $x$ is on the right-hand side of $y$ if $l_1(x) > l_1(y)$.

We denote a pair $\big(v_i, v_j\big)$ as the one-way transportation need for flow from nodes $v_i$ to $v_j$, for all $v_i, v_j \in V$. The average traffic flow volume on $P\big(v_i, v_j\big)$ is denoted as $f\big(v_i, v_j\big)$. Again, in this problem, an electric vehicle is assumed to depart from its origin with a fully charged battery and needs to reach its destination. If a set of RP locations, say $\boldsymbol{\mathcal{P}}$, is given and has been added to $G$, then we say that trip $\big(v_i, v_j\big)$ can be served if there exists a refueling walk in $G$ starting at $v_i$ and ending at $v_j$ which

has no segment without refueling with a length greater than the range limit $r$, that is, any path contained in this refueling walk starting and ending at nodes in $\{v_i, v_j\} \cup \boldsymbol{\mathcal{P}}$ has length at most $r$. Here, the term "walk" is used as opposed to "path" since a detour to refuel the vehicle might include repeated nodes or arcs. In this problem, as many refueling stops can be taken by the vehicle as necessary. Consider the comb tree with three RPs in Figure 3.3, a refueling walk for one-way trip $(v_2, v_4)$ is $v_2 \rightarrow J_2 \rightarrow \mathcal{p}_1 \rightarrow J_2 \rightarrow J_3 \rightarrow \mathcal{p}_2 \rightarrow J_3 \rightarrow J_4 \rightarrow \mathcal{p}_3 \rightarrow J_4 \rightarrow v_4$, if $d(v_2, \mathcal{p}_1) \leq r$, $d(\mathcal{p}_1, \mathcal{p}_2) \leq r$, $d(\mathcal{p}_2, \mathcal{p}_3) \leq r$, and $d(\mathcal{p}_3, v_4) \leq r$. While, the shortest path from $v_2$ to $v_4$ is $p(v_2, v_4) = v_2 \rightarrow J_2 \rightarrow J_3 \rightarrow J_4 \rightarrow v_4$.

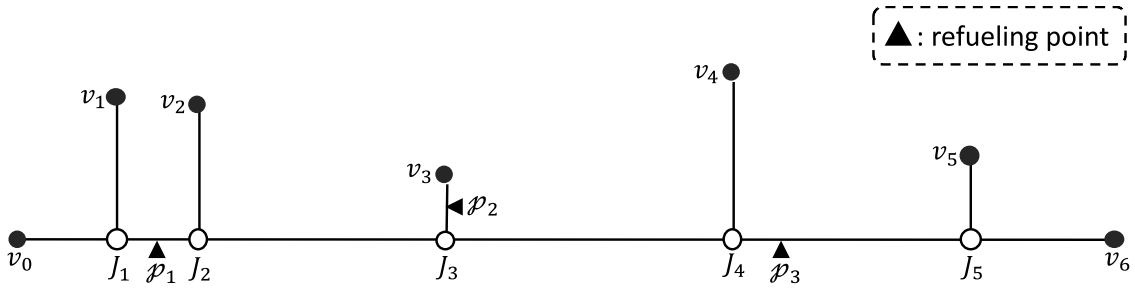



**Figure 3.3** A comb tree with three RPs

### 3.2. Minimum number of RPs needed

To determine the minimum number of RPs that are required to serve all O-D transportation needs, we introduce a two-step algorithm.

#### 3.2.1. Step One --- Comb tree trimming

The trimming procedure is simple: it picks an arbitrary comb tooth of $G$, say $a_k$, if $b_k \geq r$, adds a set $\mathcal{P}_k = \left\{\mathcal{p}_1^k, \cdots, \mathcal{p}_{\left\lfloor \frac{b_k}{r} \right\rfloor}^k\right\}$ of $\left\lfloor \frac{b_k}{r} \right\rfloor$ RP locations to $a_k$, where the $i^{th}$ RP location is at $i * r$ distance away from the endpoint $e_k^1$ of $a_k$, i.e., the leaf node on $a_k$.

It then cuts the leaf node $e_k^1$, the newly added RP locations in $\mathcal{P}_k \backslash \left\{ p_{\left\lfloor \frac{b_k}{r} \right\rfloor}^k \right\}$, and the arcs joining them, and iterates on the remaining comb until the remaining comb has no tooth with length greater than or equal to $r$. The procedure is illustrated in Figure 3.4. Let $a_k = (v_k, J_k)$ be the comb tooth chosen on which the procedure will add RP locations and then cut the comb tree. The RP locations $p_1^k$, $p_2^k$, $p_3^k$ and $p_4^k$ are added, and $v_k$, $p_1^k$, $p_2^k$ and $p_3^k$ are removed from the comb tree afterwards.

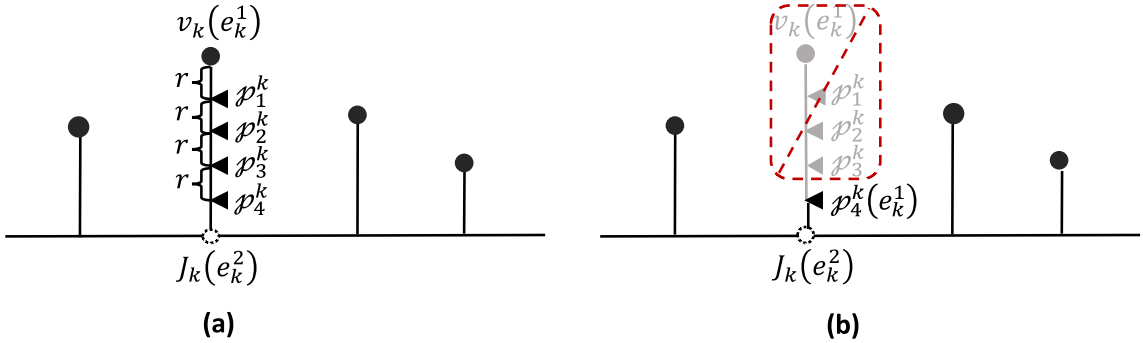



**Figure 3.4** An illustration for trimming procedure

This is a greedy method. After trimming, a new comb tree $G^* = (V^*, A^*)$ will be derived, where

$$V^* = \{v_k^* : v_k \in V(G)\} \cup \{J_k^* : J_k \in V(G)\},$$

$$A^* = \{a_k^* : a_k \in A(G)\},$$

$$b_k^* = b_k - \left\lfloor \frac{b_k}{r} \right\rfloor \cdot r, \text{ for } \forall k \text{ in } \{0, 1, \cdots, n\}, \text{ and}$$

$$f(v_i^*, v_j^*) = f(v_i, v_j), \text{ for } \forall i, j \text{ in } \{0, 1, \cdots, n\}.$$

Moreover, we assign a label "RP" to node $v_k^*$ if node $v_k$ has been cut by the procedure in $G$.

***Proposition 3.1*** Let $m$, $m^*$ be the minimum number of RPs needed to serve all O-D transportation needs on $G$ and $G^*$, respectively. Then, $m = m^* + \sum_{\{a_k : b_k \geq r\}} \left\lfloor \frac{b_k}{r} \right\rfloor$.

*Proof of Proposition 3.1* We know that $m \leq m^* + \sum_{\{a_k:\, b_k \geq r\}} \left\lfloor \frac{b_k}{r} \right\rfloor$, since if flows on $(v_i^*, v_j^*)$ can be served by a set $\mathcal{P}$ of RPs located on $G^*$, then flows on $(v_i, v_j)$ can be served by RPs in $\mathcal{P} \cup \{\bigcup_{\{a_k:\, b_k \geq r\}} \mathcal{P}_k\}$ on the original comb $G$. We shall then prove that $m \geq m^* + \sum_{\{a_k:\, b_k \geq r\}} \left\lfloor \frac{b_k}{r} \right\rfloor$. Consider a one-way trip $(v_i, v_j)$ on $G$, then for the first sub-trip from $v_i$ to $J_i$, the vehicle has to refuel for at least $\left\lfloor \frac{b(v_i, J_i)}{r} \right\rfloor$ times. Conversely, consider another one-way trip $(v_k, v_i)$ on $G$, then for the last sub-trip from $J_i$ to $v_i$. Assume that the vehicle has a remaining fuel of $\varepsilon$ after arriving at $J_i$, then the vehicle has to refuel for at least $\left\lfloor \frac{d(v_i, J_i) - \varepsilon}{r} \right\rfloor$ times. If $0 \leq \varepsilon < r$, $\left\lfloor \frac{d(v_i, J_i)}{r} \right\rfloor = \left\lfloor \frac{d(v_i, J_i) - \varepsilon}{r} \right\rfloor$, since $d(v_i, J_i) - r < d(v_i, J_i) - \varepsilon \leq d(v_i, J_i)$. If $\varepsilon = r$, $\left\lfloor \frac{d(v_i, J_i)}{r} \right\rfloor = \left\lfloor \frac{d(v_i, J_i) - \varepsilon}{r} \right\rfloor + 1$, however, in this case, there is a RP located at junction $J_i$. Since $G^*$ is the smallest comb tree we can get by trimming, then $m^* + \sum_{\{a_k:\, b_k \geq r\}} \left\lfloor \frac{b_k}{r} \right\rfloor \leq m$. ■

EXAMPLE 3.1 Consider the non-oriented comb tree in Figure 3.5. The nodes in $\{v_0, v_1, \cdots, v_7\}$ represent origins and/or destinations, and the arcs represent roads connecting the nodes. The numbers (weights) beside the arcs indicate travel distance along the arcs. Assume that the range limit equals 4. We wish to find the minimum number of RPs that all one-way trips can be served. We are ignoring the possibility of queue formation for battery swapping/recharging services.

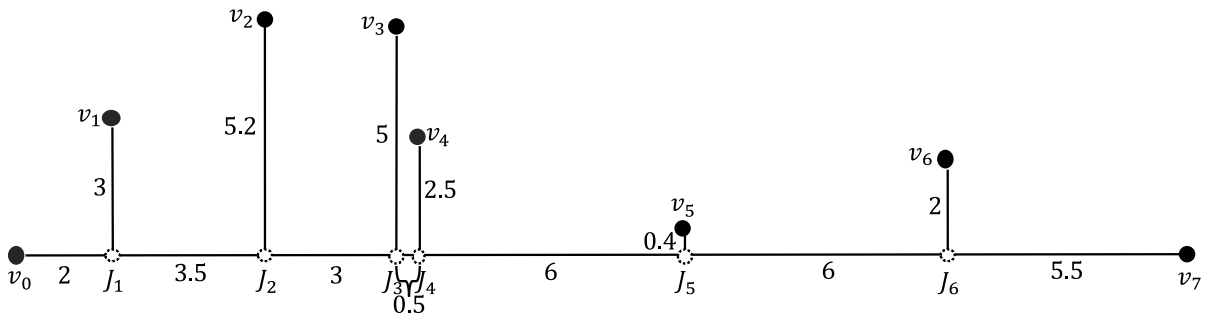



**Figure 3.5** Comb tree representation of O-D nodes and roads for example 3.1

Given the information available in Figure 3.5, we obtain first the trimmed comb tree, as shown in Figure 3.6. After examining every comb tooth, we find that $(v_2,J_2)$, $(v_3,J_3)$ and $(v_7,J_6)$ are the only three teeth with length of more than $r$. Since $\left\lfloor\frac{c(v_2,J_2)}{r}\right\rfloor = \left\lfloor\frac{5.2}{4}\right\rfloor = 1$, we have one public RP located on $(v_2,J_2)$, with distance of 4 from $v_2$. As depicted in Figure 3.6, $v_2$ is now a triangular shape node, indicating that battery swapping/recharging service is available here · and $d(v_2,J_2) = 1.2$. Similarly, another 2 public RPs are located on $(v_3,J_3)$ and$(v_7,J_6)$, respectively.

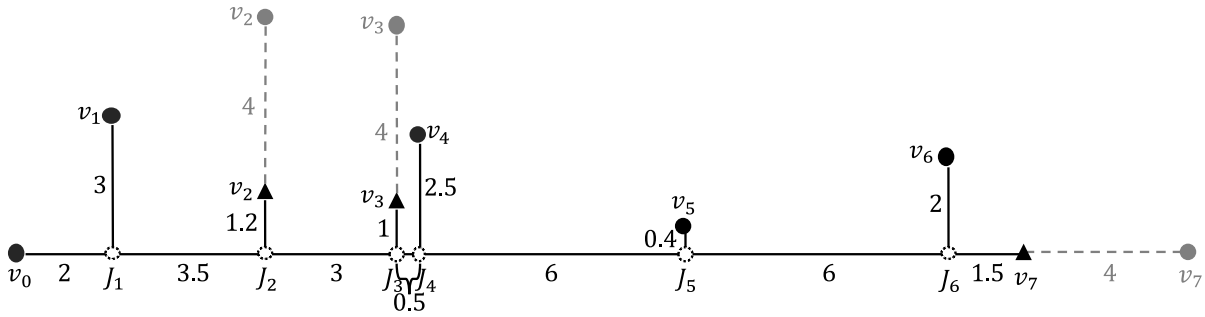



**Figure 3.6** The trimmed comb tree for example 3.1

Hereon, we will be using the trimmed comb tree $G^*$ for algorithm developing, and for notational convenience we simply refer to $G^*$ as $G$ unless otherwise specified.

### 3.2.2. Step Two --- A rightward pass and a leftward pass

Recall that in that simple example by which we started our discussion of the comb tree problem, a subtree (the intersection of all paths between points $A^1$, $A^2$, $B^1$, $B^2$, $C^1$ and $C^2$) was constructed for finding the optimal location for a single RP. Now, we verify that the search for the optimal set of RP locations can be limited to the comb span.

***Proposition 3.2*** The search for the optimal set of RP locations can be limited to the comb span and thus all interior points and the leaf node on comb tooth can be excluded from consideration.

*Proof of Proposition 3.2.* Any RP that is located on an interior point or the leaf node of a comb tooth can be moved to the junction node without any loss of flow and possibly with a decrease of distance of detours. ■

The second step for determining the minimum number $m$ of required RPs consists of a rightward pass and a leftward pass. After this procedure, not only $m$ is determined, but also for each RP a localization segment on the comb span is derived.

#### 3.2.2.1. Rightward pass

We consider a point $x$ on $G$, where $x$ can be either a leaf node of $G$ or a RP point located on the comb span (if any). Assuming that an electric vehicle departs from $x$ with a fully charged battery and heads to node $y$, where $y$ is on the right-hand side of $x$, let $x^+$ denote the farthest point on the comb span that the vehicle is able to reach from $x$ and $x^+$ is on the right-hand side of $x$, where the number of times the vehicle can stop is not limited. Moreover, let $x^+_{p\equiv 0}$ and $x^+_{p>0}$ represent the farthest point on the comb span that the vehicle can reach by restricting the number

of times to stop to zero and to at least one, respectively. Then, $l_1\left(x^+_{p\equiv 0}\right) = l_1(x) + \left(r - l_2(x)\right)$. Denote $\mathcal{R}(x)$ the set of points on $G$, where each element $y$ of $\mathcal{R}(x)$ has been assigned a label "RP", and there exists at least one refueling walk from $x$ to $y$. Then, $l_1\left(x^+_{p>0}\right) = \max\limits_{y\in\mathcal{R}(x)\backslash\{x\}}\left\{l_1\left(y^+_{p\equiv 0}\right)\right\}$. Furthermore, we have $l_1(x^+) = \max\left\{l_1\left(x^+_{p\equiv 0}\right), l_1\left(x^+_{p>0}\right)\right\}$, and $x^+ = \operatorname{argmax}\limits_{y\in\left\{x^+_{p\equiv 0}, x^+_{p>0}\right\}}\{l_1(y)\}$. Reachability between RPs is an equivalence relation, since (let $x$, $y$ and $z$ be three points on $G$, labelled "RP"):

(1) It is reflexive: there is a trivial path of length zero from any node to itself;

(2) It is symmetric: if there is a refueling walk from $x$ to $y$, the same arcs form a refueling walk from $y$ to $x$;

(3) It is transitive: if there is a refueling walk from $x$ to $y$ and a refueling walk from $y$ to $z$, the two walks can be concatenated together to form a refueling walk from $x$ to $z$.

***Fact 3.1*** Let $x$ and $y$ be two points on $G$, labelled "RP". If $d(x, y) \le r$, then we have $\mathcal{R}(x) = \mathcal{R}(y)$, and $x^+ = y^+$.

*Proof of Fact 3.1* Since the reachability between RPs is an equivalence relation, it is trivial to see that $\mathcal{R}(x) = \mathcal{R}(y)$. We shall prove $x^+ = y^+$:

$$
\begin{aligned}
l_1(x^+) &= \max\left\{l_1\left(x^+_{p\equiv 0}\right), l_1\left(x^+_{p>0}\right)\right\} \\
&= \max\left\{l_1\left(x^+_{p\equiv 0}\right), \max\left\{l_1\left(z^+_{p\equiv 0}\right) \mid z \in \mathcal{R}(x)\backslash\{x\}\right\}\right\} \\
&= \max\left\{l_1\left(x^+_{p\equiv 0}\right), l_1\left(y^+_{p\equiv 0}\right), \max\left\{l_1\left(z^+_{p\equiv 0}\right) \mid z \in \mathcal{R}(x)\backslash\{x, y\}\right\}\right\} \\
&= \max\left\{l_1\left(x^+_{p\equiv 0}\right), l_1\left(y^+_{p\equiv 0}\right), \max\left\{l_1\left(z^+_{p\equiv 0}\right) \mid z \in \mathcal{R}(y)\backslash\{x, y\}\right\}\right\} \text{ (by } \mathcal{R}(x) = \mathcal{R}(y)\text{)} \\
&= \max\left\{l_1\left(y^+_{p\equiv 0}\right), \max\left\{l_1\left(z^+_{p\equiv 0}\right) \mid z \in \mathcal{R}(y)\backslash\{y\}\right\}\right\} \\
&= \max\left\{l_1\left(y^+_{p\equiv 0}\right), l_1\left(y^+_{p>0}\right)\right\}
\end{aligned}
$$

$= l_1(y^+)$.

Since $x^+$ and $y^+$ are two points on the comb span, by proving $l_1(x^+) = l_1(y^+)$, we know that $x^+ = y^+$. ■

**Fact 3.2** For $\forall\, y \in \mathcal{R}(x)$, $x^+ = y^+$.

*Proof of Fact 3.2* $l_1(x^+) = \max\{l_1(z^+_{p\equiv 0}) \mid z \in \mathcal{R}(x)\} = \max\{l_1(z^+_{p\equiv 0}) \mid z \in \mathcal{R}(y)\} = l_1(y^+)$. ■

Now let us restate $\mathcal{R}(x)$ in a different way. Denote $\sim$ this binary reachability relation on set $V_{\mathcal{P}}$ of points labelled "RP", that is, the equivalence class of $x \in G$ under $\sim$ is defined as $\mathcal{R}(x) = [x] = \{y \in V_{\mathcal{P}} \mid y \sim x\}$. Then the set of all possible equivalence classes of $V_{\mathcal{P}}$ by $\sim$ is denoted $V_{\mathcal{P}}{}^{\sim} := \{[x] \mid x \in V_{\mathcal{P}}\}$. We call that $rep([x])$ the representative of equivalence class $[x]$, which is defined as $rep([x]) := \underset{y\in[x]}{\operatorname{argmax}}\{l_1(y^+_{p\equiv 0})\}$. Given comb tree $G$, we can come up with a graph $G^{\sim} = (V^{\sim}, A^{\sim})$ indicating the reachability between RPs, where

$$V^{\sim} = V_{\mathcal{P}}{}^{\sim},$$

$$A^{\sim} = \{(v_i^{\sim}, v_j^{\sim}) \text{: if } d(v_i^{\sim}, v_j^{\sim}) \le r \text{ on the origianl comb}\},$$

that is, each edge corresponds to a range-limited shortest path (i.e., a path of length less than or equal to $r$). The graph $G^{\sim}$ will have one or more connected components, each of which is formed by an equivalence class of the relation $\sim$.

While, for a leaf node $v$ that is not in $V_{\mathcal{P}}$, i.e., no public RP has been located at $v$, we let $v \rightsquigarrow x$ denote that if an electric vehicle starts from $v$ with a fully charged battery, it is able to reach point $x$ and $x$ is labelled "RP". By the transitive property of the reachability relation $\sim$ between RPs, we know that $v \rightsquigarrow y$ for $\forall y \in [x]$, if $v \rightsquigarrow x$.

Similarly, let $v^+$ represent the farthest point on the comb span that the vehicle is able to reach from $v$ and $v^+$ is on the right-hand side of $v$. Then we have

$$l_1(v^+) = \max\left\{l_1\left(v^+_{p\equiv 0}\right), \max\{l_1(x^+) \mid v \rightsquigarrow x\}\right\}$$

$$= \max\left\{l_1\left(v^+_{p\equiv 0}\right), \max\{l_1(x^+) \mid d(v\,, x) \le r\}\right\}.$$

If we replace every connected component $C_k$ of $G^{\sim}$ by a single node $c_k$, we can construct a bipartite graph $G^B = (B_1, B_2, E)$, where

$B_1 = \{\bigcup_{i=0}^{n} v_i\} \backslash V_{\mathcal{P}}$, the set of leaf nodes in $G$ that are without public RPs,

$B_2 = \bigcup_k c_k$,

$E = \{(v, c_k) \mid \text{if for some } x \in V(C_k), v \rightsquigarrow x\}$.

These concepts will be illustrated in example 3.2.

Now let us formally introduce the algorithm FARTHEST-POINT-ON-RHS($G$), by which we are able to find the farthest point that a vehicle can reach from $x$ with a fully charged battery. Furthermore, consider a one-way trip $\left(v_i, v_j\right)$, and without loss of generality we assume that $l_1(v_i) < l_1\left(v_j\right)$. Then we can use this algorithm to find the position $l_1({v_i}^+)$ on the comb at or before which an RP must be located, otherwise the flows from $v_i$ to $v_j$ cannot be served.

**FARTHEST-POINT-ON-RHS($G$)**

| | |
|---|---|
| *Step* 1 | Compute the minimum distance matrix |
| *Step* 2 | Compute $l_1\left(v^+_{p\equiv 0}\right)$ |
| *Step* 3 | Construct graph $G^{\sim}$ on node set $V_{\mathcal{P}}$ |
| | For each $v$ in $V_{\mathcal{P}}$, add $v$ to $G^{\sim}$ |
| | Add $(u, v)$ to $G^{\sim}$, if $u, v \in G^{\sim}$ and $d(u, v) \le r$ |
| *Step* 4 | Compute the connected components of $G^{\sim}$ |
| *Step* 5 | For each connected component $C_k$ of $G^{\sim}$ |

$$rep\big(V(C_k)\big) \leftarrow \underset{v\in V(C_k)}{\operatorname{argmax}}\{l_1(v^+_{p\equiv 0})\}$$

*Step* 6 For each $v$ in $V_{\mathcal{P}}$

$$l_1(v^+) \leftarrow l_1\left(rep\big(V(C_k)\big)^+_{p\equiv 0}\right) \text{ if } v \in V(C_k)$$

*Step* 7 For each node $v$ without an RP

$$l_1(v^+) \leftarrow \max\left\{l_1(u^+_{p\equiv 0}), \max\{l_1(u^+) \mid d(u\,, v) \le r\}\right\}$$

EXAMPLE 3.2. In this example, we illustrate the computation procedure of the algorithm FARTHEST-POINT-ON-RHS($G$) on the comb tree of Figure 3.6.

*Step* 1: We first compute the minimum distance between each leaf node and each node labelled "RP" ($v_2, v_3$ and $v_7$):

$$D_{3\times 8} = \begin{array}{c} \\ v_2 \\ v_3 \\ v_7 \end{array}\begin{array}{c} \begin{array}{cccccccc} v_0 & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 \end{array} \\ \begin{pmatrix} 6.7 & 7.7 & \backslash & 5.2 & 7.2 & 11.1 & 18.7 & 18.2 \\ 9.5 & 10.5 & 5.2 & \backslash & 4 & 7.9 & 15.5 & 15 \\ 22.5 & 23.5 & 18.2 & 15 & 16 & 7.9 & 3.5 & \backslash \end{pmatrix} \end{array}.$$

*Step* 2: Then for each node $v$ we compute $l_1(v^+_{p\equiv 0})$:

$$\left(l_1(v^+_{p\equiv 0})\right)_{1\times 8} = \begin{pmatrix} v_0 & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 \\ 4 & 3 & 8.3 & 11.5 & 10.5 & 18.6 & 23 & 26.5 \end{pmatrix},$$

where $l_1\left(v_i{}^+_{p\equiv 0}\right) = l_1(v_i) + r - l_2(v_i)$.

*Step* 3-7: Now we can construct the graph $G^{\sim}$ in Figure 3.7(a) indicating the reachability between leaf nodes labelled "RP". Note that each of $v_2$, $v_3$ and $v_7$ itself forms a connected component, since the minimum distance between any of two nodes is greater than the range limit ($d(v_2, v_3) = 5.2$, $d(v_2, v_7) = 18.2$ and $d(v_3, v_7) = 15$). Then we have $l_1({v_2}^+) = l_1\left(v_2{}^+_{p\equiv 0}\right) = 8.3$, $l_1({v_3}^+) = l_1\left(v_3{}^+_{p\equiv 0}\right) = 11.5$, and $l_1({v_7}^+) = l_1\left(v_7{}^+_{p\equiv 0}\right) = 26.5$. In Figure 3.7(b), we see that there is a link connecting leaf node $v_4$ and component node $c_2$ since $d(v_4, v_3) = 4 \le r$, and a link connecting $v_6$ and $c_3$ since $d(v_6, v_7) = 3.5 \le r$. Then, we can compute $l_1({v_4}^+) =$

$\max\left\{l_1\left(v_{4_{p\equiv 0}}^{+}\right), l_1(v_3{}^{+})\right\} = \max\{10.5, 11.5\} = 11.5$ and $l_1(v_6{}^{+}) =$

$\max\left\{l_1\left(v_{6_{p\equiv 0}}^{+}\right), l_1(v_7{}^{+})\right\} = \max\{23, 26.5\} = 26.5$. While for any other node $v_j$ in $B_1$,

$l_1(v_j{}^{+}) = l_1\left(v_{j_{p\equiv 0}}^{+}\right)$, hence, $l_1(v_0{}^{+}) = 4$, $l_1(v_1{}^{+}) = 3$, $l_1(v_5{}^{+}) = 18.6$.

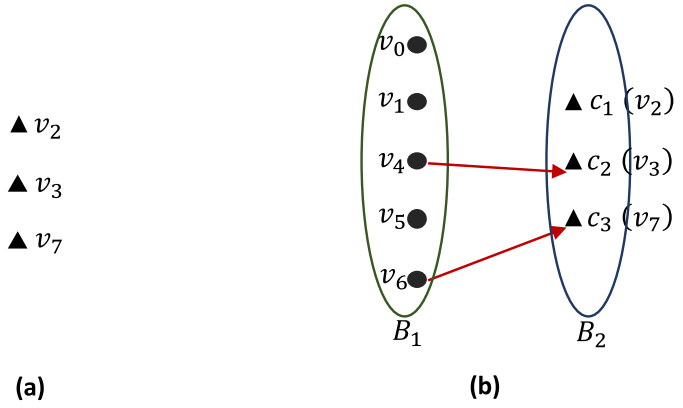



**Figure 3.7** (a) The reachability graph $\boldsymbol{G}^{\sim}$ constructed on $\{\boldsymbol{v_2}, \boldsymbol{v_3}, \boldsymbol{v_7}\}$
(b) The bipartite graph constructed for example 3.2

Algorithm RIGHTWARD-PASS(G)

Now, we are able to devise the following recursive algorithm, RIGHTWARD-PASS($G$), for identifying an "extreme" site at which an RP must be located in each iteration, otherwise some flows cannot be served. Let $\boldsymbol{\mathcal{P}} = \{\mathcal{p}_1, \mathcal{p}_2, \cdots, \mathcal{p}_m\}$ represent the set of RPs to be located on the comb span, where $m$ is the minimum number of RPs required. The basic idea in this greedy algorithm is to use a simple rule to identify the first farthest possible site $\beta_1$ from $v_0$. Once $\mathcal{p}_1$ is supposed to be located at site $\beta_1$, we obtain a minor of the comb by cutting the leaf and junction nodes on the left-hand side of $\beta_1$, and converting $\beta_1$ to a leaf node labelled "RP". We then identify the second farthest possible site $\beta_2$ from $\beta_1$, and again obtain a minor of the comb by cutting leaf and junction nodes on the left-hand side of $\beta_2$, and converting $\beta_2$

to a leaf node. We continue in this fashion until we reach the end of the comb span $v_n$.

Let us state the algorithm a bit more formally. In the pseudocode below, we again use $V_{\mathcal{P}}$ denoting the set of leaf nodes labelled "RP" in $G$.

**RIGHTWARD-PASS($G$)**

---

*Step* 1 Initially let $G$ be the comb tree, and let $\boldsymbol{\beta}$ be empty

*Step* 2 While $V(G)$ is not empty

Call method **FARTHEST-POINT-ON-RHS($G$)**

If $(l_1(u^+) \equiv l_1(v^+) \geq l_1(v_n)$ for any two nodes $u, v \in V_{\mathcal{P}})$ && $(l_1(u^+) \geq l_1(v^+)$ for any $u \in V \backslash V_{\mathcal{P}}$ and $v \in V_{\mathcal{P}})$

Stop

Else

$\beta_k \leftarrow \underset{u \in V(G)}{\operatorname{argmin}}\{l_1(u^+)\}$

$\boldsymbol{\beta} \leftarrow \boldsymbol{\beta} \cup \{\beta_k\}$

Insert a new node at $\beta_k$, labelled "RP"

Cut the comb into two smaller combs at $\beta_k$

$G \leftarrow$ Right comb

*Step* 3 Return $\boldsymbol{\beta}$

---

Remarks to the algorithm RIGHTWARD-PASS($G$)

The validity of the algorithm termination condition may be established by proving the following proposition.

***Proposition 3.3*** The following assertions are equivalent for a comb $G$.

i). There is a refueling walk between every pair of leaf nodes.

ii). The reachability graph $G^{\sim}$ has exactly one connected component $C_1$, and for every node $u \in V \backslash V_{\mathcal{P}}$, there exists a node $v \in V_{\mathcal{P}}$ (i.e., labelled "RP") such that $d(u, v) \leq r$.

iii). For any two nodes $u, v \in V_{\mathcal{P}}$, $l_1(u^+) \equiv l_1(v^+) \geq l_1(v_n)$; and for any $u \in V \backslash V_{\mathcal{P}}$ and $v \in V_{\mathcal{P}}$, $l_1(u^+) \geq l_1(v^+)$.

*Proof of Proposition 3.3* By algorithm FARTHEST-POINT-ON-RHS($G$), we know that assertions ii) and iii) are equivalent. Now suppose that assertion i) is true. It is trivial to see that ii) is true, since there are no unreachable nodes. Now suppose that assertion ii) is true. It is also trivial to see that there is a refueling walk between every pair $(u, v)$ of nodes, where

(a) $u, v \in V_{\mathcal{P}}$, since $G^{\sim}$ is connected;

(b) $u \in V \backslash V_{\mathcal{P}}$ and $v \in V_{\mathcal{P}}$, since there exists some $w \in V_{\mathcal{P}}$ such that $d(u, w) \leq r$ and (a);

(c) $u, v \in V \backslash V_{\mathcal{P}}$, since (b) and (a). ■

After iteration $k$, $k$ RPs are supposed to have been located at sites $\beta_1, \cdots, \beta_k$ on the comb span.

***Proposition 3.4*** The rightward pass algorithm returns a feasible set of RP locations, $\boldsymbol{\beta} = \{\beta_1, \cdots, \beta_m\}$.

*Proof of Proposition 3.4* To prove this proposition, we need to show that there is a refueling walk between every pair of nodes.

We first prove by induction that, for all $k \in \{1, \cdots, m\}$, there is a refueling walk between every pair of nodes $u$ and $v$ with $l_1(u), l_1(v) \leq l_1(\beta_k)$, after RPs $\beta_1$, $\beta_2$, ⋯, and $\beta_k$ have been located on the comb span $(*)$.

Base case: When $k = 1$, trivial, by greedy rule.

Induction step: Let $l \in \{1, \cdots, m\}$ be given and suppose $(*)$ is true for $k = l$. Then :

(a) $\beta_l$ and $\beta_{l+1}$ are mutually reachable; *(by greedy rule)*

(b) node $v$ and $\beta_l$ and $\beta_{l+1}$ are mutually reachable, where $l_1(v) \in (l_1(\beta_l), l_1(\beta_{l+1})]$; *(by greedy rule and (a))*

(c) nodes $u$ and $v$ are mutually reachable, where $l_1(u), l_1(v) \in (l_1(\beta_l), l_1(\beta_{l+1})]$; *(by (b))*

(d) nodes $u$ and $v$ are mutually reachable, where $l_1(u) \leq l_1(\beta_l)$ and $l_1(v) \in (l_1(\beta_l), l_1(\beta_{l+1})]$. *(by induction hypothesis and (b))*

Thus, $(*)$ holds for $k = l + 1$, and the proof of the induction step is complete. By the principle of induction, $(*)$ is true for all $k \in \{1, \cdots, m\}$.

Recall the stopping criterion of the rightward pass algorithm. We know that every node $v$ and $\beta_m$ are mutually reachable, where $l_1(v) > l_1(\beta_m)$. Hence, by $(*)$, there is a refueling walk between every pair of nodes. ■

EXAMPLE 3.3 We illustrate two iterations of the rightward pass algorithm on the comb shown in Figure 3.6.

Recall that in example 3.2, we computed $l_1(v^+)$ for each $v$:

$$\left(l_1(v^+)\right)_{1\times 8} = \begin{pmatrix} v_0 & v_1 & \boxed{v_2} & \boxed{v_3} & v_4 & v_5 & v_6 & \boxed{v_7} \\ 4 & 3 & 8.3 & 11.5 & 11.5 & 18.6 & 26.5 & 26.5 \end{pmatrix},$$

In the first iteration: note that $l_1({v_2}^+) \neq l_1({v_3}^+) \neq l_1({v_7}^+)$, hence, we let$l_1(\beta_1) = \min_{v\in\{v_0,\cdots,v_7\}}\{l_1(v^+)\} = l_1({v_1}^+) = 3$. Then we insert a RP node at $\beta_1$, split the comb into two smaller combs, and let the right comb be the input in the next iteration.

In the second iteration:

$$\left(l_1(v^+)\right)_{1\times 7} = \begin{pmatrix} \boxed{\beta_1} & \boxed{v_2} & \boxed{v_3} & v_4 & v_5 & v_6 & \boxed{v_7} \\ 8.3 & 8.3 & 11.5 & 11.5 & 18.6 & 26.5 & 26.5 \end{pmatrix},$$

Again we have $l_1({v_2}^+) \neq l_1({v_3}^+) \neq l_1({v_7}^+)$. Then we let $l_1(\beta_2) = \min_{v\in\{\beta_1,v_2,\cdots,v_7\}}\{l_1(v^+)\} =$ 8.3, insert a station node at $\beta_2$, and split the comb.

In the next three iterations, we get $\beta_3 = 12.3$, $\beta_4 = 16.3$, and $\beta_5 = 20.3$.

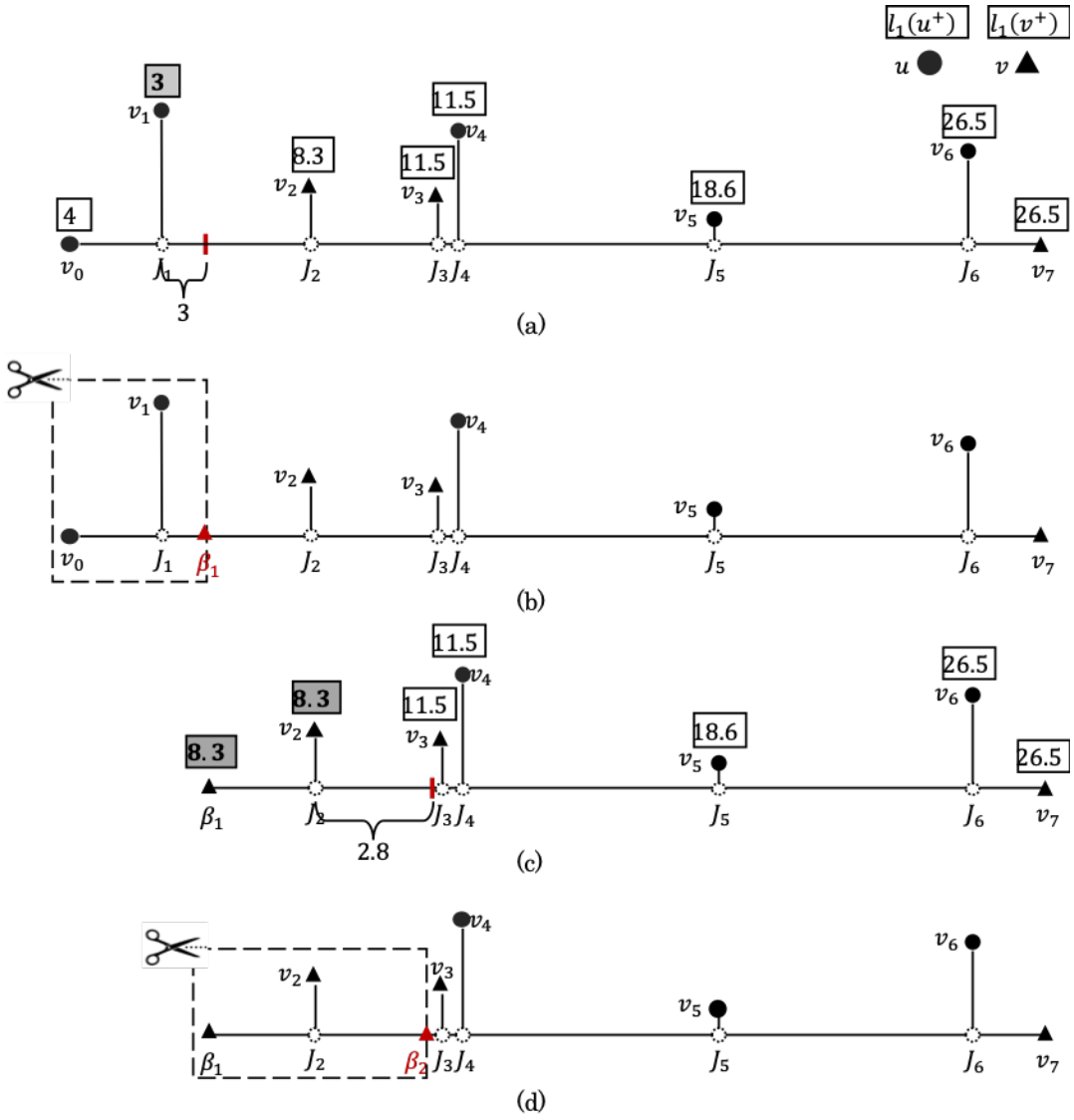



**Figure 3.8** Illustrating the rightward pass

### 3.2.2.2. Leftward pass

By performing the rightward pass algorithm, a set of extreme none refueling detouring (*XNRD*) RP sites has been identified. Then, in the leftward pass, we shall start from the rightmost (i.e., leaf node $v_n$) of the comb tree and derive a set of *XNRD* RP sites iteratively.

Consider a point $x$ on $G$, where $x$ can be either a leaf node of $G$ or a RP point that is supposed to be established on the comb span (if any). Let $x^-$ denote the farthest point on the comb span that the electric vehicle is able to reach from $x$ and $x^-$ is on the left-hand side of $x$. Let $x^-_{p\equiv 0}$ and $x^-_{p>0}$ represent the farthest point on the comb span that the vehicle can reach by restricting the number of times to stop to zero and to at least one, respectively. Then we have $l_1\left(x^-_{p\equiv 0}\right) = l_1(x) - \left(r - l_2(x)\right)$, and $l_1\left(x^-_{p>0}\right) = \min\limits_{y\in\mathcal{R}(x)\setminus\{x\}}\left\{l_1\left(y^-_{p\equiv 0}\right)\right\}$, where $\mathcal{R}(x)$ is the set of RP points reachable to $x$. Furthermore, we have $l_1(x^-) = \min\left\{l_1\left(x^-_{p\equiv 0}\right), l_1\left(x^-_{p>0}\right)\right\}$, and $x^- = \underset{y\in\{x^-_{p\equiv 0}, x^-_{p>0}\}}{\operatorname{argmin}}\{l_1(y)\}$.

To determine $l_1(v^-)$, there is no need to rewrite another algorithm, say, FARTHEST-POINT-ON-LHS($G$)). Note that by reversing the comb tree we can apply FARTHEST-POINT-ON-RHS($G$)).

**Reverse-Comb($\boldsymbol{G}$)**

*Step* 1 Initially let $G$ be the original comb graph, let $G^{Reverse}$ be empty, and $L = 0$

*Step* 2 $L \leftarrow$ length of the comb span

*Step* 3 $G^{Reverse} \leftarrow G$

*Step* 4 For each $v$ in $V(G^{Reverse})$

$l_1(v) \leftarrow L - l_1(v)$

*Step* 5 Return $G^{Reverse}$

Therefore, we have the following algorithm for the leftward pass:

**LEFTWARD-PASS($\boldsymbol{G}$)**

*Step* 1 Initially let $G$ be the comb tree, let $G'$ be empty, $\boldsymbol{\alpha}$ be empty, and $L = 0$

*Step* 2 $L \leftarrow$ length of the comb span

*Step 3* Call method **REVERSE-COMB($\boldsymbol{G}$)**, $G' \leftarrow$ REVERSE-COMB($G$)

| | |
|---|---|
| *Step* 4 | Call method **RIGHTWARD-PASS($G'$)**, $\boldsymbol{\alpha} \leftarrow$ RIGHTWARD-PASS($G'$) |
| *Step* 5 | For each $\alpha_k$ in $\boldsymbol{\alpha} \leftarrow$ Reverse($L - \boldsymbol{\alpha}$) |
| *Step* 6 | Return $\boldsymbol{\alpha}$ |

Remarks to the algorithm LEFTWARD-PASS($G$)

***Proposition 3.5*** The leftward pass algorithm returns a feasible set of RP locations, $\boldsymbol{\alpha} = \{\alpha_1, \alpha_2, \cdots, \alpha_{m^-}\}$.

*Proof of Proposition 3.5* Directly by proposition 3.4. ■

EXAMPLE 3.3 (Continue) By running leftward pass algorithm on the same comb tree, we get $\alpha_1 = 1.5$, $\alpha_2 = 5.5$, $\alpha_3 = 10.5$, $\alpha_4 = 14.5$, and $\alpha_5 = 18.5$.

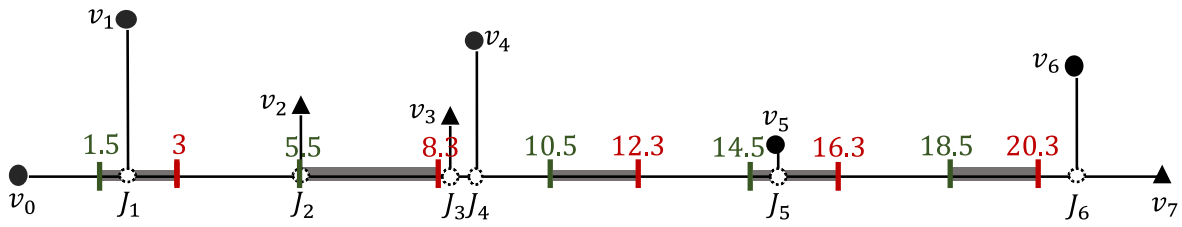



**Figure 3.9** Localization segments

### 3.2.3. Analyzing the algorithm

In this section, we want to justify that our proposed greedy method returns a minimum RP set.

As a start, we declare that $m^+ = m^-$, where $m^+$ denotes the cardinality of the set of extreme none refueling detouring RP sites, $\{\beta_1, \beta_2, \cdots, \beta_{m^+}\}$, derived by rightward pass algorithm, and where $m^-$ denotes the cardinality of the set of extreme none refueling detouring RP sites, $\{\alpha_1, \alpha_2, \cdots, \alpha_{m^-}\}$, derived by leftward pass algorithm.

***Proposition 3.6*** $\alpha_1 \leq \beta_1$

Proof of Proposition 3.6 By termination condition, we know that there is a refueling walk between every pair of nodes in $\{v: l_1\{v\} \leq \alpha_1\} \cup \{\alpha_1\}$. That is, every node in $\{v: l_1(v) \leq \alpha_1\}$ is able to reach $\alpha_1$. Then by our greedy rule, we have $\alpha_1 \leq \beta_1$.■

***Proposition 3.7*** $m^+ = m^-$

*Proof of Proposition 3.7* Suppose on the contrary that $m^+ \geq m^- + 1$. Consider the following $m^-$ segments on the comb span:

$[\alpha_1, \alpha_2)$, $[\alpha_2, \alpha_3)$, ⋯, $[\alpha_{m^--1}, \alpha_{m^-})$ and $\left[\alpha_{m^-}, l_1(v_n)\right)$.

Then by proposition 3.5 and by pigeonhole principle, we know that at least one of these segments will contain more than one $\beta$. Without loss of generality, suppose that $\beta_k, \beta_{k+1} \in [\alpha_l, \alpha_{l+1})$. Recall the rightward pass algorithm. The greedy rule tells us that given that $k$ RPs have been established at $\{\beta_1, \beta_2, \cdots, \beta_k\}$, there exists some node $v^*$ with $\beta_k \leq l_1(v) \leq \beta_{k+1}$ such that the farthest point (away from $v_0$) it can reach is $\beta_{k+1}$.

Now let us consider the leftward pass. Suppose that RPs have been established at $\{\alpha_{l+1}, \alpha_{l+1}, \cdots, \alpha_{m^-}\}$. If the electric vehicle departs from $\alpha_{l+1}$ with a fully charged battery, then the remaining level of charge would be $r - (\alpha_{l+1} - \beta_{k+1})$. That is, the vehicle is not able to reach node $v^*$. ■

***Theorem 3.1*** The rightward pass algorithm returns a minimum set of RP locations, $\boldsymbol{\beta} = \{\beta_1, \beta_2, \cdots, \beta_{m^+}\}$.

Since there may exist many sets of RP locations that can serve all O-D transportation needs and are with minimum cardinality, so for purposes of comparison, let $\mathcal{O}^*$ be a single one of them. We wish to show that $m^+ = |\boldsymbol{\beta}| = |\mathcal{P}^*|$. That is, $\boldsymbol{\beta}$ contains the same number of RPs as $\mathcal{P}^*$ and therefore the number of RPs in $\boldsymbol{\beta}$ is minimum also. We introduce some notation to help with this proof. Let the

set of RPs in $\mathcal{P}^*$ be denoted by $\wp_1, \wp_2, \ldots, \wp_m$, assuming that they are ordered in the left-to-right order by the corresponding positions on the comb span.

Recall that our intuition for this greedy method came from wanting our comb tree to become as "small" as possible after establishing the first RP on comb span and cutting the comb. Indeed, our greedy rule guarantees that $\beta_1 \succcurlyeq \wp_1$. In this sense, we want to show that our greedy rule "stays ahead" if we measure the algorithm's progress in a step-by-step fashion. That is, each RP in $\boldsymbol{\beta}$ has been established at a "right-er" position than the corresponding RP in the set $\mathcal{P}^*$. Thus, we now prove that for each $k \geq 1$, we have $\beta_k \succcurlyeq \wp_k$.

***Proposition 3.8*** For all indices $k \leq m$, we have $\beta_k \succcurlyeq \wp_k$.

*Proof of Proposition 3.8* We shall prove this statement by induction.

Base case: When $k = 1$, trivial, by greedy rule.

Induction step: Let $l \in \{1, \ldots, |\mathcal{P}^*|\}$ be given and suppose that the statement is true for $k = l$, and we will try to prove it holds true for $k = l + 1$. The induction hypothesis lets us assume that $\beta_l \succcurlyeq \wp_l$. In order for algorithm's $(l+1)^{th}$ RP not to be located at a "right-er" position compared to $\wp_{l+1}$, it would need to "stay closer to $v_0$" as shown in Figure 3.10. But there's a simple reason why this could not happen: rather than choose a position that is closer to $v_0$, our greedy algorithm always has the option (at worst) of choosing $\wp_{l+1}$ and thus fulfilling the induction step. ■

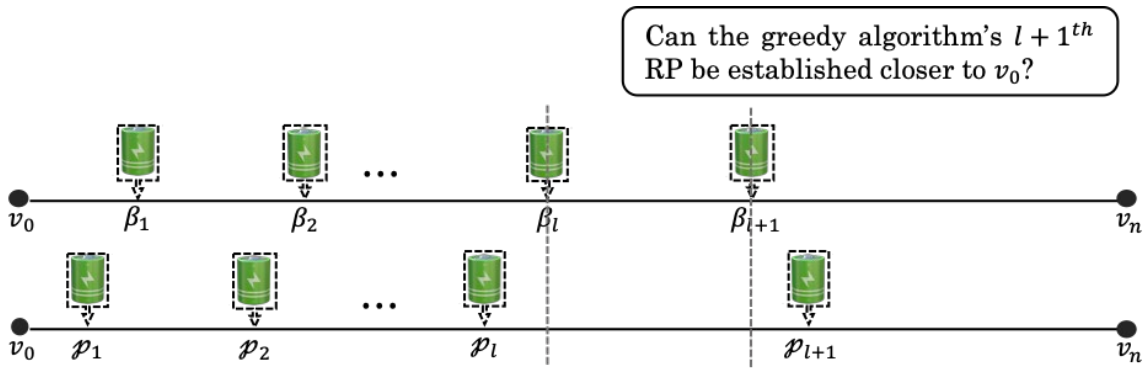



**Figure 3.10** Can the greedy algorithm's $(\boldsymbol{l+1})^{\boldsymbol{th}}$ RP be established closer to $\boldsymbol{v_0}$?

Now, we shall prove theorem 3.1.

*Proof of theorem 3.1* We will prove the theorem by contradiction. Suppose on the contrary we have $m^+ > m$. That is, at least one more RP in $\boldsymbol{\beta}$ would be located after $\mathcal{p}_m$. By proposition 3.6, we know that $\beta_m \succcurlyeq \mathcal{p}_m$. Clearly, there is no need to locate any RPs after $\beta_m$ has been established. ■

**Remarks**

- After performing the rightward pass and the leftward pass algorithm, we will identify a set of localization segments on the comb span, $\{[\alpha_k, \beta_k], k = 1,2,\cdots,m\}$.
- A junction node $J \in V$ is called an *internal junction* if it is an interior or boundary point of a localization segment, otherwise, it is called an *external junction*.

## 3.3. Math Programming Formulation

### 3.3.1. Properties of shortest refueling walk

Given a comb graph $G$ and a set of refueling points $\boldsymbol{\mathcal{P}}$ with fixed locations on the comb span, we denote by $G^{\mathcal{P}}$ the graph on $V(G) \cup \boldsymbol{\mathcal{P}}$ in which each node $\mathcal{p} \in \boldsymbol{\mathcal{P}}$ has been established at its designated location. Then every node $v \in V(G^{\mathcal{P}})$ has a left neighbor $\mathcal{p}_{ne_v^-}$ and a right neighbor $\mathcal{p}_{ne_v^+}$ in $\boldsymbol{\mathcal{P}}$, where

$$\mathscr{p}_{ne_v^-} := \underset{\mathscr{p}\in\mathcal{P}:\, l_1(\mathscr{p})<l_1(v)}{\arg\min} |l_1(v) - l_1(\mathscr{p})|, \tag{3.1}$$

$$\mathscr{p}_{ne_v^+} := \underset{\mathscr{p}\in\mathcal{P}:\, l_1(\mathscr{p})>l_1(v)}{\arg\min} |l_1(v) - l_1(\mathscr{p})|. \tag{3.2}$$

Let $s, t \in V(G^{\mathcal{P}})$ represent the starting and ending nodes of a trip by the electric vehicle. Let $\boldsymbol{\mathcal{W}}_{(s,t)}$ represent the set of refueling walks from $s$ to $t$. For $W \in \boldsymbol{\mathcal{W}}_{(s,t)}$ we put $\delta(W) = L(W) - d(s,t)$, where $L(W)$ is the length of $W$ and $d(s,t)$ is the shortest distance from $s$ to $t$, i.e., $\delta(W)$ denotes the refueling detouring distance that arises from taking walk $W$. Let $\Delta(s,t)$ denote the shortest refueling walk detouring distance from $s$ to $t$:

$$\Delta(s,t) = \begin{cases} \min\limits_{W\in\boldsymbol{\mathcal{W}}_{(s,t)}} \{\delta(W)\}, & \text{if there is a refueling walk } W \text{ from } s \text{ to } t \\ \infty, & \text{otherwise} \end{cases}. \tag{3.3}$$

Let $[\mathscr{p}_{ne_s^+}, \mathscr{p}_{ne_t^-}] = \{\mathscr{p}_{ne_s^+}, \mathscr{p}_{ne_s^+ +1}, \cdots, \mathscr{p}_{ne_t^-}\}$ denote the set of RPs that are established on the comb span and between nodes $s$ and $t$, and let $|[\mathscr{p}_{ne_s^+}, \mathscr{p}_{ne_t^-}]|$ denote the cardinality of the set. Then for $W \in \boldsymbol{\mathcal{W}}_{(s,t)}$, $[\mathscr{p}_{ne_s^+}, \mathscr{p}_{ne_t^-}] \subseteq V(W)$. While note that, $|[\mathscr{p}_{ne_s^+}, \mathscr{p}_{ne_t^-}]| \geq 1$ if $ne_t^- \geq ne_s^+$, and $|[\mathscr{p}_{ne_s^+}, \mathscr{p}_{ne_t^-}]| = 0$ if $ne_t^- = ne_s^-$ otherwise. Let walks between $\{s,t\} \cup [\mathscr{p}_{ne_s^+}, \mathscr{p}_{ne_t^-}]$ be called subwalks of $W$. Also, besides RP nodes in $[\mathscr{p}_{ne_s^+}, \mathscr{p}_{ne_t^-}]$, $W$ may contain other RP nodes that are on the left-hand side of $s$ or on the right-hand side of $t$.

Using our definitions of shortest refueling walk. We can come up with several properties.

***Property 3.1*** Let $W = W^1 \uplus W^2 \uplus \cdots \uplus W^k$ be a shortest refueling walk that goes from $s$ to $t$ through the subwalks $W^1$ through $W^k$. Any subwalk $W^i$ must be a shortest refueling walk from the origin to the destination of $W^i$. That is, a shortest

refueling walk is constructed of shortest refueling walks between any two nodes in $\{s,t\} \cup \left[\mathcal{p}_{ne_s^+}, \mathcal{p}_{ne_t^-}\right]$.

*Proof of Property 3.1* Suppose the assertion is false and $\widetilde{W}^i$ is a shorter refueling walk from the origin to the destination of $W^i$. Since the vehicle is always allowed to refuel at a RP node. Then if we replace $W^i$ in $W$ with $\widetilde{W}^i$, a new walk, $\widetilde{W} = W^1 \uplus \cdots \uplus \widetilde{W}^i \uplus \cdots \uplus W^k$, is found, which is both feasible and shorter than $W$, contradicting its optimality. ■

Property 3.1 is called the optimal substructure property.

***Property 3.2*** Let $W$ be a shortest refueling walk from $s$ to $t$. Let $v \in V\left(G^{\mathcal{P}}\right)$ be a comb leaf node without public refueling infrastructure implemented. Then, $v \notin V(W)\backslash\{s,t\}$.

***Property 3.3*** Let $W$ be a shortest refueling walk from $s$ to $t$. Let $v \in V(W)$ be a node with public refueling infrastructure implemented. Then $v$ appears exactly once in $W$.

***Property 3.4*** Let $W = W^1 \uplus W^2 \uplus \cdots \uplus W^k$ be a shortest refueling walk that goes from $s$ to $t$ through the subwalks $W^1$ through $W^k$. Let $p$ be a refueling point in $V\left(G^{\mathcal{P}}\right)$ on the left-hand side of $s$. Then if $d(s,p) > r$, $p \notin V(W^1)$. Similarly, let $p'$ be a refueling point in $V\left(G^{\mathcal{P}}\right)$ on the right-hand side of $s$. Then if $d(t,p') > r$, $p' \notin V\left(W^k\right)$.

*Proof of Property 3.4* Suppose the assertion is false and $p \in V(W^1)$. Let $p.\boldsymbol{\mathcal{A}}$ denote the set of ancestors of node $p$ in the current refueling walk and let $p.\pi$ denote the parent of node $p$ in the current refueling walk. Since $d(s,p) > r$, $p.\boldsymbol{\mathcal{A}}\backslash\{s\}$ contains at least one refueling point. Let $o = p.\pi$ and $n = o.\pi$, then we have $d(p,o) \leq r$, $d(o,n) \leq r$, and $d(p,n) > r$, otherwise $W\backslash\{o\}$ is a shorter refueling walk. Consider

the remaining level of charge of the vehicle as it arrives at junction $J_o$ in the following two cases:

(a) The predecessor of $J_o$ is $o$: the remaining level of charge is $r - l_2(o)$;

(b) The predecessor of $J_o$ is $p$: the remaining level of charge is $r - l_2(p) - \big(l_1(o) - l_1(p)\big)$.

Then, by $d(o,n) \le r$ and $d(p,n) > r$, we have $r - l_2(p) - \big(l_1(o) - l_1(p)\big) < r - l_2(o)$ (see below), which implies that there is no need for the vehicle to make a detour to visit refueling point $p$ after visiting $o$. Thus, $p \notin V(W^1)$. $p' \notin V\big(W^k\big)$ can be proved in a similar way as well.

(Math deduction:

$$\begin{cases} d(o,n) = l_2(n) + \big(l_1(n) - l_1(o)\big) + l_2(o) \le r \\ d(p,n) = l_2(n) + \big(l_1(n) - l_1(p)\big) + l_2(p) > r \end{cases}$$

$$\Rightarrow \big(l_1(n) - l_1(o)\big) + l_2(o) < \big(l_1(n) - l_1(p)\big) + l_2(p)$$

$$\Rightarrow -l_1(o) + l_2(o) < -l_1(p) + l_2(p)$$

$$\Rightarrow -l_2(p) - \big(l_1(o) - l_1(p)\big) < -l_2(o))$$ ■

***Property 3.5*** Let $W = W^1 \uplus W^2 \uplus \cdots \uplus W^k$ be a shortest refueling walk that goes from $s$ to $t$ through the subwalks $W^1$ through $W^k$. $W^1$ contains at most one RP that is on the left-hand side of $s$, and $W^k$ contains at most one RP that is on the right-hand side of $t$.

### 3.3.2. A proposed math program

Define the continuous decision variable $x_k$ for $\wp_k \in \boldsymbol{\mathcal{P}}$ and $k \in \{1, 2, \cdots, m\}$ as the position on the comb span at which $\wp_k$ is to be established. Let $\boldsymbol{x} = \{x_1, x_2, \cdots, x_m\}$ represent these $m$ locations. Then the objective function can be written as

$$\min_{\boldsymbol{x}} Z(\boldsymbol{x}) = \sum_{(s,t)} f(s,t) * \Delta_{(s,t)}(\boldsymbol{x}), \tag{3.4}$$

where $f(s,t)$ denotes the flow volume of one-way trip $(s,t)$. Since for any two one-way trips $(s,t)$ and $(t,s)$ we have $\Delta_{(s,t)} = \Delta_{(t,s)}$, then

$$Z(\boldsymbol{x}) = \sum_{(s,t):s<t} \big(f(s,t) + f(t,s)\big) * \Delta_{(s,t)}, \tag{3.5}$$

Furthermore, by splitting the set of $\{(s,t): s < t\}$ into two disjoint subsets: $\{(s,t): s < t, ne_t^- \geq ne_s^+\}$ and $\{(s,t): s < t, ne_t^- = ne_s^-\}$, we have

$$Z(\boldsymbol{x}) = \sum_{\substack{(s,t):s<t,\\ ne_t^- \geq ne_s^+}} \big(f(s,t) + f(t,s)\big) * \Delta_{(s,t)} + \sum_{\substack{(s,t):s<t,\\ ne_t^- = ne_s^-}} \big(f(s,t) + f(t,s)\big) * \Delta_{(s,t)} \tag{3.6}$$

By the properties of shortest refueling walk, we can derive the refueling detouring distance of one-way trip $(s,t)$ with $ne_t^- \geq ne_s^+$:

$$\begin{aligned}
\Delta_{(s,t)} &= \min\{\delta(W) \mid W \in \boldsymbol{\mathcal{W}}_{(s,t)}\} \\
&= \min\left\{\sum\nolimits_i \delta(W^i) \mid \uplus_i W^i = W \in \boldsymbol{\mathcal{W}}_{(s,t)}\right\} \\
&= \min\left\{\delta(W) \mid W \in \boldsymbol{\mathcal{W}}_{\left(s,\mathcal{p}_{ne_s^+}\right)}\right\} + \cdots + \min\{\delta(W) \mid W \in \boldsymbol{\mathcal{W}}_{(\mathcal{p}_k,\mathcal{p}_{k+1})}\} + \cdots \\
&\qquad + \min\{\delta(W) \mid W \in \boldsymbol{\mathcal{W}}_{\left(\mathcal{p}_{ne_t^-},t\right)}\} \\
&= \Delta_{\left(s,\mathcal{p}_{ne_s^+}\right)} + \cdots + \Delta_{(\mathcal{p}_k,\mathcal{p}_{k+1})} + \cdots + \Delta_{\left(\mathcal{p}_{ne_t^-},t\right)}
\end{aligned} \tag{3.7}$$

where each component of (3.4) is a function of the RP locations $\boldsymbol{x}$. Hence, we can rewrite the first sum in (3.6) as

$$\sum_{\substack{(s,t):s<t,\\ ne_t^- \geq ne_s^+}} \big(f(s,t) + f(t,s)\big) * \left(\Delta_{\left(s,\mathcal{p}_{ne_s^+}\right)} + \cdots + \Delta_{(\mathcal{p}_k,\mathcal{p}_{k+1})} + \cdots + \Delta_{\left(\mathcal{p}_{ne_t^-},t\right)}\right), \tag{3.8}$$

and by way of re-arranging, we get

$$(3.8) = \sum_k \sum_{\substack{(s,t):\, s\le l_1(\wp_k),\\ t\ge l_1(\wp_{k+1})}} \big(f(s,t)+f(t,s)\big) * \Delta_{(\wp_k,\wp_{k+1})}$$

$$+ \sum_k \sum_{\substack{(s,t):\, s\in(l_1(\wp_{k-1}),l_1(\wp_k)],\\ t\ge l_1(\wp_k)}} \big(f(s,t)+f(t,s)\big) * \Delta_{(s,\wp_k)}$$

$$+ \sum_k \sum_{\substack{(s,t):\, t\in[l_1(\wp_k),l_1(\wp_{k+1})),\\ s\le l_1(\wp_k)}} \big(f(s,t)+f(t,s)\big) * \Delta_{(\wp_k,t)} \qquad (3.9)$$

By replacing the first sum in (3.6) with (3.9), now the objective function is

$$Z(\boldsymbol{x}) = (3.9) + \sum_{\substack{(s,t):s<t,\\ ne_t^-=ne_s^-}} \big(f(s,t)+f(t,s)\big) * \Delta_{(s,t)}. \qquad (3.10)$$

Given RP locations $\boldsymbol{x}$, let $V_k$ denote the set of nodes that are on the right-hand side of RP $\wp_k$ and on the left-hand side of $\wp_{k+1}$, that is $V_k = \{v \in V(G): x_k \le l_1(v) \le x_{k+1}\}$. Let $V_k^{\boldsymbol{\mathcal{P}}}$ be the set of RP nodes in $V_k$ (i.e., nodes labelled "RP"). Suppose that we have shut down $\wp_{k+1}$, then $V_k^{\boldsymbol{\mathcal{P}}} \cup \{\wp_k\}$ can be decomposed into a collection of mutually disjoint subsets, $\{V_{k,0}^{\boldsymbol{\mathcal{P}}}, V_{k,1}^{\boldsymbol{\mathcal{P}}}, \cdots, V_{k,q}^{\boldsymbol{\mathcal{P}}}\}$, where for any two distinct nodes $u \in V_{k,i}^{\boldsymbol{\mathcal{P}}}$ and $v \in V_{k,j}^{\boldsymbol{\mathcal{P}}}$, if there is a refueling walk between them then $i \equiv j$. Specifically, we let $V_{k,0}^{\boldsymbol{\mathcal{P}}}$ denote the subset that contains $\wp_k$. For any $l \in \{0, \cdots, q\}$, $V_{k,l}^{\boldsymbol{\mathcal{P}}}$ may be a singleton or contain multiple elements. We call $rep\big(V_{k,l}^{\boldsymbol{\mathcal{P}}}\big)$ the representative of subset $l$, which is defined by $rep\big(V_{k,l}^{\boldsymbol{\mathcal{P}}}\big) = \underset{z\in V_{k,l}^{\boldsymbol{\mathcal{P}}}}{\operatorname{argmax}} \Big(l_1(z) + \big(r - l_2(z)\big)\Big)$. Furthermore, let $N_k = V_k \backslash V_k^{\boldsymbol{\mathcal{P}}}$ represent the set of nodes without a public RP. The representative of $N_k^{\boldsymbol{\mathcal{P}}}$ is defined by $rep(N_k) = \underset{z\in N_k}{\operatorname{argmin}}\big(l_1(z^+)\big)$, where recall that $z^+$ is the farthest point on the comb span that the vehicle is able to reach from $z$. To ensure that all flows can be refueled, every $v \in V_k$ should be able to reach RPs $\wp_k$ and $\wp_{k+1}$. That is, we should have the following constraint:

$$l_1(\wp_{k+1}) \le \min\{l_1(z) + (r - l_2(z)) \mid z \in \{\cup_l Rep(V_{k,l}^{\mathcal{P}})\} \cup Rep(N_k)\}. \quad (3.11)$$

The math program to our comb tree problem is now the following:

| | |
|---|---|
| Minimize | $Z(\boldsymbol{x})$ |
| Subject to | $x_{k+1} \le \min\{l_1(z) + (r - l_2(z)) \mid z \in \{\cup_l rep(V_{k,l}^{\mathcal{P}})\} \cup rep(N_k)\}$ $\quad 1 \le k < \mathrm{m}$ |
| | $\alpha_k \le x_k \le \beta_k \quad 1 \le k \le m$ |

Remarks We are not going to solve detouring-flow comb tree problem by using math programming. However, this proposed math program will be used to prove the existence of a finite dominating set to the problem.

### 3.4. Existence of finite dominating set

In this section, we will show that there exists a finite dominating set (FDS) to the comb tree problem, i.e., a finite set of points where an optimal solution must belong.

#### 3.4.1. Set of breakpoints

For any leaf node $v$, let $v^-(d)$ represent a point on the comb span such that $v^-(d)$ is on the left-hand side of $v$ and at $d$ distance away from $v$, i.e., $l_1(v^-(d)) = l_1(v) - (d - l_2(v))$ and $l_2(v^-(d)) = 0$. Let $v^+(d)$ represent the point on the comb span such that $v^+(d)$ is on the right-hand side of $v_j$ and at $d$ distance away from $v_j$, i.e., $l_1(v^+(d)) = l_1(v) + (d - l_2(v))$ and $l_2(v^+(d)) = 0$. Specifically, we say that $v^-(r)$ and $v^+(r)$ are two extreme none refueling detouring (XNRD) sites for $\wp_{ne_v^-}$ and $\wp_{ne_v^+}$, respectively. By "XNRD" we mean that $v^-(r)$ is the farthest allowable site on the left-hand side of $v$ such that no refueling detouring will occur for a subtrip between $v$ and $\wp_{ne_v^-}$ (i.e., the first subtrip of an one-way trip $(v, u)$ or the last subtrip of $(u, v)$

where $u \preccurlyeq \mathscr{p}_{ne_v^-}$, and that $v^+(r)$ is the farthest allowable site on the right-hand side of $v$ such that no refueling detouring will occur for a subtrip between $v$ and $\mathscr{p}_{ne_v^+}$ (i.e., the first subtrip of an one-way trip $(v,u)$ or the last subtrip of $(u,v)$ where $u \succcurlyeq \mathscr{p}_{ne_v^-}$.

Define $\boldsymbol{\mathcal{B}}$ as the set of breakpoints, and $\boldsymbol{\mathcal{B}}$ is composed of the following four parts:

- $B_1 = \bigcup_{k=1}^{m}\{\alpha_k, \beta_k\}$, the set of endpoints of each localization segment;
- $B_2 = \{J_j : J_j \in V(G)\} \cap \{\bigcup_{k=1}^{m} S_k\}$, the set of internal junction nodes;
- $B_3 = \{v_j^-(r), v_j^+(r) : v_j \in V(G)\} \cap \{\bigcup_{k=1}^{m} S_k\}$, the set of XNRD sites that are either interior points or boundary points of some localization segments;
- $B_4 = \bigcup_{x \in B_1 \cup B_2 \cup B_3}(\{x^-(ir), x^+(ir) : i = 1, 2, \cdots, m\} \cap \{\bigcup_{i=k}^{m} S_k\})$. Supposed that a RP is to be established at some point $x \in B_1 \cup B_2 \cup B_3$, then with $B_4$ we are able to identify a set of $m$ locations on the comb span such that the distance between every two consecutive RPs is not more than the range limit $r$, if any. Unlike the line problem, here it is possible that the distance between the right endpoint of localization segment $S_k$ and the left endpoint of localization segment $S_{k+1}$ is greater than $r$.

By identifying the set of breakpoints $\boldsymbol{\mathcal{B}}$, each localization segment can be further divided into several sub-segments. A segment is called indivisible if it does not contain any breakpoint as its interior point. Consider a localization segment $S_k$, let $n_k$ denote the number of breakpoints that are either interior or boundary points of $S_k$. Then $S_k$ can be decomposed into $n_k - 1$ indivisible sub-segments.

Example 3.4 Consider the same comb tree used in example 3.3. Then the set of breakpoints can be computed as:

$B_1 = \cup_{k=1}^{5}\{\alpha_k, \beta_k\} = \{1.5, 3\} \cup \{5.5, 8.3\} \cup \{10.5, 12.3\} \cup \{14.5, 16.3\} \cup \{18.5, 20.3\}$;

$B_2 = \{J_j : J_j \in V(G)\} \cap \{\cup_{k=1}^{5} S_k\} = \{J_1, J_2, J_5\} = \{2, 5.5, 15\}$;

$$B_3 = \{v_j^{-}(r), v_j^{+}(r) : v_j \in V(G)\} \cap \left\{\bigcup_{k=1}^{5} S_k\right\}$$

$= \{4\} \cup \{\cancel{1}, 3\} \cup \{2.7, 8.3\} \cup \{5.5, 11.5\} \cup \{7.5, 10.5\} \cup \{11.4, 18.6\} \cup \{19, \cancel{23}\} \cup \{18.5\}$

$= \{2.7, 3, 5.5, 7.5, 8.3, 10.5, 11.4, 11.5, 18.5, 18.6, 19\}$;

$$B_4 = \bigcup_{x \in B_1 \cup B_2 \cup B_3} \left( \{x^{-}(ir), x^{+}(ir) : i = 1, \cdots, 5\} \cap \left\{\bigcup_{i=k}^{5} S_k\right\} \right)$$

$= \{5.5, \cancel{9.5}, \cancel{13.5}, \cancel{17.5}\} \cup \{7, 11, 15, 19\} \cup \{\cancel{4.3}, 12.3, 16.3, 20.3\} \cup \{6.5, 2.5, 14.5, 18.5\} \cup$ $\{6, \cancel{10}, \cancel{14}, \cancel{18}\} \cup \{6.7, 10.7, 14.7, 18.7\} \cup \{\cancel{3.5}, 11.5, 15.5, 19.5\} \cup \{7.4, \cancel{3.4}, 15.4, 19.4\} \cup$ $\{14.6, 10.6, 6.6, 2.6\}$.

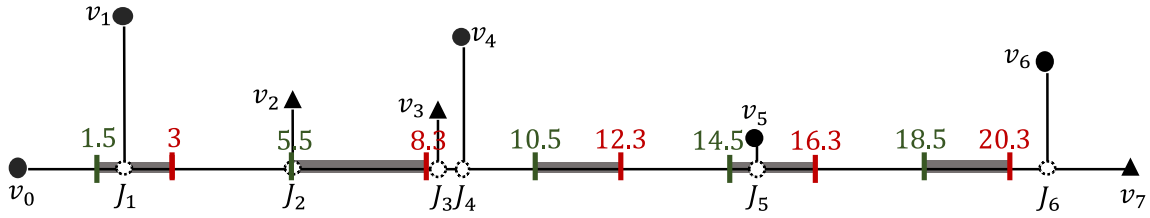



**Figure 3.11** A copy of Figure 3.8

Hence, we can see that

$S_1$ can be divided into 5 sub-segments: [1.5,2], [2,2.5], [2.5,2.6], [2.6,2.7] and [2.7,3];

$S_2$ can be divided into 8 sub-segments: [5.5,6], [6,6.5], [6.5,6.6], [6.6,6.7], [6.7,7], [7, 7.4], [7.4,7.5] and [7.5, 8.3];

$S_3$ can be divided into 6 sub-segments: [10.5,10.6], [10.6,10.7], [10.7,11], [11, 11.4], [11.4, 11.5] and [11.5,12.3];

$S_4$ can be divided into 6 sub-segments: [14.5,14.6], [14.6,14.7], [14.7,15], [15, 15.4], [15.4, 15.5] and [15.5,16.3];

$S_5$ can be divided into 4 sub-segments:[18.5,18.6], [18.6,18.7], [18.7,19], [19, 19.4], [19.4, 19.5] and [19.5,20.3].

Now we claim that there exists an FDS to the comb tree problem.

***Theorem 3.2*** $\boldsymbol{\mathcal{B}}$ is an FDS to the comb tree problem.

### 3.4.2. Restricted problem

We may prove the existence of an FDS by considering a set of $\prod_{i=1}^{m}(n_k - 1)$ restricted problems by requiring each RP to be established within an indivisible sub-segment of its original localization segment. Then an optimal solution of at least one of these restricted problems is optimal to the original problem. It suffices to show that $\boldsymbol{\mathcal{B}}$ is an FDS for each restricted problem. A restricted problem is formulated as:

Minimize $Z(\boldsymbol{x})$

Subject to $x_{k+1} \le \min\{l_1(z) + (r - l_2(z)) \mid z \in \{\bigcup_l rep(V_{k,l}^{\mathcal{P}})\} \cup rep(N_k)\}\ 1 \le k < m$

$\alpha_k^{\boldsymbol{h}} \le x_k \le \beta_k^{\boldsymbol{h}}$

$1 \le k < m$

where $\boldsymbol{h} = (h_1, h_2, \cdots, h_m)$ is a particular combination of indivisible sub-segments and $h_k \in \{1, 2, \cdots, n_k - 1\}$, and where $[\alpha_k^{\boldsymbol{h}}, \beta_k^{\boldsymbol{h}}]$ denotes the ${h_k}^{th}$ indivisible sub-segment between two consecutive breakpoints $\alpha_k^{\boldsymbol{h}}$ (the ${h_k}^{th}$ breakpoint in $S_k$) and $\beta_k^{\boldsymbol{h}}$ (the $(h_k + 1)^{th}$ breakpoint in $S_k$).

Since $[\alpha_k^{\boldsymbol{h}}, \beta_k^{\boldsymbol{h}}]$ is an indivisible sub-segment, then by the way we defined the set of breakpoints $\boldsymbol{\mathcal{B}}$, we know that the sub-segment $[\alpha_k^{\boldsymbol{h}} + r, \beta_k^{\boldsymbol{h}} + r]$ is indivisible, if

$\alpha_k^{\boldsymbol{h}} + r, \beta_k^{\boldsymbol{h}} + r \in \boldsymbol{\mathcal{B}}$, and that if $\left[\alpha_k^{\boldsymbol{h}} + r, \beta_k^{\boldsymbol{h}} + r\right]$ is indivisible, then for any $z \in V_k$ we have either $z^+(r) \leq \alpha_k^{\boldsymbol{h}} + r$ or $z^+(r) \geq \beta_k^{\boldsymbol{h}} + r$. Let

$$f_k^0(\boldsymbol{x}) = \max\left\{l_1(z) + \left(r - l_2(z)\right) \mid z \in V_{k,0}^{\boldsymbol{\mathcal{P}}} \backslash \{\boldsymbol{\mathcal{p}}_k\}\right\}, \tag{3.12}$$

$$f_k^1(\boldsymbol{x}) = \min\left\{l_1(z) + \left(r - l_2(z)\right) \mid z \in \left\{\bigcup_{l \neq 0} rep\left(V_{k,l}^{\boldsymbol{\mathcal{P}}}\right)\right\} \cup rep(N_k)\right\} \tag{3.13}$$

Given RP locations $\boldsymbol{x}$. Let $ub_{k+1}(\boldsymbol{x})$ denote the upper bound of $x_{k+1}$. Then,

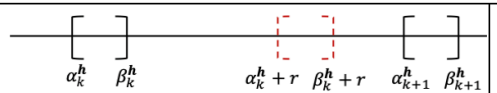
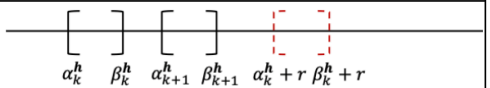
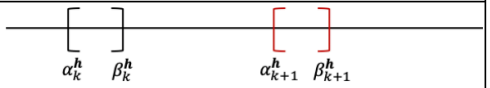
$$\begin{aligned} ub_{k+1}(\boldsymbol{x}) &= \min\left\{\beta_{k+1}^{\boldsymbol{h}}, \min\left\{l_1(z) + \left(r - l_2(z)\right) \mid z \in \left\{\bigcup_l rep\left(V_{k,l}^{\boldsymbol{\mathcal{P}}}\right)\right\} \cup rep(N_k)\right\}\right\} \\ &= \min\left\{\beta_{k+1}^{\boldsymbol{h}}, \min\left\{\max\{f_k^0(\boldsymbol{x}), x_k + r\}, f_k^1(\boldsymbol{x})\right\}\right\} \\ &= \begin{cases} \min\left\{\beta_{k+1}^{\boldsymbol{h}}, f_k^1(\boldsymbol{x})\right\}, & \text{if } f_k^1(\boldsymbol{x}) \leq \alpha_k^{\boldsymbol{h}} + r \\ \min\left\{\beta_{k+1}^{\boldsymbol{h}}, \min\left\{f_k^0(\boldsymbol{x}), f_k^1(\boldsymbol{x})\right\}\right\}, & \text{if } f_k^1(\boldsymbol{x}) \geq \beta_k^{\boldsymbol{h}} + r, f_k^0(\boldsymbol{x}) \geq \beta_k^{\boldsymbol{h}} + r \\ \min\left\{\beta_{k+1}^{\boldsymbol{h}}, x_k + r\right\}, & \text{if } f_k^1(\boldsymbol{x}) \geq \beta_k^{\boldsymbol{h}} + r, f_k^0(\boldsymbol{x}) \leq \alpha_k^{\boldsymbol{h}} + r \end{cases} \end{aligned} \tag{3.14}$$

Consider any $x_k \in \left[\alpha_k^{\boldsymbol{h}}, \beta_k^{\boldsymbol{h}}\right]$, let $\mathbb{X}_{k+1}$ represent the set of allowable sites of $\boldsymbol{\mathcal{p}}_{k+1}$ in the solution space with $l_1(\boldsymbol{\mathcal{p}}_k) = x_k$, that is, site positions that are within the segment between $\alpha_k^{\boldsymbol{h}}$ and $ub_{k+1}(\boldsymbol{x})$. We claim that $\mathbb{X}_{k+1}$ is one of the four following types:

- $\mathbb{X}_{k+1} = \emptyset$
- $\mathbb{X}_{k+1} = \left\{\alpha_{k+1}^{\boldsymbol{h}}\right\}$
- $\mathbb{X}_{k+1} = \left[\alpha_{k+1}^{\boldsymbol{h}}, \beta_{k+1}^{\boldsymbol{h}}\right]$
- $\mathbb{X}_{k+1} = \left[\alpha_{k+1}^{\boldsymbol{h}}, x_k + r\right]$

depending on the values of $f_k^0(\boldsymbol{x})$ and $f_k^1(\boldsymbol{x})$, and the relative position of segments $\left[\alpha_{k+1}^{\boldsymbol{h}}, \beta_{k+1}^{\boldsymbol{h}}\right]$ and $\left[\alpha_k^{\boldsymbol{h}} + r, \beta_k^{\boldsymbol{h}} + r\right]$. The detail is depicted in Table 3.1.

**Table 3.1** A table of $\mathbb{X}_{k+1}$

| | | | $f_k^1 \le \alpha_k^h + r$ | $f_k^1 \ge \beta_k^h + r$ | |
|---|---|---|---|---|---|
| | | | | $f_k^0 \ge \beta_k^h + r$ | $f_k^0 \le \alpha_k^h + r$ |
| $ub_{k+1}$ | | | $\min\{\beta_{k+1}^h, f_k^1\}$ | $\min\{\beta_{k+1}^h, \min\{f_k^0, f_k^1\}\}$ | $\min\{\beta_{k+1}^h, x_k + r\}$ |
| a) |   | $\mathbb{X}_{k+1}$ | $\emptyset$ | $\begin{cases} \emptyset, & \min\{f_k^0, f_k^1\} < \alpha_{k+1}^h \\ \{\alpha_{k+1}^h\}, & \min\{f_k^0, f_k^1\} = \alpha_{k+1}^h \\ [\alpha_{k+1}^h, \beta_{k+1}^h], & \min\{f_k^0, f_k^1\} \ge \beta_{k+1}^h \end{cases}$ | $\emptyset$ |
| b) |   | | $\begin{cases} \emptyset, & f_k^1 < \alpha_{k+1}^h \\ \{\alpha_{k+1}^h\}, & f_k^1 = \alpha_{k+1}^h \\ [\alpha_{k+1}^h, \beta_{k+1}^h], & f_k^1 \ge \beta_{k+1}^h \end{cases}$ | $[\alpha_{k+1}^h, \beta_{k+1}^h]$ | $[\alpha_{k+1}^h, \beta_{k+1}^h]$ |
| c) |   | | $\begin{cases} \emptyset, & f_k^1 < \alpha_{k+1}^h \\ \{\alpha_{k+1}^h\}, & f_k^1 = \alpha_{k+1}^h \end{cases}$ | $[\alpha_{k+1}^h, \beta_{k+1}^h]$ | $[\alpha_{k+1}^h, x_k + r]$ |

***Proposition 3.7*** If $x_k < \beta_k^{\boldsymbol{h}}$, then given any $\varepsilon > 0$, a $\delta > 0$ can be found such that for every $\tilde{x}_k \in [\alpha_k^{\boldsymbol{h}}, \beta_k^{\boldsymbol{h}})$ and within the neighborhood of $x_k$ of radius $\delta$, $-\varepsilon < ub_{k+1}(\tilde{\boldsymbol{x}}) - ub_{k+1}(\boldsymbol{x}) < \varepsilon$.

*Proof of Proposition 3.7* Since $[\alpha_k^{\boldsymbol{h}}, \beta_k^{\boldsymbol{h}}]$ is an indivisible sub-segment, we have

- $V_k(\boldsymbol{x}) = V_k(\tilde{\boldsymbol{x}})$,
- for any $z \in V_k(\boldsymbol{x})$, $z^-(r) > x_k$ if and only if $z^-(r) > \tilde{x}_k$. $(*)$

Then we claim that $z$ is not able to reach $\mathcal{p}_k$ in $G^{\boldsymbol{\mathcal{P}}}(\boldsymbol{x})$ if and only if $z$ is not able to reach $\mathcal{p}_k$ in $G^{\boldsymbol{\mathcal{P}}}(\tilde{\boldsymbol{x}})$:

$z \in V_k(\boldsymbol{x})$ is not able to reach $\mathcal{p}_k$ in $G^{\boldsymbol{\mathcal{P}}}(\boldsymbol{x})$

$\Leftrightarrow$ In $G^{\boldsymbol{\mathcal{P}}}(\boldsymbol{x})$: $d(z, \mathcal{p}_k) > r$, and $z$ is **not** able to reach any RP node that is within $r$ distance to $\mathcal{p}_k$ (by definition of reachability)

$\Leftrightarrow$ In $G^{\boldsymbol{\mathcal{P}}}(\tilde{\boldsymbol{x}})$: $d(z, \mathcal{p}_k) > r$, and $z$ is **not** able to reach any RP node that is within $r$ distance to $\mathcal{p}_k$ (by $(*)$)

$\Leftrightarrow$ $z$ is not able to reach $\mathcal{p}_k$ in $G^{\boldsymbol{\mathcal{P}}}(\tilde{\boldsymbol{x}})$.

As a result, for any $l$, $V_{k,l}^{\boldsymbol{\mathcal{P}}}(\boldsymbol{x}) = V_{k,l}^{\boldsymbol{\mathcal{P}}}(\tilde{\boldsymbol{x}})$, and $N_k(\boldsymbol{x}) = N_k(\tilde{\boldsymbol{x}})$. By definition of $f_k^0(\boldsymbol{x})$ and $f_k^1(\boldsymbol{x})$, we have $f_k^0(\boldsymbol{x}) = f_k^0(\tilde{\boldsymbol{x}})$ and $f_k^1(\boldsymbol{x}) = f_k^1(\tilde{\boldsymbol{x}})$. Recall that $ub_{k+1}(\boldsymbol{x}) = \min\{\beta_{k+1}^{\boldsymbol{h}}, \min\{\max\{f_k^0(\tilde{\boldsymbol{x}}), x_k + r\}, f_k^1(\tilde{\boldsymbol{x}})\}\}$, then we have either $ub_{k+1}(\boldsymbol{x}) - ub_{k+1}(\tilde{\boldsymbol{x}}) = 0$, or $ub_{k+1}(\boldsymbol{x}) - ub_{k+1}(\tilde{\boldsymbol{x}}) = x_k - \tilde{x}_k$. ■

While if $x_k = \beta_k^{\boldsymbol{h}}$, such a radius $\delta > 0$ depicted in proposition 3.7 may not exist.

***Proposition 3.8*** If $x_k = \beta_k^{\boldsymbol{h}}$, for any $\tilde{x}_k < x_k$, it is possible to have

$$ub_{k+1}(\boldsymbol{x}) = \beta_{k+1}^{\boldsymbol{h}} \text{ and } ub_{k+1}(\tilde{\boldsymbol{x}}) = \alpha_{k+1}^{\boldsymbol{h}}, \qquad (**)$$

*Proof of Proposition 3.8* To show $(**)$, let us consider the following two cases:

- Suppose that $f_k^0(\tilde{\boldsymbol{x}}) = \alpha_{k+1}^{\boldsymbol{h}}$, $f_k^1(\tilde{\boldsymbol{x}}) \geq \beta_{k+1}^{\boldsymbol{h}}$, and $\alpha_{k+1}^{\boldsymbol{h}} \geq \beta_k^{\boldsymbol{h}} + r$. Then $ub_{k+1}(\tilde{\boldsymbol{x}}) = \alpha_{k+1}^{\boldsymbol{h}}$. If a RP node $\tilde{z}$ can be found such that $l_1(\tilde{z}) - \big(r - l_2(\tilde{z})\big) = \beta_k^{\boldsymbol{h}}$ and $\tilde{z} \in V_{k,l}^{\boldsymbol{\mathcal{P}}}(\tilde{\boldsymbol{x}})$ where $l \neq 0$. Then in $G^{\boldsymbol{\mathcal{P}}}(\boldsymbol{x})$, where RP $\wp_k$ is established at position $\beta_k^{\boldsymbol{h}}$, we have $\tilde{z} \in V_{k,0}^{\boldsymbol{\mathcal{P}}}(\boldsymbol{x})$, and hence $V_{k,l}^{\boldsymbol{\mathcal{P}}}(\tilde{\boldsymbol{x}}) \subseteq V_{k,0}^{\boldsymbol{\mathcal{P}}}(\boldsymbol{x})$. Then, $f_k^0(\boldsymbol{x}) \geq f_k^1(\tilde{\boldsymbol{x}}) \geq \beta_{k+1}^{\boldsymbol{h}}$. Therefore, $ub_{k+1}(\boldsymbol{x}) = \beta_{k+1}^{\boldsymbol{h}}$.
- Conversely, suppose that $f_k^0(\tilde{\boldsymbol{x}}) \geq \beta_{k+1}^{\boldsymbol{h}}$ and $f_k^1(\tilde{\boldsymbol{x}}) = \alpha_{k+1}^{\boldsymbol{h}}$. Then $ub_{k+1}(\tilde{\boldsymbol{x}}) = \alpha_{k+1}^{\boldsymbol{h}}$. If a node set $\tilde{Z} \subseteq V_k(\tilde{\boldsymbol{x}})$ can be found, where:

  a) $\tilde{Z} \cap V_{k,0}^{\boldsymbol{\mathcal{P}}}(\tilde{\boldsymbol{x}}) = \emptyset$,

  b) for every $\tilde{z} \in \tilde{Z}$, $l_1(z) - \big(r - l_2(z)\big) = \beta_k^{\boldsymbol{h}}$,

  c) in $G^{\boldsymbol{\mathcal{P}}(\tilde{\boldsymbol{x}})}$, for every $z \in V_k(\tilde{\boldsymbol{x}}) \backslash \tilde{Z} \backslash \{\cup_{l \notin \tilde{L}} V_{k,l}^{\boldsymbol{\mathcal{P}}}(\tilde{\boldsymbol{x}})\}$ (where \ denotes set minus), $z$ is able to reach $\beta_{k+1}^{\boldsymbol{h}}$, where $\tilde{L} = \cup_{\tilde{z} \in \tilde{Z}} \{l \mid \tilde{z} \in V_{k,l}^{\boldsymbol{\mathcal{P}}}(\tilde{\boldsymbol{x}})\}$.

  Let $f_k^{1'}(\tilde{\boldsymbol{x}}) := \min\{l_1(z) + \big(r - l_2(z)\big) \mid z \in \Big\{\cup_{l \notin \{0,\tilde{L}\}} Rep\left(V_{k,l}^{\boldsymbol{\mathcal{P}}}(\tilde{\boldsymbol{x}})\right)\Big\} \cup \big(Rep\big(N_k(\tilde{\boldsymbol{x}})\big) \backslash \tilde{Z}\big)\}$, we have $f_k^{1'}(\tilde{\boldsymbol{x}}) \geq \beta_{k+1}^{\boldsymbol{h}}$. Then in $G^{\boldsymbol{\mathcal{P}}(x)}$, where RP $\wp_k$ is located at $\beta_k^{\boldsymbol{h}}$, we will have $\cup_{l:l \neq 0} V_{k,l}^{\boldsymbol{\mathcal{P}}}(\boldsymbol{x}) \subseteq \cup_{l:l \notin \{0,\tilde{L}\}} V_{k,l}^{\boldsymbol{\mathcal{P}}}(\tilde{\boldsymbol{x}})$, and $N_k(\boldsymbol{x}) \subseteq N_k(\tilde{\boldsymbol{x}}) \backslash \tilde{Z}$, implying that $f_k^1(\boldsymbol{x}) \geq f_k^{1'}(\tilde{\boldsymbol{x}}) \geq \beta_{k+1}^{\boldsymbol{h}}$. Therefore, $ub_{k+1}(\boldsymbol{x}) = \beta_{k+1}^{\boldsymbol{h}}$.

In either case, (**) would happen, which may lead to a non-convex solution space. ■

As illustrated in Figure 3.12, that every interior point on the red line segment is not within the solution space (the black line segment). However, we can overcome this issue by further restricting $\mathbb{X}_k \times \mathbb{X}_{k+1}$ on $\big[\alpha_k^{\boldsymbol{h}}, \beta_k^{\boldsymbol{h}}\big] \times \big[\alpha_{k+1}^{\boldsymbol{h}}, \alpha_{k+1}^{\boldsymbol{h}}\big]$ and $\big[\beta_k^{\boldsymbol{h}}, \beta_k^{\boldsymbol{h}}\big] \times \big[\alpha_{k+1}^{\boldsymbol{h}}, \beta_k^{\boldsymbol{h}}\big]$. Then we have $ub_{k+1}(\tilde{\boldsymbol{x}}) \equiv ub_{k+1}(\boldsymbol{x})$.

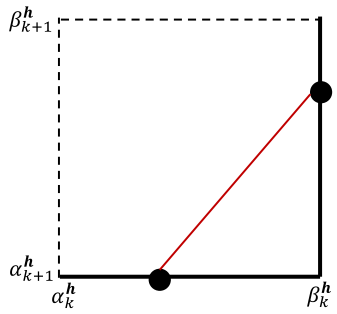



**Figure 3.12** An illustration of non-convex solution space resulting from $ub_{k+1}(\boldsymbol{x}) = \beta_{k+1}^{\boldsymbol{h}}$ and $ub_{k+1}(\tilde{\boldsymbol{x}}) = \alpha_{k+1}^{\boldsymbol{h}}$, where $\tilde{x}_k < x_k = \beta_k^{\boldsymbol{h}}$

Convexity --- feasible region

Let $\boldsymbol{C}$ represent the solution space. Let $\boldsymbol{x}, \boldsymbol{x}' \in \boldsymbol{C}$ and $0 < \theta < 1$. To prove $\boldsymbol{C}$ is a convex region, we shall show that $\boldsymbol{x}'' = (1-\theta)\boldsymbol{x} + \theta\boldsymbol{x}' \in \boldsymbol{C}$. Since we have

- $x''_{k+1} = (1-\theta)x_{k+1} + \theta x'_{k+1} \geq \alpha_{k+1}^{\boldsymbol{h}}$,
- $x''_{k+1} = (1-\theta)x_{k+1} + \theta x'_{k+1} \leq (1-\theta)ub_{k+1}(\boldsymbol{x}) + \theta ub_{k+1}(\boldsymbol{x}')$,

then if we can show that $(1-\theta)ub_{k+1}(\boldsymbol{x}) + \theta ub_{k+1}(\boldsymbol{x}') \leq ub_{k+1}(\boldsymbol{x}'')$, the convexity is proved. By proposition 3.7 and by way of further restricting, we have

- $ub_{k+1}(\boldsymbol{x}) = ub_{k+1}(\boldsymbol{x}') = ub_{k+1}(\boldsymbol{x}'')$, or
- $ub_{k+1}(\boldsymbol{x}) = x_k + r, ub_{k+1}(\boldsymbol{x}') = x'_k + r, ub_{k+1}(\boldsymbol{x}'') = x''_k + r$.

Thus, $(1-\theta)ub_{k+1}(\boldsymbol{x}) + \theta ub_{k+1}(\boldsymbol{x}') = ub_{k+1}(\boldsymbol{x}'')$.

Concavity --- objective function

The objective function of a restricted problem can be rewritten as

$$Z^{\boldsymbol{h}}(\boldsymbol{x}) = \Bigg\{ \sum_{k} \sum_{\substack{(s,t):\, s \le \alpha_k^{\boldsymbol{h}}, \\ t \ge \beta_{k+1}^{\boldsymbol{h}}}} \big(f_{(s,t)} + f_{(t,s)}\big) * \Delta^{\boldsymbol{h}}_{(\mathscr{p}_k \mathscr{p}_{k+1})}(\boldsymbol{x}) + \sum_{k} \sum_{\substack{(s,t):\, s \in [\beta_{k-1}^{\boldsymbol{h}}, \alpha_k^{\boldsymbol{h}}], \\ t \ge \beta_k^{\boldsymbol{h}}}} \big(f_{(s,t)} + f_{(t,s)}\big) * \Delta^{\boldsymbol{h}}_{(s,\mathscr{p}_k)}(\boldsymbol{x}) + \sum_{k} \sum_{\substack{(s,t):\, t \in [\beta_k^{\boldsymbol{h}}, \alpha_{k+1}^{\boldsymbol{h}}], \\ s \le \alpha_k^{\boldsymbol{h}}}} \big(f_{(s,t)} + f_{(t,s)}\big) * \Delta^{\boldsymbol{h}}_{(\mathscr{p}_k,t)}(\boldsymbol{x}) \Bigg\} + \sum_{(s,t):\beta_k^{\boldsymbol{h}} \le s < t \le \alpha_{k+1}^{\boldsymbol{h}}} \big(f_{(s,t)} + f_{(t,s)}\big) * \Delta^{\boldsymbol{h}}_{(s,t)}(\boldsymbol{x}). \tag{3.15}$$

Let $\boldsymbol{\mathcal{W}}_k^{\boldsymbol{h}}(\boldsymbol{x})$ denote the set of refueling walks between $\mathscr{p}_k$ and $\mathscr{p}_{k+1}$, where $l_1(\mathscr{p}_k) = x_k$, and let $\Delta_k^{\boldsymbol{h}}(\boldsymbol{x})$ denote the detouring distance of the shortest refueling walk between $\mathscr{p}_k$ and $\mathscr{p}_{k+1}$. Then

$$\Delta_k^{\boldsymbol{h}}(\boldsymbol{x}) = \min_{W \in \boldsymbol{\mathcal{W}}_k^{\boldsymbol{h}}(\boldsymbol{x})} \{\delta(W)\} = \min_{W \in \boldsymbol{\mathcal{W}}_k^{\boldsymbol{h}}(\boldsymbol{x})} \{\textstyle\sum_{z \in V(W)} 2 \times l_2(z)\}. \tag{3.16}$$

Specifically, we let $\boldsymbol{\mathcal{W}}_k^{\boldsymbol{h}}(\boldsymbol{x}) = \begin{cases} \boldsymbol{\mathcal{W}}_{k,0,0}^{\boldsymbol{h}}, & \text{if } x_k = \alpha_k^{\boldsymbol{h}}, x_{k+1} = \alpha_{k+1}^{\boldsymbol{h}} \\ \boldsymbol{\mathcal{W}}_{k,0,1}^{\boldsymbol{h}}, & \text{if } x_k = \alpha_k^{\boldsymbol{h}}, x_{k+1} = \beta_{k+1}^{\boldsymbol{h}} \\ \boldsymbol{\mathcal{W}}_{k,1,0}^{\boldsymbol{h}}, & \text{if } x_k = \beta_k^{\boldsymbol{h}}, x_{k+1} = \alpha_{k+1}^{\boldsymbol{h}} \\ \boldsymbol{\mathcal{W}}_{k,1,1}^{\boldsymbol{h}}, & \text{if } x_k = \beta_k^{\boldsymbol{h}}, x_{k+1} = \beta_{k+1}^{\boldsymbol{h}} \end{cases}$, and let $\Delta_{k,i,j}^{\boldsymbol{h}} = \min_{W \in \boldsymbol{\mathcal{W}}_{k,i,j}^{\boldsymbol{h}}} \{\sum_{z \in V(W)} 2 \times l_2(z)\}$, where $i, j = \{0, 1\}$.

Given $\boldsymbol{x}$, let $R^+(\mathscr{p}_k)|_{\boldsymbol{x}}$ and $R^-(\mathscr{p}_k)|_{\boldsymbol{x}}$ represent the set of nodes that are reachable to $\mathscr{p}_k$ and on the right-hand side and left-hand side of $\mathscr{p}_k$, respectively. By the indivisibility of each sub-segment, we have

$$R^{+}(\mathcal{p}_k)|_{\boldsymbol{x}} = \{z \mid (l_1(z) - x_k) + l_2(z) \le r\}$$

$$= \{z \mid l_1(z) + l_2(z) - r \le x_k\}$$

$$= \left\{z \mid l_1(z) + l_2(z) - r \le \alpha_k^{\boldsymbol{h}}\right\} \cup \{z \mid a_k < l_1(z) + l_2(z) - r \le x_k\}$$

$$= \begin{cases} \left\{z \mid l_1(z) + l_2(z) - r \le \alpha_k^{\boldsymbol{h}}\right\}, & \text{if } x_k < \beta_k^{\boldsymbol{h}} \\ \left\{z \mid l_1(z) + l_2(z) - r \le \alpha_k^{\boldsymbol{h}}\right\} \cup \left\{z \mid l_1(z) + l_2(z) - r = \beta_k^{\boldsymbol{h}}\right\}, & \text{if } x_k = \beta_k^{\boldsymbol{h}} \end{cases} \tag{3.17}$$

and

$$R^{-}(\mathcal{p}_k)|_{\boldsymbol{x}} = \left\{z \mid \left(x_i - l_1(z)\right) + l_2(z) \le r\right\}$$

$$= \{z \mid -l_1(z) + l_2(z) + r \ge x_k\}$$

$$= \left\{z \mid -l_1(z) + l_2(z) + r \ge \beta_k^{\boldsymbol{h}}\right\} \cup \left\{z \mid x_i \le -l_1(z) + l_2(z) + r < \beta_k^{\boldsymbol{h}}\right\}$$

$$= \begin{cases} \left\{z \mid -l_1(z) + l_2(z) + r \ge \beta_k^{\boldsymbol{h}}\right\}, & \text{if } x_k > \alpha_k^{\boldsymbol{h}} \\ \left\{z \mid -l_1(z) + l_2(z) + r \ge \beta_k^{\boldsymbol{h}}\right\} \cup \left\{z \mid -l_1(z) + l_2(z) + r = \alpha_k^{\boldsymbol{h}}\right\}, & \text{if } x_k = \alpha_k^{\boldsymbol{h}} \end{cases} \tag{3.18}$$

Then we can claim that for any $(x_k, x_{k+1}) \in [\alpha_k^{\boldsymbol{h}}, \beta_k^{\boldsymbol{h}}) \times (\alpha_{k+1}^{\boldsymbol{h}}, \beta_{k+1}^{\boldsymbol{h}}]$, we have $\boldsymbol{\mathcal{W}}_k^{\boldsymbol{h}}(\boldsymbol{x}) = \boldsymbol{\mathcal{W}}_{k,0,1}^{\boldsymbol{h}}$, $\boldsymbol{\mathcal{W}}_{k,0,1}^{\boldsymbol{h}} \subseteq \boldsymbol{\mathcal{W}}_{k,0,0}^{\boldsymbol{h}}$ and $\boldsymbol{\mathcal{W}}_{k,1,1}^{\boldsymbol{h}} \subseteq \boldsymbol{\mathcal{W}}_{k,1,0}^{\boldsymbol{h}}$, implying that $\Delta_k^{\boldsymbol{h}}(\boldsymbol{x}) = \Delta_{k,0,1}^{\boldsymbol{h}}$, $\Delta_{k,0,1}^{\boldsymbol{h}} \ge \Delta_{k,0,0}^{\boldsymbol{h}}$ and $\Delta_{k,1,1}^{\boldsymbol{h}} \ge \Delta_{k,1,0}^{\boldsymbol{h}}$. Hence, we should safely be able to claim that the function $\Delta_k^{\boldsymbol{h}}(\boldsymbol{x})$ is concave.

Let $s$ denote the origin node of some one-way trip. Let $\boldsymbol{\mathcal{W}}_{(s,+)}^{\boldsymbol{h}}(\boldsymbol{x})$ denote the set of refueling walks from $s$ to its closest RP $\mathcal{p}_{ne_s^+}$ that is on the right-hand side of $s$. Note that $\boldsymbol{\mathcal{W}}_{(s,+)}^{\boldsymbol{h}}(\boldsymbol{x})$ can be decomposed into two disjoint subsets: $\boldsymbol{\mathcal{W}}_{(s,+,in)}^{\boldsymbol{h}}(\boldsymbol{x})$ and $\boldsymbol{\mathcal{W}}_{(s,+,ex)}^{\boldsymbol{h}}(\boldsymbol{x})$, where for every $W \in \boldsymbol{\mathcal{W}}_{(s,+,in)}^{\boldsymbol{h}}(\boldsymbol{x})$, $W$ goes through $\mathcal{p}_{ne_s^-}$, and for every $W \in \boldsymbol{\mathcal{W}}_{(s,+,ex)}^{\boldsymbol{h}}(\boldsymbol{x})$, $W$ does not go through $\mathcal{p}_{ne_s^-}$. By shortest refueling walk properties 3.4 and 3.5, we have:

$$\Delta^{\boldsymbol{h}}_{(s,+)}(\boldsymbol{x}) = \begin{cases} \min\{\Delta^{\boldsymbol{h}}_{(s,+,ex)}(\boldsymbol{x}), \Delta^{\boldsymbol{h}}_{(s,+,in)}(\boldsymbol{x})\}, & d(s, \mathscr{p}_{ne_s^-}) \le r \\ \Delta^{\boldsymbol{h}}_{(s,+,ex)}(\boldsymbol{x}), & d(s, \mathscr{p}_{ne_s^-}) > r \end{cases}$$

$$= \begin{cases} \min\{\Delta^{\boldsymbol{h}}_{(s,+,ex)}(\boldsymbol{x}), 2(l_1(s) - x_{ne_s^-}) + \Delta^{\boldsymbol{h}}_{ne_s^-}(\boldsymbol{x})\}, & d(s, \mathscr{p}_{ne_s^-}) \le r \\ \Delta^{\boldsymbol{h}}_{(s,+,ex)}(\boldsymbol{x}), & d(s, \mathscr{p}_{ne_s^-}) > r \end{cases} \tag{3.19}$$

Without loss of generality, we suppose that $ne_s^+ = k+1$. Then for any $x_{k+1} \in (\alpha^{\boldsymbol{h}}_{k+1}, \beta^{\boldsymbol{h}}_{k+1}]$, we have

$$\Delta^{\boldsymbol{h}}_{(s,+,ex)}(\boldsymbol{x})|_{x_{k+1}=\alpha^{\boldsymbol{h}}_{k+1}} \le \Delta^{\boldsymbol{h}}_{(s,+,ex)}(\boldsymbol{x}) = \Delta^{\boldsymbol{h}}_{(s,+,ex)}(\boldsymbol{x})|_{x_{k+1}=\beta^{\boldsymbol{h}}_{k+1}}.$$

Therefore, the function $\Delta^{\boldsymbol{h}}_{(s,+,ex)}(\boldsymbol{x})$ is concave. Furthermore, we are able to claim that $\Delta^{\boldsymbol{h}}_{(s,+)}(\boldsymbol{x})$ is concave.

When $ne_s^+ = ne_t^+$, $\boldsymbol{\mathcal{W}}^{\boldsymbol{h}}_{(s,t)}(\boldsymbol{x})$ can be decomposed into four disjoint subsets: $\boldsymbol{\mathcal{W}}^{\boldsymbol{h}}_{(s,t,ex,ex)}(\boldsymbol{x})$, $\boldsymbol{\mathcal{W}}^{\boldsymbol{h}}_{(s,t,in,ex)}(\boldsymbol{x})$, $\boldsymbol{\mathcal{W}}^{\boldsymbol{h}}_{(s,t,ex,in)}(\boldsymbol{x})$ and $\boldsymbol{\mathcal{W}}^{\boldsymbol{h}}_{(s,t,in,in)}(\boldsymbol{x})$. Likewise, by shortest refueling walk properties 3.4 and 3.5, we have

$$\Delta^{\boldsymbol{h}}_{(s,t)}(\boldsymbol{x})$$

$$= \begin{cases} \min\{\Delta^{\boldsymbol{h}}_{(s,t,ex,ex)}(\boldsymbol{x}), \Delta^{\boldsymbol{h}}_{(s,t,in,ex)}(\boldsymbol{x}), \Delta^{\boldsymbol{h}}_{(s,t,ex,in)}(\boldsymbol{x}), \Delta^{\boldsymbol{h}}_{(s,t,in,in)}(\boldsymbol{x})\}, & d(s, \mathscr{p}_{ne_s^-}) \le r, d(t, \mathscr{p}_{ne_s^+}) \le r \\ \min\{\Delta^{\boldsymbol{h}}_{(s,t,ex,ex)}(\boldsymbol{x}), \Delta^{\boldsymbol{h}}_{(s,t,ex,in)}(\boldsymbol{x})\}, & d(s, \mathscr{p}_{ne_s^-}) > r, d(t, \mathscr{p}_{ne_s^+}) \le r \\ \min\{\Delta^{\boldsymbol{h}}_{(s,t,ex,ex)}(\boldsymbol{x}), \Delta^{\boldsymbol{h}}_{(s,t,in,ex)}(\boldsymbol{x})\}, & d(s, \mathscr{p}_{ne_s^-}) \le r, d(t, \mathscr{p}_{ne_s^+}) > r' \\ \Delta^{\boldsymbol{h}}_{(s,t,ex,ex)}(\boldsymbol{x}), & d(s, \mathscr{p}_{ne_s^-}) > r, d(t, \mathscr{p}_{ne_s^+}) > r \end{cases}$$

(3.20)

where $\Delta^{\boldsymbol{h}}_{(s,t,in,ex)}(\boldsymbol{x}) = 2(l_1(s) - x_{ne_s^-}) + \Delta^{\boldsymbol{h}}_{(t,-,ex)}(\boldsymbol{x})$, (3.21)

$$\Delta^{\boldsymbol{h}}_{(s,t,ex,in)}(\boldsymbol{x}) = \Delta^{\boldsymbol{h}}_{(s,+,ex)}(\boldsymbol{x}) + 2\left(x_{ne_s^+} - l_1(t)\right), \tag{3.22}$$

$$\Delta^{\boldsymbol{h}}_{(s,t,in,in)}(\boldsymbol{x}) = 2(l_1(s) - x_{ne_s^-}) + \Delta^{\boldsymbol{h}}_{ne_s^-}(\boldsymbol{x}) + 2\left(x_{ne_s^+} - l_1(t)\right). \tag{3.23}$$

To prove the concavity of $\Delta^{\boldsymbol{h}}_{(s,t)}(\boldsymbol{x})$, it suffices to show that $\Delta^{\boldsymbol{h}}_{(s,t,ex,ex)}(\boldsymbol{x})$ is concave, which is easy, since $\Delta^{\boldsymbol{h}}_{(s,t,ex,ex)}(\boldsymbol{x})$ would be a constant for any $(x_k, x_{k+1}) \in [\alpha^{\boldsymbol{h}}_k, \beta^{\boldsymbol{h}}_k] \times [\alpha^{\boldsymbol{h}}_{k+1}, \beta^{\boldsymbol{h}}_{k+1}]$.

Therefore, at least one of the extreme points will be the optimal solution for each restricted problem. Let $\mathcal{E}(\boldsymbol{C})$ denote the set of extreme points of the solution space $\boldsymbol{C}$. Then follow the same argument by which we proved theorem 2.4, we are able to prove that $\mathcal{E}(\boldsymbol{C}) \subseteq \{(b_1, b_2, \cdots, b_m)\colon b_k \in \boldsymbol{\mathcal{B}}, k = 1, \dots, m\}$.

Thus, theorem 3.2 has been proved.

### 3.5. Solution method

In this section, we formulate our problem as a shortest path problem on an acyclic network.

#### 3.5.1. Network construction

The network in the formulation consists of a pseudo-source node, a pseudo-sink node, and $m$ layers of nodes: it has one layer corresponding to each RP $\mathcal{p}_k \in \{\mathcal{p}_1, \mathcal{p}_2, \cdots, \mathcal{p}_m\}$. The layer $k$ has $n_k$ nodes, $\{n^k_1, n^k_2, \cdots, n^k_{n_k}\}$, where $n^k_i$ denotes the $i^{th}$ breakpoint (in the left to right order) in localization segment $S_k$ and signifies that $\mathcal{p}_k$ is established at that breakpoint.

Connecting the nodes

Consider two breakpoints $n^k_i$ and $n^{k+1}_j$. Recall that in line problem, $n^k_i$ and $n^{k+1}_j$ are connected to each other if and only if the distance between them is less than or equal to $r$. However, we should note that in comb tree problem, between the two candidate sites $n^k_i$ and $n^{k+1}_j$, there may exist leaf nodes with RPs. That is, the

reachability between $n_i^k$ and $n_j^{k+1}$ does not necessarily require that the distance between $n_i^k$ and $n_j^{k+1}$ is not more than $r$. Then how shall we decide whether or not they can be connected? Recall the reachability graph that we introduced in section 3.2. Here, we can solve the node connection problem on a reachability graph $G^{\sim}$, which is defined as:

- $V(G^{\sim}) = \{n_i^k, n_j^{k+1}\} \cup V_k^{\mathcal{P}}(n_i^k, n_j^{k+1})$, i.e., the node set of $G^{\sim}$ contains $\wp_k$ and $\wp_{k+1}$ (which are supposed to have been established at $n_i^k$ and $n_j^{k+1}$, as well as all RP nodes that are on the right-hand side of $n_i^k$ and on the left-hand side of $n_j^{k+1}$ (denoted as $V_k^{\mathcal{P}}(n_i^k, n_j^{k+1})$).
- $A(G^{\sim}) = \{(x, y): d(x, y) \leq r\}$, i.e., for any two nodes $x, y \in V(G^{\sim})$, the graph contains an arc $(x, y)$ if $d(x, y) \leq r$ on the original comb.

Then, we connect the two breakpoints $n_i^k$ and $n_j^{k+1}$ if

- The number of connected components of graph $G^{\sim}$, $b_0(G^{\sim})$, is equal to 1 (by performing search algorithm, $b_0(G^{\sim})$ can be determined).
- For every node $u \in V_k(n_i^k, n_j^{k+1}) \backslash V_k^{\mathcal{P}}(n_i^k, n_j^{k+1})$, there exists a node $v \in V(G^{\sim})$ such that $d(u, v) \leq r$.

<u>Defining edge weights</u>

Denote $w(n_i^k, n_j^{k+1})$ the weight of edge $(n_i^k, n_j^{k+1})$. Let

$$w\left(n_i^k, n_j^{k+1}\right) = \sum_{\substack{(s,t):s\le n_i^k,\\ t\ge n_j^{k+1}}} \left(f_{(s,t)} + f_{(t,s)}\right) * \Delta_{(\mathscr{p}_k,\mathscr{p}_{k+1})} + \sum_{\substack{(s,t):s\in(n_i^k,n_j^{k+1}],\\ t\ge n_j^{k+1}}} \left(f_{(s,t)} + f_{(t,s)}\right) * \Delta_{(s,\mathscr{p}_{k+1})}$$

$$+ \sum_{\substack{(s,t):s\le n_i^k,\\ t\in[n_i^k,n_j^{k+1})}} \left(f_{(s,t)} + f_{(t,s)}\right) * \Delta_{(\mathscr{p}_k,t)} + \sum_{(s,t):n_i^k<s<t<n_j^{k+1}} \left(f_{(s,t)} + f_{(t,s)}\right) * \Delta_{(s,t)}. \tag{3.24}$$

Particularly,

$$w\left(\text{source}, n_j^1\right) = \sum_{\substack{(s,t):s\in\left[0,n_j^1\right],\\ l_1(t)\ge n_j^1}} \left(f_{(s,t)} + f_{(t,s)}\right) * \Delta_{(s,\mathscr{p}_1)} + \sum_{(s,t):0\le s<t<n_j^1} \left(f_{(s,t)} + f_{(t,s)}\right) * \Delta_{(s,t)} \tag{3.25}$$

$$w\left(n_j^m, \text{sink}\right) = \sum_{\substack{(s,t):s\le n_j^m,\\ t\in[n_j^m,l_1(v_n))}} \left(f_{(s,t)} + f_{(t,s)}\right) * \Delta_{(\mathscr{p}_m,t)} + \sum_{(s,t):n_j^m<s<t\le l_1(v_n)} \left(f_{(s,t)} + f_{(t,s)}\right) * \Delta_{(s,t)} \tag{3.26}$$

To calculate such terms $\Delta_{(\mathscr{p}_k,\mathscr{p}_{k+1})}$, $\Delta_{(s,\mathscr{p}_{k+1})}$, $\Delta_{(\mathscr{p}_k,t)}$ and $\Delta_{(s,t)}$ in (3.24), we will use the shortest electric vehicle walk problem (Adler, J.D. et al., 2014). For example, to calculate $\Delta_{(s,t)}$ with $n_i^k < s < t < n_j^{k+1}$, we need to construct a meta-network on node set $\{s,t\} \cup \{\mathscr{p}_k, \mathscr{p}_{k+1}\} \cup V_k^{\mathcal{P}}\left(n_i^k, n_j^{k+1}\right)$, where the nodes in this meta-network have an edge if the two nodes are mutually reachable in a single charge in the original comb tree graph. That is, the edge between two nodes in this meta-network corresponds to a shortest path with length less than or equal to $r$ in the original comb graph. Let $s$ be the origin node and $t$ be the destination node. Then the shortest path in this

meta-network from $s$ to $t$ corresponds to a shortest walk from $s$ to $t$ in the original comb graph without a limit on refueling stops. Hence,

$$\Delta_{(s,t)} = \text{length of the shortest path } \mathrm{P}^*(s,t) \text{ on meta network} - d(s,t). \tag{3.27}$$

In example 3.5, we will illustrate the idea of how to connect the breakpoints and compute the corresponding weights.

### 3.5.2. Correctness

Let $P = \left(\text{source}, n_{r_1}^1, n_{r_2}^2, \cdots, n_{r_m}^m, \text{sink}\right)$ be a path from the pseudo source to the pseudo sink on the network, where $r_k$ denote the rank of the node $n_{r_k}^k$ in layer $k$. For notational convenience, we let $f_{st} = f_{(s,t)} + f_{(t,s)}$ be the total flow volume of O-D pair $(s,t)$ and O-D pair $(t,s)$. Then,

$$W(P) = w\left(\text{source}, n_{r_1}^1\right) + \sum_{k=1}^{m-1} w\left(n_{r_k}^k, n_{r_{k+1}}^{k+1}\right) + w\left(n_{r_m}^m, \text{sink}\right)$$

$$= \sum_{\substack{(s,t): s\in\left[0, n_j^1\right], \\ l_1(t)\geq n_j^1}} f_{st} * \Delta_{(s,p_1)} + \sum_{(s,t):0\leq s<t<n_j^1} f_{st} * \Delta_{(s,t)}$$

$$+ \sum_{k=1}^{m-1} \left( \sum_{\substack{(s,t): s\leq n_{r_k}^k, \\ t\geq n_{r_{k+1}}^{k+1}}} f_{st} * \Delta_{(p_k,p_{k+1})} + \sum_{\substack{(s,t): s\in\left(n_{r_k}^k, n_l^{i+1}\right], \\ t\geq n_{r_{k+1}}^{k+1}}} f_{st} * \Delta_{(s,p_{k+1})} \right.$$

$$\left. + \sum_{\substack{(s,t): s\leq n_{r_k}^k, \\ t\in\left[n_{r_k}^k, n_{r_{k+1}}^{k+1}\right)}} f_{st} * \Delta_{(p_k,t)} + \sum_{(s,t): n_{r_k}^k<s<t<n_{r_{k+1}}^{k+1}} f_{st} * \Delta_{(s,t)} \right)$$

$$+ \sum_{\substack{(s,t): s\leq n_j^m, \\ t\in\left[n_j^m, l_1(v_n)\right)}} f_{st} * \Delta_{(p_m,t)} + \sum_{(s,t): n_j^m<s<t\leq l_1(v_n)} f_{st} * \Delta_{(s,t)}$$

$$= \sum_{k=1}^{m-1} \sum_{\substack{(s,t): s \le l_1(p_k), \\ t \ge l_1(p_{k+1})}} f_{st} * \Delta_{(p_k, p_{k+1})} + \sum_k \sum_{\substack{(s,t):\, s \in (l_1(p_k), l_1(p_{k+1})], \\ t \ge l_1(p_{k+1})}} f_{st} * \Delta_{(s, p_{k+1})}$$

$$+ \sum_k \sum_{\substack{(s,t):\, t \in [l_1(p_k), l_1(p_{k+1})), \\ s \le l_1(p_k)}} f_{st} * \Delta_{(p_k, t)}$$

$$+ \sum_k \sum_{(s,t): l_1(p_k) < s < t < l_1(p_{k+1})} f_{st} * \Delta_{(s,t)}$$

$= Z(\boldsymbol{x})$ ($t$otal refueling detouring distance, see $eq.$ (3.10))

Therefore, the total weight of this path $P$ is equal to the total refueling detouring distance associated with locating RPs at $n_{r_1}^1, n_{r_2}^2, \cdots, n_{r_m}^m$.

Hence, the shortest path in our constructed network corresponds to an optimal set of RP locations on the comb.

Example 3.5 Consider the same comb tree used in example 3.4. Recall in example 3.4, we have computed the set of breakpoints, where

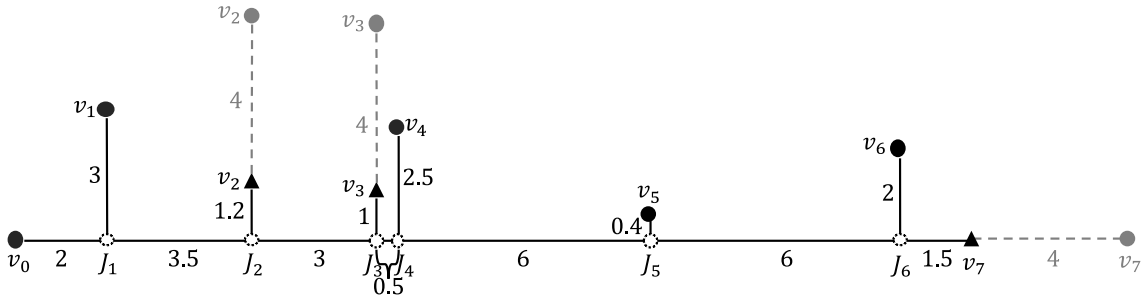



**Figure 3.13** A copy of Figure 3.6

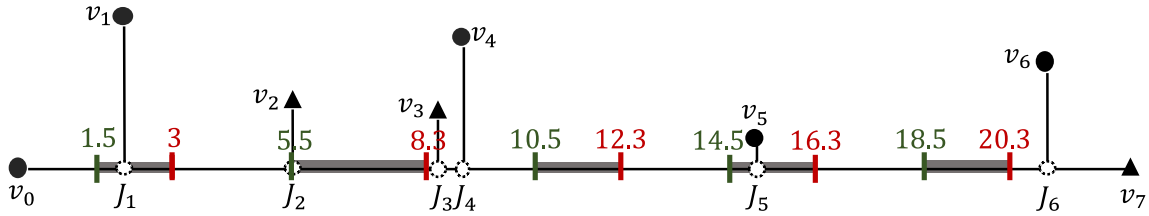



**Figure 3.14** A copy of Figure 3.8

$S_1$ contains 6 breakpoints: 1.5, 2, 2.5, 2.6, 2.7, 3;

$S_2$ contains 9 breakpoints: 5.5, 6, 6.5, 6.6, 6.7, 7, 7.4, 7.5, 8.3;

$S_3$ contains 7 breakpoints: 10.5, 10.6, 10.7, 11, 11.4, 11.5, 12.3;

$S_4$ contains 7 breakpoints: 14.5, 14.6, 14.7, 15, 15.4, 15.5, 16.3;

$S_5$ contains 7 breakpoints: 18.5, 18.6, 18.7, 19, 19.4, 19.5, 20.3.

Then, our network should contain 5 layers of nodes, and one source node and one sink node, as shown in Figure 3.15.

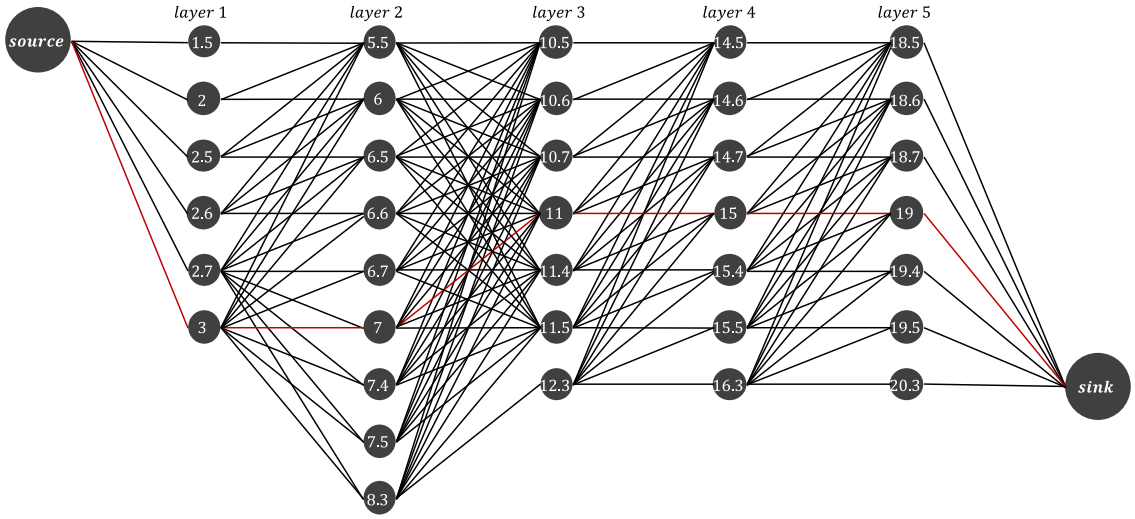



**Figure 3.15** The constructed network, the edges weights are not listed

To illustrate the idea of connecting two breakpoints, we take $(n_1^2, n_6^3) = (5.5, 11.5)$ for example. Since $v_3^-(r) = l_1(v_3) - \big(r - l_2(v_3)\big) = 8.5 - (4 - 1) = 5.5$, $v_3^+(r) = l_1(v_3) + \big(r - l_2(v_3)\big) = 8.5 + (4 - 1) = 11.5$ and $d(v_3, v_4) = 1 + 0.5 + 2.5 = 4 \leq r$. Therefore, $n_1^2$ and $n_6^3$ should be connected.

To illustrate the idea of computing edge weights, we take $w(n_9^2, n_4^3) = w(8.3, 11)$ for example:

$w(n_9^2, n_4^3) = w(8.3, 11)$

$$= \sum_{\substack{(s,t):s\le 8.3,\\ t\ge 11}} f_{st} * \Delta_{(\mathcal{p}_2,\mathcal{p}_3)} + \sum_{\substack{(s,t):s\in(8.3,11],\\ t\ge 11}} f_{st} * \Delta_{(s,\mathcal{p}_3)}$$

$$+ \sum_{\substack{(s,t):s\le 8.3,\\ t\in[8.3,11)}} f_{st} * \Delta_{(\mathcal{p}_2,t)} + \sum_{(s,t):8.3<s<t<11} f_{st} * \Delta_{(s,t)}$$

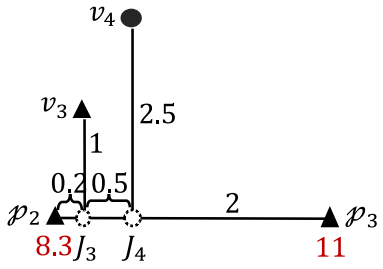
where

$\sum_{\substack{(s,t):s\in(8.3,11],\\ t\ge 11}} f_{st} * \Delta_{(s,\mathcal{p}_3)} = \sum_{s\in\{v_3,v_4\}} \sum_{t:\, t>s} f_{st} * \Delta_{(s,\mathcal{p}_3)}$;

$\sum_{\substack{(s,t):s\le 8.3,\\ t\in[8.3,11)}} f_{st} * \Delta_{(\mathcal{p}_2,t)} = \sum_{t\in\{v_3,v_4\}} \sum_{s:\, s<t} f_{st} * \Delta_{(\mathcal{p}_2,t)}$;

$\sum_{(s,t):8.3<s<t<11} f_{st} * \Delta_{(s,t)} = f_{st} * \Delta_{(v_3,v_4)}$.

Let's consider the following subgraph in Figure 3.16 of our original comb by establishing $\mathcal{p}_2$ at 8.3, and $\mathcal{p}_3$ at 11, and the shortest distance matrix is also shown below:




| $d(,)$ | $\mathcal{p}_2$ | $\mathcal{p}_3$ | $v_3$ | $v_4$ |
|---|---|---|---|---|
| $\mathcal{p}_2(=8.3)$ | 0 | 2.7 | 1.2 | 3.2 |
| $\mathcal{p}_3(=11)$ | | 0 | 3.5 | 4.5 |
| $v_3$ | | | 0 | 4 |
| $v_4$ | | | | 0 |

**Figure 3.16** A subgraph of the original comb, assuming that $\mathcal{p}_2$ has established at **8.3**, and $\mathcal{p}_3$ has been established at **11**

From the shortest distance matrix, we know that $\Delta_{(v_3,\mathcal{p}_3)} = 0$, $\Delta_{(v_3,v_4)} = 0$, and by constructing the following meta-network, where we let $v_4$ be the origin node and $\mathcal{p}_3$ be the destination node, and the numbers beside an arc is the corresponding shortest path distance on the subgraph in Figure 3.17. The red path corresponds to a shortest refueling walk from $v_4$ to $\mathcal{p}_3$ on the original comb, i.e., $v_4 \to \mathcal{p}_2 \to \mathcal{p}_3$, with a refueling detouring distance of $2 \times (0.2 + 0.5) = 1.4$. And $\Delta_{(v_4,\mathcal{p}_3)} = 3.2 + 2.7 - 4.5 = 1.4$.

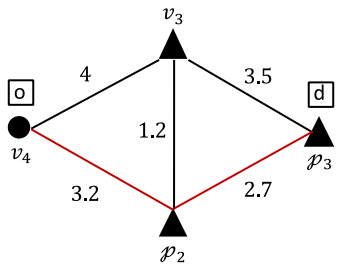



**Figure 3.17** A meta-network, where $\mathcal{p}_2$ has been established at **8.3**, and $\mathcal{p}_3$ has been established at 11

Assuming that the traffic flow on all O-D pairs are equal, with $f_{st} = 20$. Then, the shortest path on the constructed network is "source → 3 → 7 → 11 → 15 → 19 → sink", as shown in Figure 3.14, and the path length is $40 + 0 + 300 + 0 + 0 + 0 = 340$, where $w(\text{source}, 3) = 40$, $w(7, 11) = 300$, and $w(3, 7) = w(11, 15) = w(15, 19) = w(19, \text{sink}) = 0$.

By this algorithm, we derive the optimal RP locations: $\mathcal{p}_1 = 3$, $\mathcal{p}_2 = 7$, $\mathcal{p}_3 = 11$, $\mathcal{p}_4 = 15$ and $\mathcal{p}_5 = 19$, and the total refueling detouring distance equal to 340, with $f_{st} = 20$ for all $(s, t)$.

**Remarks** The comb tree problem can be solved in polynomial-time.

***Theorem 3.3*** The comb problem can be solved in $\mathcal{O}(n^5)$.

*Proof of Theorem 3.3.* An upper bound of the minimum number of RPs that are necessary and sufficient to serve all one-way trips is $\left\lfloor \frac{l_1(v_n)}{r} \right\rfloor + n - 1$. Let $T_1$ represents the time required to construct the network and let $T_2$ represent the time required to find the shortest path in the constructed network.

(1) $T_1 = \mathcal{O}(mn^3)$. By the nature of the line network problem and by the construction of the multistage network note that the number of nodes in each node layer in our constructed multistage network is at most $2n$. Also, note that we add

one artificial source node and one artificial sink node. By the way we connect two nodes and calculate an edge weight, we know that the time required to determine whether or not two nodes should be connected and to find the weight of an edge is $\mathcal{O}(n^2)$. Thus, $T_1 = \mathcal{O}\left(\left(\left\lfloor\frac{l_1(v_n)}{r}\right\rfloor + n - 1\right) * 4n^2 * n^2\right) = \mathcal{O}(n^5)$.

(2) $T_2 = \mathcal{O}(n^2)$. Note that the total number of nodes in our multistage network is at most $\left(\left\lfloor\frac{l_1(v_n)}{r}\right\rfloor + n - 1\right) * 2n + 2$. Thus, finding the shortest path in the constructed multistage network will take $\mathcal{O}\left(\left(\left(\left\lfloor\frac{l_1(v_n)}{r}\right\rfloor + n - 1\right) * 2n + 2\right)^2\right) = \mathcal{O}(n^2)$ time.

The overall run time T of the line problem is $T_1 + T_2 = \mathcal{O}(n^5)$. ■

### 3.6. Conclusion

In this chapter, we studied the continuous location problem related to locating RPs on comb tree networks. To find the fewest number of RPs needed to serve all one-way O-D pairs, we proposed a 2-step greedy method. Then, we proposed a math programming formulation, based on which we proved the existence of a finite dominating set to the comb tree problem. Then we formulated the problem as a shortest path problem whereby the shortest path of the constructed network gives us an optimal set of RP locations.

Beyond the scope of current study, there are several issues worth of a further investigation. For instance, it is our interest to see how to revise the current proposed math programming formulation so that it can be solved using commercial solvers. How to construct the network that will be used in the shortest path problem in a more efficient way will require a further research too. Moreover, an extension of

current work to the round-trip scenario needs to be investigated as well, which will not cost too much extra work.

# CHAPTER 4

# PROBABILISTIC LINE AND COMB PROBLEM

## 4.1. Overview

In the last two chapters, we presented the problem of locating multiple RPs on a deterministic line network and/or comb network. The assumption behind our results is that battery charge decreases uniformly on distances throughout the network. In this chapter, we consider the location problem on probabilistic networks. How many RPs must be located on a network and where should these RPs be located such that all transportation needs are satisfied, and the average transportation cost is minimized? Consider this simple case of a line network:

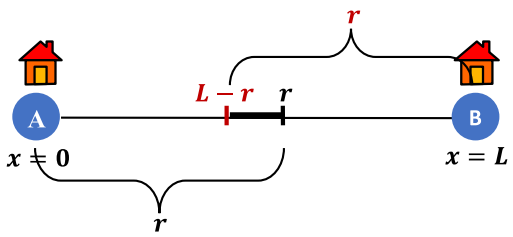



**Figure 4.1** A simple line network

here $r$ is a driving range, say in miles, and $L$ is distance miles. These are physical distances and a point in the network is a fixed physical point. Let's say that this is one possibility of the state of the network, say with probability 2/3. Here $r$ the "discharging range" coincides with physical distance. Now we have another state, a congested state, where the new discharging range $r^{new}$ is 0.75 of the old $r$. What can we say about the localization set?

The class of probabilistic networks considered in this chapter includes networks for which the values of the travel attributes associated with the links are random. Another class of probabilistic networks encountered in the literature consists of network whose topology itself is random. For example, when the network is under attack, determining the nodes that have the highest probability of being connected with a given node or selecting a given number of nodes that have the least possibility of being disconnected. However, we do not consider such networks in this thesis.

## 4.2. Minimum number of RPs needed

Now consider the same line network that we used in chapter 2,

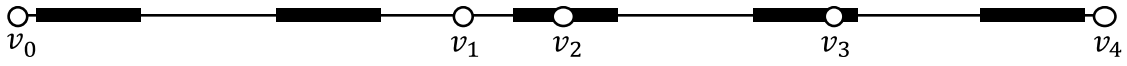



**Figure 4.2** A copy of Figure 2.4

assume discharging range for state 1 is $r$, and new discharging ranges for state 2 are

between $v_0$ and $v_1$ is $0.9 \cdot r$;

between $v_1$ and $v_2$ is $0.8 \cdot r$;

between $v_2$ and $v_3$ is $0.8 \cdot r$;

between $v_3$ and $v_4$ is $0.9 \cdot r$.

What can we say about localization sets now?

***Proposition 4.1*** Given a set of RP locations on the line network, if flows on round trips $(v_0, v_n, v_0)$ and $(v_n, v_0, v_n)$ can be refueled, then flows on any round trip $(v_i, v_j, v_i)$ can be refueled, where $0 \leq i, j \leq n$.

*Proof of Proposition 4.1.* Let $\mathcal{P} = \{\wp_1, \cdots, \wp_m\}$ be a set of RPs established on the line network such that both round trips $(v_0, v_n, v_0)$ and $(v_n, v_0, v_n)$ can be served. Then we are able to conclude that:

- the vehicle is able to reach $\wp_1$ with $\frac{r}{2}$ level of charge from node $v_0$;
- the vehicle is able to traverse between every two adjacent RPs, $\wp_k$ and $\wp_{k+1}$, with a single charge, where $1 \leq k < m - 1$;
- the vehicle is able to reach node $v_n$ with $\frac{r}{2}$ level of charge from $\wp_m$.

Consider any round trip $(v_i, v_j, v_i)$, and without loss of generality we assume that $v_i$ is on the LHS of $v_j$. Then, the outbound trip $v_i \to v_j$ is refuellable, since the vehicle is able to traverse each of the following intervals with a single charge: the interval between node $v_i$ and $\wp_{v_i}^+$ (the closest RP on the RHS of $v_i$), the interval between $\wp_{v_j}^-$ (the closest RP on the LHS of $v_j$) and node $v_j$, and the interval between every two adjacent RPs, $\wp_k$ and $\wp_{k+1}$, where $\tau(\wp_{v_i}^+) \leq k < \tau\left(\wp_{v_j}^-\right)$. The inbound trip $v_j \to v_i$ is refuellable as well. Let's consider the following two cases:

- if the vehicle is able to reach $v_j$ from $\wp_{v_j}^-$ with $\frac{r}{2}$ level of charge, after arriving at $v_j$, the vehicle is able to return to $\wp_{v_j}^-$ without running out of battery;
- if strictly greater than $\frac{r}{2}$ level of charge is needed for the vehicle to traverse from $\wp_{v_j}^-$ to $v_j$, however, the vehicle is able to make a detour to visit $\wp_{v_j}^+$ for refueling and goes back to $\wp_{v_j}^-$, since the level of charge needed to traverse between $\wp_{v_j}^-$ and $\wp_{v_j}^+$ is not more than $r$. ■

For each segment $j$ between nodes $v_{j-1}$ and $v_j$, denote $b_j$ the length of this segment, and denote $\vartheta_j$ the ratio of new discharging range for state 2 to the discharging range for state 1 of this segment.

Assuming that round trip $(v_0, v_n, v_0)$ is refuellable with this set of RPs, let's see how these RPs should be allocated on the line. First, consider the outbound trip $v_0 \to v_n$. Assuming that the vehicle starts with a fully charged battery, then the farthest point it can reach with a single charge is

$$v_0{}^+(r) = \begin{cases} \vartheta_1 r, & \vartheta_1 r \le b_1 \\ b_1 + \vartheta_2\left(r - \frac{b_1}{\vartheta_1}\right), & 0 < \vartheta_2\left(r - \frac{b_1}{\vartheta_1}\right) \le b_2 \\ b_1 + b_2 + \vartheta_3\left(r - \frac{b_1}{\vartheta_1} - \frac{b_2}{\vartheta_2}\right), & 0 < \vartheta_3\left(r - \frac{b_1}{\vartheta_1} - \frac{b_2}{\vartheta_2}\right) \le b_3 \\ \vdots, & \vdots \end{cases}. \tag{4.1}$$

That is,

- if the length of segment 1 ($b_1$) is greater than or equal to $\vartheta_1 r$, the vehicle will not be able to traverse the whole segment, and the farthest point it can reach is $\vartheta_1 r$;
- if $b_1 < \vartheta_1 r$, then the remaining level of charge of the vehicle when it arrives at node $v_1$ is $r - \frac{b_1}{\vartheta_1}$, and furthermore,
  - if $\vartheta_2\left(r - \frac{b_1}{\vartheta_1}\right) \le b_2$, the vehicle will not be able to traverse segment 2 and the farthest point it can reach is $b_1 + \vartheta_2\left(r - \frac{b_1}{\vartheta_1}\right)$;
  - if $\vartheta_2\left(r - \frac{b_1}{\vartheta_1}\right) > b_2$, the vehicle is able to reach node $v_2$, with a remaining level of charge equal to $r - \frac{b_1}{\vartheta_1} - \frac{b_2}{\vartheta_2}$, ….

After identifying the point $v_0{}^+(r)$, which is regarded as an extreme (farthest from node $v_0$) site for the first RP to be established at, we can continue this fashion

to find the set of extreme sites for the rest of RPs, until the vehicle is able to reach node $v_n$ from the last site that has been identified without running out of battery.

Then, let's consider the inbound trip $v_n \to v_0$. The farthest point from node $v_n$ at which the last RP should be established at is as follows,

$$v_n^-(r) = \begin{cases} L - \frac{\vartheta_n r}{2}, & \vartheta_n r \le 2b_n \\ L - b_n - \vartheta_{n-1}\left(\frac{r}{2} - \frac{b_n}{\vartheta_n}\right), & 0 < \vartheta_{n-1}\left(\frac{r}{2} - \frac{b_n}{\vartheta_n}\right) \le b_{n-1} \\ L - b_n - b_{n-1} - \vartheta_{n-2}\left(\frac{r}{2} - \frac{b_n}{\vartheta_n} - \frac{b_{n-1}}{\vartheta_{n-1}}\right), & 0 < \vartheta_{n-2}\left(\frac{r}{2} - \frac{b_n}{\vartheta_n} - \frac{b_{n-1}}{\vartheta_{n-1}}\right) \le b_{n-2} \\ \vdots, & \vdots \end{cases} \quad (4.2)$$

More specifically,

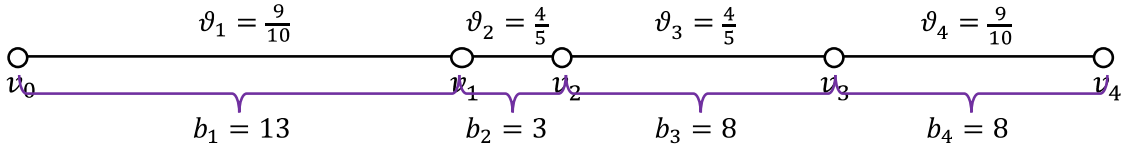
- if $\frac{b_n}{\vartheta_n} \ge \frac{r}{2}$, i.e., the level of charge that is necessary for the vehicle to traverse segment $n$ is at least $\frac{r}{2}$, then the last RP should be established within $\frac{\vartheta_n r}{2}$ distance from node $v_n$;
- if $\frac{b_n}{\vartheta_n} < \frac{r}{2}$, it is trivial to see that the last RP can be established on the right-hand side of node $v_{n-1}$, and furthermore,
  - if $\vartheta_{n-1}\left(\frac{r}{2} - \frac{b_n}{\vartheta_n}\right) \le b_{n-1}$ (i.e., $\frac{b_{n-1}}{\vartheta_{n-1}} + \frac{b_n}{\vartheta_n} \ge \frac{r}{2}$), the level of charge that is necessary for the vehicle to traverse segments $n-1$ and $n$ is at least $\frac{r}{2}$, then the last RP should be established within $b_n + \vartheta_{n-1}\left(\frac{r}{2} - \frac{b_n}{\vartheta_n}\right)$ distance from node $v_n$;
  - if $\vartheta_{n-1}\left(\frac{r}{2} - \frac{b_n}{\vartheta_n}\right) > b_{n-1}$, the vehicle is able to traverse segments $n-1$ and $n$ with $\frac{r}{2}$ level of charge, ….

After identifying the point $v_n^-(r)$, which is regarded as an extreme site (farthest to node $v_n$) for the last RP to be established at, we can continue this fashion to find the

set of extreme sites for the rest of RPs, until the vehicle is able to reach node $v_0$ with a single charge.

For round trip $(v_n, v_0, v_n)$, we can follow the same procedure as above.

Now, let's turn to the situation described in the problem, where $b_1 = 13$, $b_2 = 3$, $b_3 = 8$, $b_4 = 8$, $\vartheta_1 = \vartheta_4 = \frac{9}{10}$, $\vartheta_2 = \vartheta_3 = \frac{4}{5}$, and the discharging range for state 1 is $r = 7$.




First, consider round trip $(v_0, v_4, v_0)$. Let $\boldsymbol{\beta}^1$ denote the set of extreme sites for RPs to refuel the outbound trip $v_0 \to v_4$, and $\boldsymbol{\alpha}^1$ denote the set of extreme sites for RPs to refuel the inbound trip $v_4 \to v_0$. We can iteratively compute $\boldsymbol{\beta}^1$ using equation (1):

1. $\boxed{\beta_1^1}$ Since $\frac{b_1}{\vartheta_1} = \frac{13}{\frac{9}{10}} > r$, then $\beta_1^1 = \vartheta_1 r = \frac{9}{10} \times 7 = 6\frac{3}{10}$, and we let $b_1 \leftarrow b_1 - \vartheta_1 r = 6\frac{7}{10}$;
2. $\boxed{\beta_2^1}$ Since $\frac{b_1}{\vartheta_1} = \frac{6\frac{7}{10}}{\frac{9}{10}} > r$, then $\beta_2^1 = \beta_1^1 + \vartheta_1 r = 12\frac{3}{5}$, and we let $b_1 \leftarrow b_1 - \vartheta_1 r = 6\frac{7}{10} - 6\frac{3}{10} = \frac{2}{5}$;

3. $\boxed{\beta_3^1}$ Since $\begin{cases} \frac{b_1}{\vartheta_1} = \frac{\frac{2}{5}}{\frac{9}{10}} < r \\ \frac{b_1}{\vartheta_1} + \frac{b_2}{\vartheta_2} = \frac{\frac{2}{5}}{\frac{9}{10}} + \frac{3}{\frac{4}{5}} < r \\ \frac{b_1}{\vartheta_1} + \frac{b_2}{\vartheta_2} + \frac{b_3}{\vartheta_3} = \frac{\frac{2}{5}}{\frac{9}{10}} + \frac{3}{\frac{4}{5}} + \frac{8}{\frac{4}{5}} > r \end{cases}$, then $\beta_3^1 = (\beta_2^1 + b_1 + b_2) + \vartheta_3\left(r - \frac{b_1}{\vartheta_1} - \frac{b_2}{\vartheta_2}\right) = 13 + \frac{4}{5}\left(7 - \frac{\frac{2}{5}}{\frac{9}{10}} - \frac{3}{\frac{4}{5}}\right) = 18\frac{11}{45}$, and we let $b_3 \leftarrow b_3 - \vartheta_3\left(r - \frac{b_1}{\vartheta_1} - \frac{b_2}{\vartheta_2}\right) = 5\frac{34}{45}$, $b_2 \leftarrow 0$, $b_1 \leftarrow 0$;

4. $\boxed{\beta_4^1}$ Since $\frac{b_3}{\vartheta_3} = \frac{5\frac{34}{45}}{\frac{4}{5}} > r$, then $\beta_4^1 = \beta_3^1 + \vartheta_3 r = 23\frac{38}{45}$, and we let $b_3 \leftarrow b_3 - \vartheta_3 r = \frac{7}{45}$;

5. $\boxed{\beta_5^1}$ Since $\begin{cases} \frac{b_3}{\vartheta_3} = \frac{\frac{7}{45}}{\frac{4}{5}} < r \\ \frac{b_3}{\vartheta_3} + \frac{b_4}{\vartheta_4} = \frac{\frac{7}{45}}{\frac{4}{5}} + \frac{8}{\frac{9}{10}} > r \end{cases}$, then $\beta_5^1 = (\beta_4^1 + b_3) + \vartheta_4\left(r - \frac{b_3}{\vartheta_3}\right) = 24 + \frac{9}{10}\left(7 - \frac{\frac{7}{45}}{\frac{4}{5}}\right) = 30\frac{1}{8}$, and we let $b_4 \leftarrow b_4 - \vartheta_4\left(r - \frac{b_3}{\vartheta_3}\right) = 1\frac{7}{8}$, $b_3 \leftarrow 0$.

6. $\boxed{\text{End}}$ Since $\frac{b_4}{\vartheta_4} = \frac{1\frac{7}{8}}{\frac{9}{10}} < r$, the vehicle is able to reach node $v_4$ in a single charge.

Then, by using equation (2), we can compute $\alpha_5^1$, and again by using equation (1), we can iteratively compute the rest of $\boldsymbol{\alpha}^1$:

1. $\boxed{\alpha_5^1}$ Since $\frac{b_4}{\vartheta_4} = \frac{8}{\frac{9}{10}} > \frac{r}{2}$, then $\alpha_5^1 = L - \frac{\vartheta_4 r}{2} = 32 - \frac{\frac{9}{10}\times 7}{2} = 28\frac{17}{20}$, and we let $b_4 \leftarrow b_4 - \frac{\vartheta_4 r}{2} = 4\frac{17}{20}$;

2. $\boxed{\alpha_4^1}$ Since $\begin{cases} \frac{b_4}{\vartheta_4} = \frac{4\frac{17}{20}}{\frac{9}{10}} < r \\ \frac{b_4}{\vartheta_4} + \frac{b_3}{\vartheta_3} = \frac{4\frac{17}{20}}{\frac{9}{10}} + \frac{8}{\frac{4}{5}} > r \end{cases}$, then $\alpha_4^1 = (\alpha_5^1 - b_4) - \vartheta_3\left(r - \frac{b_4}{\vartheta_4}\right) = 24 - \frac{4}{5}\left(7 - \frac{4\frac{17}{20}}{\frac{9}{10}}\right) = 22\frac{32}{45}$, and we let $b_3 \leftarrow b_3 - \vartheta_3\left(r - \frac{b_4}{\vartheta_4}\right) = 6\frac{32}{45}$, $b_4 \leftarrow 0$;

3. $\boxed{\alpha_3^1}$ Since $\frac{b_3}{\vartheta_3}=\frac{6\frac{32}{45}}{\frac{4}{5}}>r$, $\alpha_3^1=\alpha_4^1-\vartheta_3 r=22\frac{32}{45}-\frac{4}{5}\times 7=17\frac{1}{9}$, and we let $b_3\leftarrow b_3-\vartheta_3 r=1\frac{1}{9}$;

4. $\boxed{\alpha_2^1}$ Since $\begin{cases}\frac{b_3}{\vartheta_3}=\frac{1\frac{1}{9}}{\frac{4}{5}}<r\\ \frac{b_3}{\vartheta_3}+\frac{b_2}{\vartheta_2}=\frac{1\frac{1}{9}}{\frac{4}{5}}+\frac{3}{\frac{4}{5}}<r\\ \frac{b_3}{\vartheta_3}+\frac{b_2}{\vartheta_2}+\frac{b_1}{\vartheta_1}=\frac{1\frac{1}{9}}{\frac{4}{5}}+\frac{3}{\frac{4}{5}}+\frac{13}{\frac{9}{10}}>r\end{cases}$, then $\alpha_2^1=(\alpha_3^1-b_3-b_2)-\vartheta_1\left(r-\frac{b_3}{\vartheta_3}-\frac{b_2}{\vartheta_2}\right)=13-\frac{9}{10}\left(7-\frac{1\frac{1}{9}}{\frac{4}{5}}-\frac{3}{\frac{4}{5}}\right)=11\frac{13}{40}$, and we let $b_1\leftarrow b_1-\vartheta_1\left(r-\frac{b_3}{\vartheta_3}-\frac{b_2}{\vartheta_2}\right)=11\frac{13}{40}$, $b_3\leftarrow 0$, $b_2\leftarrow 0$.

5. $\boxed{\alpha_1^1}$ Since $\frac{b_1}{\vartheta_1}=\frac{11\frac{13}{40}}{\frac{9}{10}}>r$, then $\alpha_1^1=\alpha_2^1-\vartheta_1 r=11\frac{13}{40}-\frac{9}{10}\times 7=5\frac{1}{40}$, and we let $b_1\leftarrow b_1-\vartheta_1 r=5\frac{1}{40}$;

6. $\boxed{\text{End}}$ Since $\frac{b_1}{\vartheta_1}=\frac{5\frac{1}{40}}{\frac{9}{10}}<r$, the vehicle is able to reach node $v_0$ in a single charge.

Therefore, to refuel round trip $(v_0, v_4, v_0)$, the minimum number of RPs needed is 5, and for each RP a localization segment is identified:

| To refuel $(v_0, v_4, v_0)$ | $\mathcal{P}_1$ | $\mathcal{P}_2$ | $\mathcal{P}_3$ | $\mathcal{P}_4$ | $\mathcal{P}_5$ |
|---|---|---|---|---|---|
| $\boldsymbol{\alpha}^1$ | $5\frac{1}{40}$ | $11\frac{13}{40}$ | $17\frac{1}{9}$ | $22\frac{32}{45}$ | $28\frac{17}{20}$ |
| $\boldsymbol{\beta}^1$ | $6\frac{3}{10}$ | $12\frac{3}{5}$ | $18\frac{11}{45}$ | $23\frac{38}{45}$ | $30\frac{1}{8}$ |

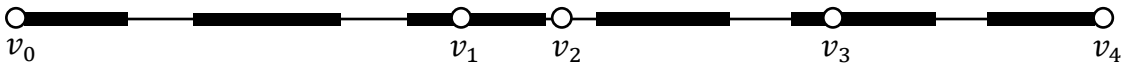
Similarly, to refuel round trip $(v_4, v_0, v_4)$, the minimum number of RPs needed is 5, and for each RP a localization segment is identified:

| To refuel $(v_4, v_0, v_4)$ | $\mathcal{P}_1$ | $\mathcal{P}_2$ | $\mathcal{P}_3$ | $\mathcal{P}_4$ | $\mathcal{P}_5$ |
|---|---|---|---|---|---|
| $\boldsymbol{\beta}^2$ | $1\frac{1}{8}$ | $7\frac{17}{40}$ | $13\frac{29}{45}$ | $19\frac{41}{45}$ | $25\frac{7}{10}$ |

| $\boldsymbol{\alpha^2}$ | $3\frac{3}{20}$ | $9\frac{9}{20}$ | $15\frac{4}{9}$ | $21\frac{2}{45}$ | $26\frac{39}{40}$ |
|---|---|---|---|---|---|

However, from the above two tables, we should note that these two set of localization segments are nonoverlapped, which implies that by establishing 5 RPs on the line network, there will be unsatisfied transportation needs. In fact, the minimum number of RPs that are necessary and sufficient to serval all round trips is 6, and the corresponding localization segments are:

| $\mathcal{P}_1$ | $\mathcal{P}_2$ | $\mathcal{P}_3$ | $\mathcal{P}_4$ | $\mathcal{P}_5$ | $\mathcal{P}_6$ |
|---|---|---|---|---|---|
| $[0, \alpha_1^2]$ | $[\alpha_1^1, \alpha_2^2]$ | $[\alpha_2^1, \alpha_3^2]$ | $[\alpha_3^1, \alpha_4^2]$ | $[\alpha_4^1, \alpha_5^2]$ | $[\alpha_5^1, L]$ |
| $\left[0, 3\frac{3}{20}\right]$ | $\left[5\frac{1}{40}, 9\frac{9}{20}\right]$ | $\left[11\frac{13}{40}, 15\frac{4}{9}\right]$ | $\left[17\frac{1}{9}, 21\frac{2}{45}\right]$ | $\left[22\frac{32}{45}, 26\frac{39}{40}\right]$ | $\left[28\frac{17}{20}, 32\right]$ |

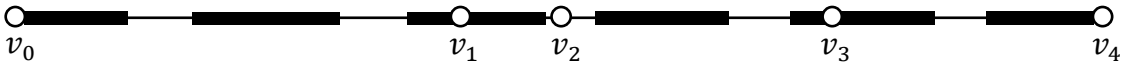



where,

- The first RP will be established with $3\frac{3}{20}$ distance from node $v_0$ (i.e., site $\alpha_1^2$, which is an extreme site to serve the inbound trip $v_0 \to v_4$ of $(v_4, v_0, v_4)$);
- The last RP will be established with $\left(32 - 28\frac{17}{20}\right)$ distance from node $v_4$ (i.e., site $\alpha_5^1$, which is an extreme site to serve the inbound trip $v_4 \to v_0$ of $(v_0, v_4, v_0)$);
- The vehicle is able to traverse between every two adjacent RPs with a single charge.

From part C), we get the following 6 localization segments:

| $\mathcal{P}_1$ | $\mathcal{P}_2$ | $\mathcal{P}_3$ | $\mathcal{P}_4$ | $\mathcal{P}_5$ | $\mathcal{P}_6$ |
|---|---|---|---|---|---|
| $\left[0, 3\frac{3}{20}\right]$ | $\left[5\frac{1}{40}, 9\frac{9}{20}\right]$ | $\left[11\frac{13}{40}, 15\frac{4}{9}\right]$ | $\left[17\frac{1}{9}, 21\frac{2}{45}\right]$ | $\left[22\frac{32}{45}, 26\frac{39}{40}\right]$ | $\left[28\frac{17}{20}, 32\right]$ |




### 4.3. Find optimal RPs' locations

For this two-state problem, we can conclude that there exists a finite dominating set to the problem, since the objective function is the expected value of the total refueling detouring distance in two states.

We can define the set of breakpoints as follows:

(1) $B_1$, the set of endpoints of each localization segments;

(2) $B_2$, the set of internal nodes;

(3) $B_3$, the union of the set of extreme none refueling detouring sites (within localization segments) in state 1 and state 2;

(4) $B_4 = \bigcup_{x\in B_1\cup B_2\cup B_3}\{\{x^-(ir), x^+(ir): i = 1,2,\cdots,m\} \cap \{\bigcup_{k=1}^{m} S_k\}\}$, where $x^{\mp}\big((i+1)r\big)$ denotes the point on the line that the vehicle is able to reach from $x^{\mp}(ir)$ with a fully charged battery in state 2. We should note that in this two-state problem, the distance between $x^{\mp}\big((i+1)r\big)$ and $x^{\mp}(ir)$ is not necessarily $r$.

For this specific problem, we have:

- $B_1 = \left\{0, 3\frac{3}{20}\right\} \cup \left\{5\frac{1}{40}, 9\frac{9}{20}\right\} \cup \left\{11\frac{13}{40}, 15\frac{4}{9}\right\} \cup \left\{17\frac{1}{9}, 21\frac{2}{45}\right\} \cup \left\{22\frac{32}{45}, 26\frac{39}{40}\right\} \cup \left\{28\frac{17}{20}, 32\right\}$;
- $B_2 = \{13, 24\}$, the set of internal nodes;
- In state 1, the set of extreme none refueling detouring sites are $B_3^1 = \left\{\cancel{3\frac{1}{2}}\right\} \cup \left\{\cancel{9\frac{1}{2}}, \cancel{16\frac{1}{2}}\right\} \cup \left\{12\frac{1}{2}, 19\frac{1}{2}\right\} \cup \left\{20\frac{1}{2}, \cancel{27\frac{1}{2}}\right\} \cup \left\{\cancel{28\frac{1}{2}}\right\}$; and in state 2, the set of extreme none refueling detouring sites are$B_3^2 = \left\{3\frac{3}{20}\right\} \cup \left\{\cancel{9\frac{17}{20}}, \cancel{15\frac{4}{5}}\right\} \cup \left\{13\frac{1}{5}, 18\frac{4}{5}\right\} \cup \left\{\cancel{21\frac{1}{5}}, \cancel{27\frac{3}{20}}\right\} \cup \left\{28\frac{17}{20}\right\}$. Then, $B_3 = \left\{3\frac{3}{20}, 12\frac{1}{2}, 13\frac{1}{5}, 18\frac{4}{5}, 19\frac{1}{2}, 20\frac{1}{2}, 28\frac{17}{20}\right\}$.

Let's illustrate how to compute the two extreme none refueling detouring sites associated with each node in state 2. For example, consider node $v_2$. The extreme none refueling detouring site on the LHS of $v_2$ should be at $16-\frac{4}{5}\times 7\times\frac{1}{2}=13\frac{1}{5}$, and the extreme none refueling detouring site on the RHS of $v_2$ should be at $16+\frac{4}{5}\times 7\times\frac{1}{2}=18\frac{4}{5}$.

- $B_4=\left\{\frac{4}{10},6\frac{7}{10},18\frac{3}{5},24\frac{9}{40},30\frac{21}{40}\right\}\cup\left\{\frac{7}{40},6\frac{19}{40},12\frac{31}{40},18\frac{2}{5},30\frac{3}{10}\right\}\cup$ $\left\{6\frac{1}{5},18\frac{7}{45},23\frac{34}{45},30\frac{1}{40}\right\}\cup\left\{\frac{5}{8},6\frac{37}{40},18\frac{4}{5},24\frac{9}{20},30\frac{3}{4}\right\}\cup\left\{1\frac{33}{80},7\frac{57}{80},13\frac{9}{10},25\frac{19}{80},31\frac{43}{80}\right\}\cup$ $\left\{2\frac{43}{80},8\frac{67}{80},14\frac{9}{10},26\frac{29}{80}\right\}$.

Let's take $x=13$ for example, $x^-(r)=13-\frac{9}{10}\times 7=6\frac{7}{10}$, $x^-(2r)=6\frac{7}{10}-\frac{9}{10}\times 7=\frac{4}{10}$, $x^+(r)=13+\frac{4}{5}\times 7=18\frac{3}{5}$, $x^+(2r)=24+\frac{9}{10}\times\left(7-\frac{24-18\frac{3}{5}}{\frac{4}{5}}\right)=24\frac{9}{40}$, and $x^+(3r)=24\frac{9}{40}+\frac{9}{10}\times 7=30\frac{21}{40}$.

Then, we are able to construct the network that will be used for the shortest path problem, where each node of the constructed network corresponds to a breakpoint on the line network, and for any two nodes that are in adjacent two layers, we connect them if the vehicle can traverse between them without running out battery in state 2. In fact, if the vehicle can traverse between these two nodes in a single in state 2, then it can traverse between them in state 1 as well, however, the opposite is not necessarily true.

For the edge weight of the constructed network, say $w\left(n_i^k,n_j^{k+1}\right)$ of edge $\left(n_i^k,n_j^{k+1}\right)$, we can compute is as

$$w\left(n_i^k,n_j^{k+1}\right)=p_1\times w_1\left(n_i^k,n_j^{k+1}\right)+p_2\times w_2\left(n_i^k,n_j^{k+1}\right),$$

where $p_1$ and $p_2$ are the probabilities of two states, and $w_1(n_i^k, n_j^{k+1})$ and $w_2(n_i^k, n_j^{k+1})$ are the corresponding weighted detouring distance associated with RP $k$ and $k+1$ (assuming that RP $k$ is established at $n_i^k$ and RP $k+1$ is established at $n_j^{k+1}$) in state 1 and state 2, respectively. $w_1(n_i^k, n_j^{k+1})$ and $w_2(n_i^k, n_j^{k+1})$ can be computed using the formula that we proposed in the proposal. We just need to keep in mind that we should replace the original discharging range with the new discharging range when we compute $w_2(n_i^k, n_j^{k+1})$.

To find the optimal set of RP locations, it is equivalent to find the shortest path from the source node to the sink node in the constructed network. Below is the constructed network of this problem.

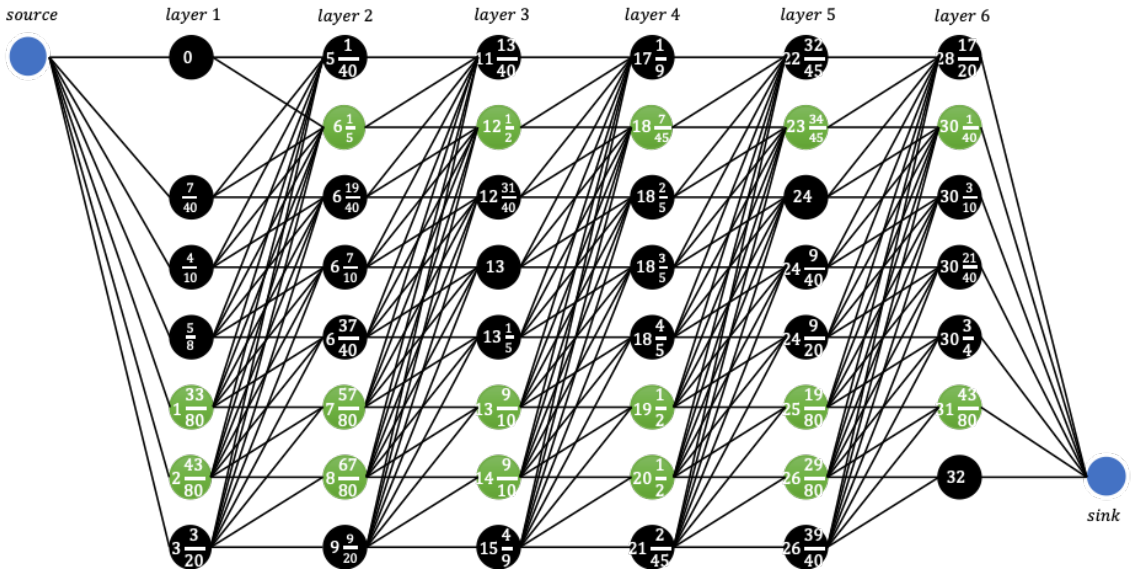



**Figure 4.3** The constructed network for solving the SPP

## 4.4. Conclusion

In this chapter, we considered the location problem on a class of probabilistic networks for which the values of the travel attributes associated with the links are random. Again, we are able to convert the original problem to a shortest path problem and solve the problem in polynomial times.

# CHAPTER 5

# THE GENERAL TREE PROBLEM

## 5.1. Overview

While the US national highway system consists of a set of circuit networks, if it is partitioned into local highway systems by operating authority, then many of them form trees or tree-like networks. Also, note that road networks in sparsely settled areas are generally trees, since tree road networks are the cheapest to construct.

Trees are central to the structural understanding of networks and graphs and often occur with additional attributes such as roots and vertex-ordering. They have a wide range of applications, including data storage, searching, information processing, and facility location. Because it is easier to get insights into tree network problems, numerous articles in classical facility location problems on transportation networks without cycles are available in the literature.

## 5.2. Problem on caterpillars and stars

In our previous discussion in Chapter 3, where the degree of any junction node, which is on the central path, is at most three, now let us consider the same problem on a caterpillar tree and on a star.

### 5.2.1. Definition of caterpillar and star (graph theory)

Caterpillar

In graph theory, a caterpillar tree is a tree in which all the vertices are within distance 1 of a central path. Caterpillars were first studied in a series of

papers by Harary and Schwenk. The name was suggested by A. Hobbs. As Harary and Schwenk (1973) colorfully write, "A caterpillar is a tree which metamorphoses into a path when its cocoon of endpoints is removed." Some equivalent characterizations are as follows: (1) They are the trees for which removing the leaves and incident edges produces a path graph; (2) They are the trees in which there exists a path that contains every vertex of degree two or more; (3) They are the trees in which every vertex of degree at least three has at most two non-leaf neighbors.

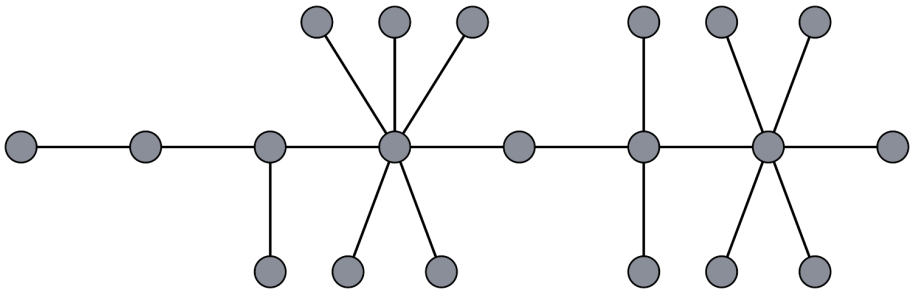


**Figure 5.1** An example of a caterpillar tree

Star

A start $S_k$ is the complete bipartite graph $K_{1,k}$: a tree with one internal node and $k$ leaves (but, no internal nods and $k+1$ leaves when $k \leq 1$). Alternatively, some authors define $S_k$ to be the tree of order $k$ with maximum diameter 2; in which case a star of $k > 2$ has $k-1$ leaves.

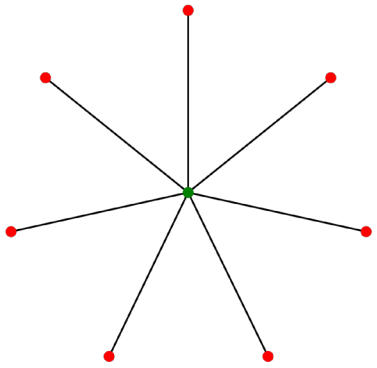


**Figure 5.2** An example of a star

### 5.3. Problem on general trees

Now, we address the continuous deviation-flow location problem on a tree network. Let $T(V, E)$ be an undirected tree network consisting of a set $V$ with $n$ vertices and a set $E$ with $n-1$ edges, where $n \geq 2$; otherwise, the network is trivial. An edge $(v_i, v_j) \in E$ is defined if $v_i \in V$ and $v_j \in V$ are directedly connected. We also denote $P(v_i, v_j)$ as the unique simple path between $v_i$ and $v_j$ for $i < j$, for all $v_i, v_j \in V$. Let $L$ is defined as the set of all possible paths in $T$; that is, $L = \{P(v_i, v_j) \mid i < j, for\ all\ v_i, v_j \in V\}$. The average traffic flow along $P(v_i, v_j)$ is denoted as $f(v_i, v_j)$. The length of $P(v_i, v_j)$ is denoted as $d(v_i, v_j)$, and $d(v_i, v_j) = d(v_j, v_i)$. Similarly, $P(v_i, x)$ denotes the unique simple path between $v_i \in V$ and any point $x \in V$, and the length of this path is denoted as $d(v_i, x) = d(x, v_i)$.

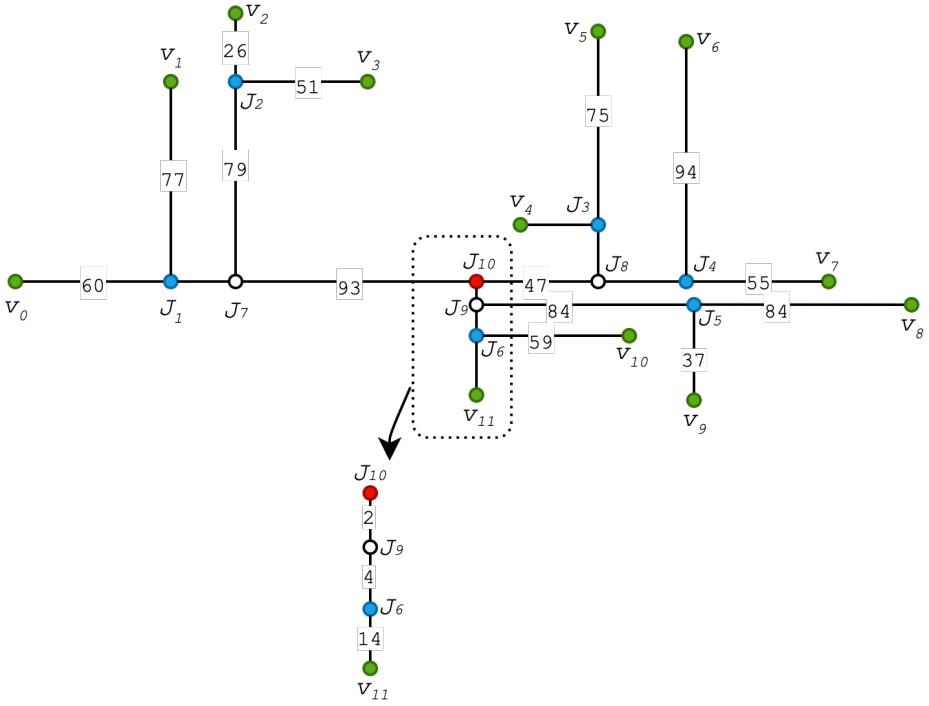



**Figure 5.3** An example of an undirected tree network

We denote a vertex as a central vertex if its greatest distance from any other vertex is as small as possible. Here, the distance between a pair of vertices is defined as the minimum number of arcs between them. For the example tree network in Figure 5.3, the central vertex is the red vertex $J_{10}$.

***Proposition 5.1*** A tree has either one single central vertex, or two adjacent central vertices.

*Proof of Proposition 5.1* First observe that in a tree, if $v_i$ is some vertex and $v_j$ is at maximal distance from $v_i$, then $v_j$ must be a leaf, because otherwise there is another vertex further away from $v_i$. Therefore, if we remove all leaves at once, all greatest distances are reduced by 1, and the set of central vertices remains the same. We can repeatedly remove all leaves until this is no longer possible, which

must be because we are left with a single degree 0 vertex, or we are left with no vertices. In the first case, the degree 0 vertex is central, and it must have been the only central vertex in each previous step, including in the original tree. In the second case, we must have had a single edge in the previous step, whose vertices must have been the central ones in the original tree.

Then, for any tree $T(V, E)$ with $|V| > 2$, we repeatedly remove all leaves until this is no longer possible. In the previous step, we must either have a single central vertex, or a single edge connecting the two central vertices, and in the penultimate step, we must either have a star, or a caterpillar.

### 5.3.1. Minimum number of RPs needed

To determine the minimum number of RPs that are required to serve all O-D transportation needs, we introduce a two-step algorithm.

#### Step One --- Inward Searching Procedure

*Step 1.*The initial tree trimming. Examine each leaf vertex $v$ in the tree network, if the length of the arc which connects the leaf with its parent is greater than or equal to the driving range $r$, iteratively add RP locations onto the arc, then cut the tree.

In Chapter 3, we also proposed a two-step algorithm in order for finding the minimum number of RPs, where the first step is comb tree trimming, and the second step is a rightward searching pass followed by a leftward searching pass. Recall that in the trimming procedure, we systematically examine each comb tooth $(v, J)$, iteratively add RP locations at distance $i * r$ from the leaf vertex $v$ if the length of the tooth $d(v, J)$ is greater than or equal to $r$, where $i = 1, \dots, \left\lfloor \frac{d(v,J)}{r} \right\rfloor \times r$, and then cut the comb. While, we will do the same thing in the very first step. For each leaf vertex

$v$ in tree $T(V, E)$, let $parent(v)$ denote the parent vertex of $v$. If the arc length $d\big(v, parent(v)\big)$ is greater than or equal to $r$, add a set of RP locations $\left\{\mathcal{P}_1, \cdots, \mathcal{P}_{\left\lfloor \frac{d(v,parent(v))}{r} \right\rfloor}\right\}$ to the arc, where the $i^{th}$ RP is at distance $i * r$ from the leaf vertex $v$.

In the example of the tree network in Figure 5.3, if we let driving range $r$ equal to 57, then there are six arcs with an arc length greater than $r$ but smaller than $2r$: $(v_0, J_1)$, $(v_1, J_1)$, $(v_5, J_3)$, $(v_6, J_4)$, $(v_8, J_5)$ and $(v_{10}, J_6)$. Thus, onto each of these arcs, we should add one RP location.

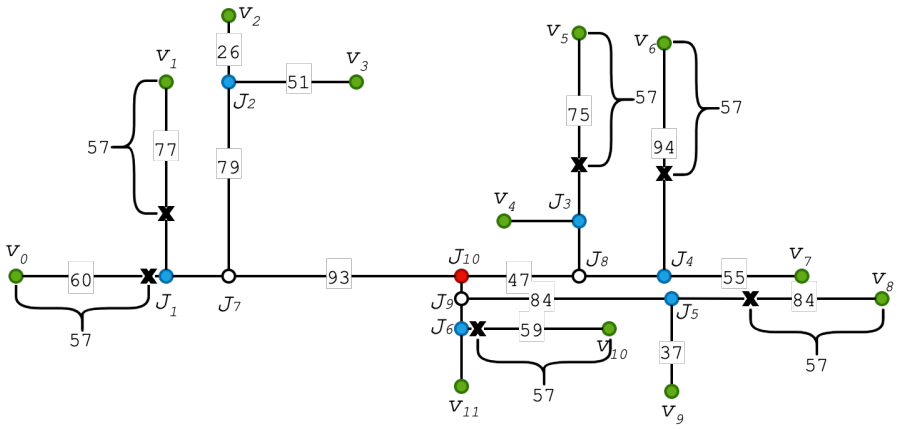



**Figure 5.4** Added RP locations

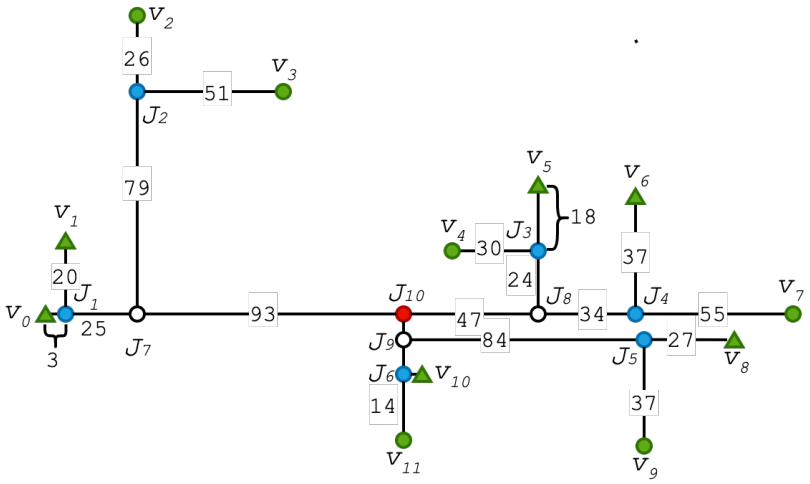



**Figure 5.5** After the initial trimming

In order to further trim the tree network, we first examine the minimum remaining travel distance of vehicles at a junction vertex $J$, which is adjacent to only one junction vertex ($\neq J$) besides the leaves. The minimum remaining travel distance of vehicles at such a junction vertex $J$ is denoted as $re(J)$ and computed as follows:

$$re(J) = r - \max\left\{\max\left\{\begin{array}{c} d(u,J) \mid u \text{ is a leaf vertex adjacent to } J, \\ \text{but NOT able to reach any other leaf vertex with an RP} \end{array}\right\}, \min\{d(v,J) \mid v \text{ is a leaf vertex adjacent to } J \text{ and with an RP}\}\right\}. \quad (5.1)$$

The intuition behind $re(J)$ is that $re(J)$ tells the minimum remaining travel distance at junction vertex $J$ that vehicles can drive up, when they enter the network at any leaf vertex that is a neighbor of $J$. The minimum remaining travel distance, $re(J)$, is calculated by subtracting the maximum value between $\max_u\{d(u,J)\}$ and $\min_v\{d(v,J)\}$ from driving range $r$, where $u$ is any leaf vertex (with or without an RP) adjacent to $J$ but not able to reach other leaf vertices that are equipped with RPs, and where $v$ is any leaf vertex (with an RP) adjacent to $J$, implying that the leaf $v$

can become a refueling service hub for vehicles entering the network at other leaves adjacent to $J$ before continuing to $J$, if possible.

In the example of the trimmed tree network in Figure 5.5, $re(J_1) = 54$, $re(J_2) = 6$, $re(J_3) = 39$, $re(J_4) = 2$, $re(J_5) = 20$ and $re(J_6) = 55$. Specially, $re(J_1)$ is computed as follows. Junction vertex $J_1$ has two leaf neighbors: $v_0$ and $v_1$, where each of them is equipped with an $RP$, and vehicles entering the network at either $v_0$ and $v_1$ can drive up to $J_1$ directly with a fully charged battery. Therefore, $re(J_1) = r - \min\{d(v_0, J_1), d(v_1, J_1)\} = 57 - \min\{3,20\} = 54$.

In addition to the minimum travel distance of each junction $J$, we shall also store the location information of $J'$s nearest RP if there is any within driving range $r$, that is, we shall store the minimum travel distance from $J$ to an RP. Consider the following situation in the example tree network in Figure 5.5, if vehicles entering the tree network at junction $J_7$ and heading to junction $J_{10}$ with some level of remaining charge, this information – the minimum travel distance from $J_1$ to an RP – can be used to determine that whether $J_1$ could become a refueling detour routing point for these vehicles or not.

*Step 2.* The main part of tree inward searching. Repeat Step 2.1 - Step 2.3, until we are left with a star network, in which $re(v) \leq d(v, J)$ for any leaf $v$ in the star.

*Step 2.1.* For each junction vertex $J$ which has leaf vertices as its neighbors, we compute the minimum remaining travel distance of $J$ when vehicles enter the network at any of the leaf neighbors. Additionally, we store the minimum travel distance from $J$ to an RP that has been added to the tree network in previous steps, if none exists, we set this distance equal to $+\infty$.

*Step 2.2.* Cut off leave vertices.

After step 2.2, the junction vertices in step 2.1 now become leaf vertices.

*Step 2.3.* According to these remaining travel distances computed in step 2.1, we then determine the set of RPs that are needed to be located along the arc which connects the new leaf vertex and its parent vertex.

In Step 2.1, we compute the minimum remaining travel distance of a junction vertex $J$ as follows:

$$re(J) = \min_{v:v \text{ is a leaf adjacent to } J} \left\{ \max \left\{ re(v) - d(v,J), \max_{\substack{u:u \neq v \text{ is a leaf adjacent to } J, \\ \text{and } u \text{ is a refueling detour routing point for } v}} \{re(u) - d(u,J)\} \right\} \right\}, \tag{5.2}$$

where $re(v) - d(v,J)$ measures the level of remaining charge at $J$ when detouring is not an option, and $\max_{\substack{u:u \neq v \text{ is a leaf adjacent to } J, \\ \text{and } u \text{ is a refueling detour routing point for } v}} \{re(u) - d(u,J)\}$ measures the maximum level of remaining charge at $J$ when detouring is considered. Equation (5.1) is a special case of equation (5.2).

Let us further describe the Step 2.3. Consider the tree network in Figure 5.6, if $re(v_0) > d(v_0,J)$, no RP location is needed to be added onto arc $(v_0,J)$, whereas if $re(v_0) < d(v_0,J)$, we shall locate RPs sequentially at distance $re(v_0)$, $re(v_0) + r$, …, $re(v_0) + \left\lfloor \frac{d(v_0,J) - re(v_0)}{r} \right\rfloor \times r$ away from $v_0$. If any RP has been added onto the arc $(v_0,J)$, we turn the RP location at distance $r - \left( re(v_0) + \left\lfloor \frac{d(v_0,J) - re(v_0)}{r} \right\rfloor \times r \right)$ from $J$ into a new leaf vertex (let us continue to use $v_0$), cut off the tree, update $re(v_0) = r$ and $d(v_0, RP_{nearest}) = 0$.

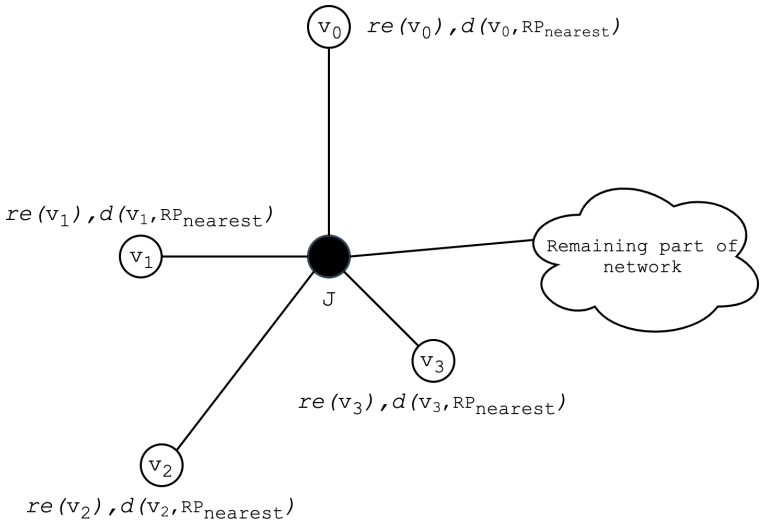



**Figure 5.6** A portion of the tree network

As we mentioned earlier, any tree network has either one single central vertex, or two adjacent central vertices, where a central vertex is a vertex such that its greatest distance from any other vertex is as small as possible. Thus, after iterations of trimming, we should be left with a star network or a single arc connecting two vertices which is also a star, as shown in Figure 5.7.

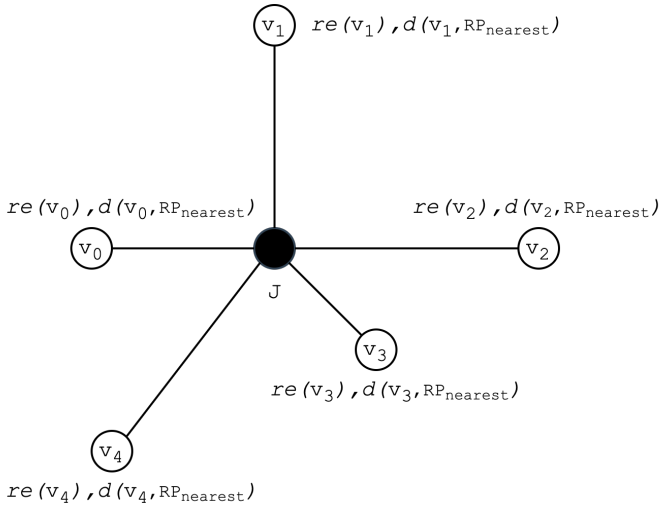



**Figure 5.7** The star network we are left with

Illustrating Step 2 for the network in Figure 5.5

Input data: tree network in Figure 5.5, where vertices $v_0$, $v_1$, $v_5$, $v_6$, $v_8$ and $v_{10}$ are equipped with RPs, and $re(v_i) = r = 57$ for $\forall i = 0,1,\dots,11$.

Perform Step 2.1 using equation (5.1):

$re(J_1) = r - \max\{\min\{d(v_0, J_1), d(v_1, J_1)\}\} = 57 - \min\{3, 20\} = 54$, $d(J_1, RP_{nearest}) = 3$;

$re(J_2) = r - \max\{\max\{d(v_2, J_2), d(v_3, J_2)\}\} = 57 - \max\{26, 51\} = 6$, $d(J_2, RP_{nearest}) = +\infty$;

$re(J_3) = r - \max\{\min\{d(v_5, J_3)\}\} = 57 - 18 = 39$, $d(J_3, RP_{nearest}) = 18$;

$re(J_4) = r - \max\{\max\{d(v_7, J_4)\}, \min\{d(v_6, J_4)\}\} = 57 - \max\{55, 37\} = 2$, $d(J_4, RP_{nearest}) = 37$;

$re(J_5) = r - max\{max\{d(v_9, J_5)\}, min\{d(v_8, J_5)\}\} = 57 - max\{37, 27\} = 20$, $d(J_5, RP_{nearest}) = 27$;

$re(J_6) = r - \max\{\min\{d(v_{10}, J_6)\}\} = 57 - 2 = 55$, $d(J_6, RP_{nearest}) = 2$.

Perform Step 2.2:

We cut off the leaf vertices in $\{v_i \mid i = 0, 1, \dots, 11\}$. The tree network that we get after trimming is shown in Figure 5.8.

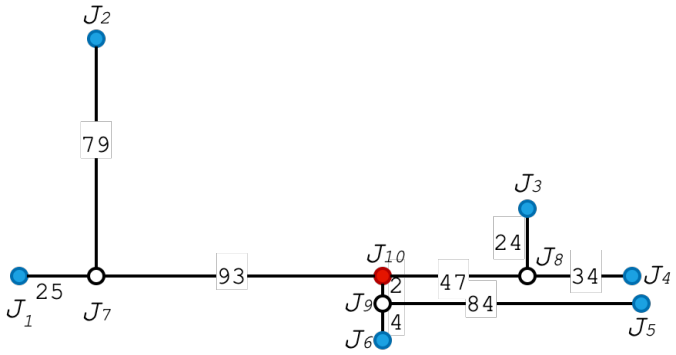


**Figure 5.8** The product after performing Step 2.2

Perform Step 2.3:

Input date: tree network in Figure 5.7

$re(J_1) = 54$, $d(J_1, RP_{nearest}) = 3$; $re(J_2) = 6$, $d(J_2, RP_{nearest}) = +\infty$;

$re(J_3) = 39$, $d(J_3, RP_{nearest}) = 18$; $re(J_4) = 2$, $d(J_4, RP_{nearest}) = 37$;

$re(J_5) = 20$, $d(J_5, RP_{nearest}) = 27$; $re(J_6) = 55$, $d(J_6, RP_{nearest}) = 2$.

For arc $(J_1, J_7)$: $d(J_1, J_7) < re(J_1)$, no RP is needed.

For arc $(J_2, J_7)$: $d(J_2, J_7) > re(J_2)$, add one RP at distance 6 from $J_7$, update $re(J_2) = 57$ and $d(J_2, J_7) = 73$; add another RP at distance 57 from $J_2$, update $re(J_2) = 57$ and $d(J_2, J_7) = 16$.

For arc $(J_3, J_8)$: $d(J_3, J_8) < re(J_3)$, no RP is needed.

For arc $(J_4, J_8)$: $d(J_4, J_8) > re(J_4)$, add one RP at distance 2 from $J_4$, update $re(J_4) = 57$ and $d(J_4, J_8) = 32$.

For arc $(J_5, J_9)$: $d(J_5, J_9) > re(J_5)$, add one RP at distance 20 from $J_5$, update $re(J_5) = 57$ and $d(J_5, J_9) = 64$; add another RP at distance 57 from $J_5$, update $re(J_5) = 57$ and $d(J_5, J_9) = 7$.

For arc $(J_6, J_9)$: $d(J_6, J_9) < re(J_6)$, no RP is needed.

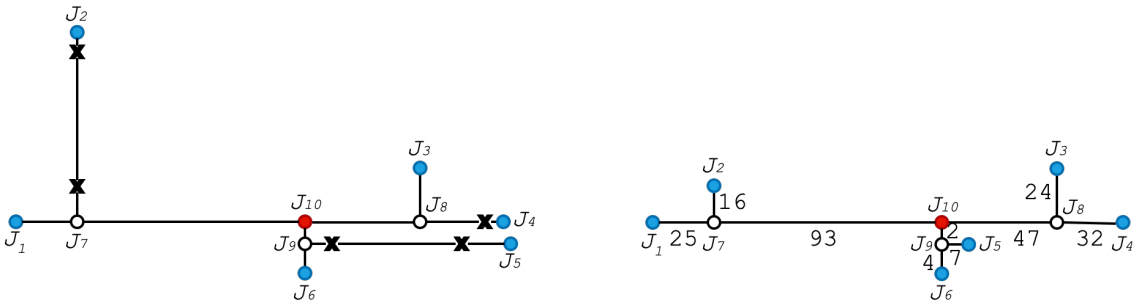



**Figure 5.9** One iteration of Step 2.3

Anther iteration of Step 2:

Perform Step 2.1:

Input date: tree network (on the left) in Figure 5.9

$re(J_1) = 54,\ d(J_1, RP_{nearest}) = 3;\ re(J_2) = 57, d(J_2, RP_{nearest}) = 0;$

$re(J_3) = 39, d(J_3, RP_{nearest}) = 18;\ re(J_4) = 57, d(J_4, RP_{nearest}) = 0;$

$re(J_5) = 57, d(J_5, RP_{nearest}) = 0;\ re(J_6) = 55, d(J_6, RP_{nearest}) = 2.$

$re(J_7) = r - d(J_2, J_7) = 41,\ d(J_7, RP_{nearest}) = 16;$

$re(J_8) = \min\{re(J_3) - d(J_3, J_8), r - d(J_4, J_8)\} = 15, d(J_8, RP_{nearest}) = 32;$

$re(J_9) = re(J_6) - d(v_6, J_9) = 57 - 6 = 51, d(J_9, RP_{nearest}) = 6.$

Perform Step 2.2:

We cut off the leaf vertices in $\{J_i \mid i = 1,2, \ldots,6\}$. The tree network that we get after trimming is shown in Figure 5.9.

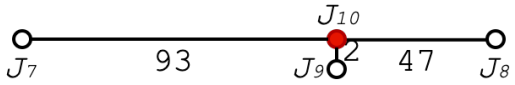



**Figure 5.10** The product after performing Step 2.2

Perform Step 2.3:

Input date: tree network in Figure 5.10

$re(J_7) = 41$ , $d(J_7, RP_{nearest}) = 16$ ; $re(J_8) = 15$ , $d(J_8, RP_{nearest}) = 32$ ; $re(J_9) = 51$ , $d(J_9, RP_{nearest}) = 6$.

For arc $(J_7, J_{10})$: $d(J_7, J_{10}) > re(J_7)$, add one RP at distance 41 from $J_7$, update $re(J_7) = 57$ and $d(J_7, J_{10}) = 52$.

For arc $(J_8, J_{10})$: $d(J_8, J_{10}) > re(J_{10})$, add one RP at distance 15 from $J_8$, update $re(J_8) = 57$ and $d(J_8, J_{10}) = 32$.

For arc $(J_9, J_{10})$: $d(J_9, J_{10}) < re(J_{10})$, no RP is needed.

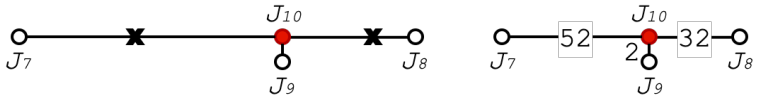



**Figure 5.11** The resulting tree network after another iteration of Step 2

It is also worth to mention that at most one RP will be needed for the star network we are left with after Step 2.

***Proposition 5.2*** Given any tree network, after performing the Steps 1 and 2, we are left with a star network, and at most one RP will be needed for this star.

Next, we present a procedure for finding the localization tree for the star network, if one is needed.

*Step 3.* Determine the localization tree for the star. If no more RP is needed, terminate the inward searching algorithm; otherwise, determine the localization subtree for this star using the following steps.

First, we describe the condition where no more RP is needed. Given a simple path $P(v_i, v_j)$, when no detouring option is available, the trip from vertices $v_i$ to $v_j$ is feasible if $re(v_i) - \left(d(v_i, J) + d(J, v_j)\right) \geq r - re(v_j)$, that is, vehicles entering the star network at $v_i$ with the minimum remaining travel distance $re(v_i)$ is able to drive up to $v_j$ directly while keeping the minimum level of recharge at $r - re(v_j)$ when leave the star network at $v_j$, implying that vehicles can further drive up to any vertex adjacent to $v_j$.

Whereas when the detouring option is available for the trip from $v_i$ to $v_j$, the trip is feasible if $v_i \rightsquigarrow v_k \mapsto v_j$, where $v_k \notin \{v_i, v_j\}$ is a neighbor of $J$. By "$v_i \rightsquigarrow v_k$", we mean that there is a feasible refueling walk from $v_i$ to $v_k$ and there is a feasible simple path from $v_k$ to $v_j$, while maintaining the minimum level of charge at $r - re(v_j)$ when vehicles leave the star at $v_j$. There are three different scenarios for $v_i \rightsquigarrow v_k$:

a) A feasible simple path from $v_i$ to $v_k$ exists, and $v_k$ has an RP. That is, $re(v_i) - \left(d(v_i, J) + d(J, v_k)\right) \geq 0$ and $r - \left(d(v_k, J) + d(J, v_j)\right) \geq r - re(v_j)$;

b) A feasible simple path from $v_i$ to $v_k$ exists, but $v_k$ does not have an RP, implying that after arriving at $v_k$, vehicles can further drive up to $v_k$'s nearest RP, get

refueled and continue to $v_j$. That is, $re(v_i) \geq d(v_i, J) + d(J, v_k) + d(v_k, RP_{nearest})$ and $\left(r - d(v_k, RP_{nearest})\right) - \left(d(v_k, J) + d(J, v_j)\right) \geq r - re(v_j)$;

c) No feasible simple path from $v_i$ to $v_k$ exists, but we have $v_i \rightsquigarrow v_l \mapsto v_k \mapsto v_j$.

Consider the star network $S_3$ in Figure 5.11, where $re(J_7) = 57$, $d(J_7, RP_{nearest}) = 0$, $d(J_7, J_{10}) = 52$; $re(J_8) = 57$, $d(J_8, RP_{nearest}) = 0$, $d(J_8, J_{10}) = 32$; and $re(J_9) = 51$, $d(J_9, RP_{nearest}) = 6$, $d(J_9, J_{10}) = 2$. Let us take $P(J_7, J_9)$ and $P(J_8, J_9)$ for example.

For $P(J_7, J_9)$: the simple path is infeasible; $J_9$ is not a feasible refueling detour routing point, since $re(J_7) \leq d(J_7, J_{10}) + d(J_{10}, J_9) + d(J_9, RP_{nearest})$. Thus, we can conclude that one more RP is needed.

For $P(J_8, J_9)$: the simple path is feasible.

If we find that one more RP is needed for the star network, how do we identify the corresponding localization tree?

Denote the star network as $S_k$, where $S_k$ is a complete bipartite graph $K_{1,k}$, i.e., a tree with one junction vertex and $k$ leaves, however, when $k \leq 1$ there is no junction vertex.

In order to obtain the localization tree for $S_k$, the first step we need to do is, for each simple path $P(v_i, v_j)$ where $v_i, v_j \in V(S_k)$, to identify the localization segment that contains all RP locations covering the trip from $v_i$ to $v_j$ when detouring option is not considered. Let $LS(v_i, v_j)$ be the localization segment of $P(v_i, v_j)$. Since vehicles entering $S_k$ at $v_i$ have a minimum remaining travel distance $re(v_i)$, points in $LS(v_i, v_j)$ must be within distance $re(v_i)$ from $v_i$. Since vehicles entering $S_k$ at $v_j$ have a minimum remaining travel distance $re(v_j)$, which implies that the level of

remaining charge at $v_j$ when vehicles leave $S_k$ should be at least $r - re(v_j)$. Then points in $LS(v_i, v_j)$ must be within distance $re(v_j)$ from $v_j$. That is,

$$LS(v_i, v_j) = \{x \in P(v_i, v_j) \mid d(v_i, x) \leq re(v_i), d(x, v_j) \leq re(v_j)\}. \quad (5.3)$$

If $LS(v_i, v_j)$ contains either vertex $v_i$ or $v_j$, we may expand the localization segment on path $P(v_i, v_j)$.

Next, we include the detouring option when finding the localization tree for $P(v_i, v_j)$. In this subsection, we use the terms of "Cycle Starting Vertex" and "Cycle Returning Point" defined in Kweon et. al (2017).

Cycle Starting Vertex

A vertex is called a cycle starting vertex if a symmetric cycle begins its deviation from a simple path at this vertex. A cycle starting vertex is the only vertex in common between the simple path and the symmetric cycle. We denote a cycle starting vertex as $v_{CSV}$.

In order to identify a cycle starting vertex of a simple path $P(v_i, v_j)$, we need to examine the maximum allowable detouring distance of vehicles at a vertex within $P(v_i, v_j)$, as well as its degree. Note that in a star, the junction vertex is the only vertex that can be a cycle starting vertex.

Denote the maximum allowable detouring distance of vehicles at a cycle starting vertex $v_{CSV}$ as $\delta(v_{CSV}|v_i, v_j)$ and we can compute it as follows:

$$\delta(v_{CSV}|v_i, v_j) = \min \begin{Bmatrix} \max\left\{re(v_i) - d(v_i, v_{CSV}), \max_{v_k}\{re_{v_i \twoheadrightarrow v_k}(v_{CSV})\}\right\}, \\ re(v_j) - d(v_{CSV}, v_j) \end{Bmatrix}. \quad (5.4)$$

Intuitively, $\delta(v_{CSV}|v_i, v_j)$ measures the maximum allowable outbound detouring distance at $v_{CSV}$ that vehicles can drive up when they enter the network at $v_i$ and

leave at $v_j$, where the $P(v_i, v_j)$ is a portion of some one-way trip in the original tree network. Any point within a symmetric cycle must be reachable from $v_i$ (with or without detouring) and be able to reach $v_j$. The value of $\delta(v_{CSV}|v_i, v_j)$ must be positive for vehicles to start a symmetric cycle originating at $v_{CSV}$.

Next, we examine the degree of a cycle starting vertex $v_{CSV}$. Denote the degree of $v_{CSV}$ as $deg(v_{CSV})$. For path $P(v_i, v_j)$, $v_{CSV}$ can only be a cycle starting vertex if we have $deg(v_{CSV}) \geq 3$. That is, $v_{CSV}$ has at least one more adjacent arc other than $(v_{CSV}, v_i)$ and $(v_{CSV}, v_j)$, and a portion of this arc or the entire arc can form a sub-path for a symmetric cycle originating at $v_{CSV}$.

Cycle Returning Point

For each symmetric cycle originating at $v_{CSV}$, we need to identify the farthest point that vehicles can reach before returning to $v_{CSV}$, which is not necessarily a network vertex. We call this point a cycle returning point and denote it as $r_{CRP}$. Given that the purpose of starting a symmetric cycle is to refuel vehicles, then the cycle returning point can be regarded as the farthest feasible candidate site for the RP.

Compared to the cycle starting vertex, the cycle returning point belongs to the symmetric cycle only but does not belong to the simple path. The cycle starting vertex is always a vertex on the original tree network, while the cycle returning point can be an interior point on an arc or can be a vertex on the original network.

According to the topology of the tree network, several cycle returning points may arise from one cycle starting vertex, implying that multiple symmetric cycles can start at the same cycle starting vertex. Given a cycle starting vertex $v_{CSV}$ of

$P(v_i, v_j)$, denote $CRP(v_{CSV}|v_i, v_j)$ as the set of cycle returning points arising from $v_{CSV}$. The location of a cycle returning point, $r_{CRP} \in CRP(v_{CSV}|v_i, v_j)$, is determined by comparing the value of $\delta(v_{CSV}|v_i, v_j)$ to the length of the arc originating at $v_{CSV}$:

a) If the value of $\delta(v_{CSV}|v_i, v_j)$ is less than the length of the arc, then the cycle returning point is located at a distance $\delta(v_{CSV}|v_i, v_j)$ from $v_{CSV}$, since vehicles at $v_{CSV}$ can only drive up to this distance before getting recharged.

b) If the value of $\delta(v_{CSV}|v_i, v_j)$ is greater than or equal to the length of the arc, we further consider the following three cases:

b1) If the end vertex of this arc is a leaf vertex in the original tree network, then the cycle returning point is located at that end vertex;

b2) If the end vertex of this arc is literally a vertex of degree greater than one in the original tree network, then the cycle returning point can be located on the arcs adjacent to that end vertex;

b3) If the end vertex of this arc is an interior point in the original tree network, then the cycle returning point can be located on the arc which contains this sub-arc in the original tree network, or it can be located on the arcs incident to the arc which contains this sub-arc in the original tree network.

In either case b2) or b3), we shall expand the current star network by adding these reachable arcs back into the star and determine the cycle returning points.

Next, we describe an algorithm to identify the locations of all cycle returning points in $CRP(v_{CSV}|v_i, v_j)$ for a given cycle starting vertex $v_{CSV}$ of path $P(v_i, v_j)$. The algorithm explores the arcs along separate sub-paths starting at $v_{CSV}$, compute the

minimum remaining travel distance at each reachable vertex, expand the star network if necessary, until all cycle returning points are identified.

**Algorithm (Cycle Returning Point Detection Algorithm)**

Given $P(v_i, v_j)$ and the cycle starting vertex $v_{CSV}$ on it:

---

*Step 1.* Compute $\delta(v_{CSV}|v_i, v_j)$.

*Step 2.* Initialize an empty first-in first-out queue $Q$. Place $v_{CSV}$ at the tail of queue $Q$.

*Step 3.* Repeat the following sub-steps until there is no remaining vertex in $Q$:

*Step 3.1.* Select vertex $v$ at the head of $Q$, pop it up from $Q$.

*Step 3.2.* Determine the set of child vertices of $v$ in the original tree network.

*Step 3.3.* For each $child(v)$, perform the following steps:

*Step 3.3.1.* Compute $\delta(child(v)|v_i, v_j) = \delta(v|v_i, v_j) - d(v, child(v))$;

*Step 3.3.2.* Perform one of the following three procedures:

a) If $\delta(child(v)|v_i, v_j) < 0$: $r_{CRP}$ is located on the arc $(v, child(v))$ at a distance $\delta(v|v_i, v_j)$ from $v$. Add $r_{CRP}$ into set $CRP(v_{CSV}|v_i, v_j)$.

b) If $\delta(child(v)|v_i, v_j) = 0$ or if $\delta(child(v)|v_i, v_j) > 0$ and $deg(child(v)) = 1$ in the original tree network: $r_{CRP}$ is located exactly at $child(v)$. Add $r_{CRP}$ into set $CRP(v_{CSV}|v_i, v_j)$.

c) If $\delta(child(v)|v_i, v_j) > 0$ and $deg(child(v)) \geq 2$ in the original tree network: place $child(v)$ at the head of $Q$ and set the parent vertex of $child(v)$ as $v$.

---

Illustration

Again, let us take $P(J_7, J_9)$, $P(J_7, J_8)$ and $P(J_8, J_9)$ for the star network in Figure 5.11 for example. Recall that $\delta(v_{CSV}|v_i, v_j)$ is computed as

$$\delta(v_{CSV}|v_i, v_j) = \min\begin{Bmatrix} \max\left\{re(v_i) - d(v_i, v_{CSV}), \max_{v_k}\{re_{v_i \rightsquigarrow v_k}(v_{CSV})\}\right\}, \\ re(v_j) - d(v_{CSV}, v_j) \end{Bmatrix}.$$

Example 1 - Identify $CRP(J_{10}|J_7, J_9)$:

*Step 1.* $\delta(J_{10}|J_7, J_9) = \min\{re(J_7) - d(J_7, J_{10}), re(J_9) - d(J_{10}, J_9)\} = \min\{5, 49\} = 5$

*Step 2.* Initialize an empty first-in first-out queue $Q$. Enqueue $J_{10}$.

*Step 3.*

*Step 3.1.* Dequeue $J_{10}$.

*Step 3.2.* $child(J_{10}) = J_8$.

*Step 3.3.* For $J_8$, perform the following steps:

*Step 3.3.1.* Compute $\delta(J_8|J_7, J_9) = \delta(J_{10}|J_7, J_9) - d(J_{10}, J_8) = -27$;

*Step 3.3.2.* Perform procedure a) since $\delta(J_8|J_7, J_9) < 0$:

a) $r_{CRP}$ is located on the arc $(J_{10}, J_8)$ at a distance 5 from $J_{10}$. Add $r_{CRP}$ into set $CRP(J_{10}|J_7, J_9)$.

*Terminate.*

Example 2 – Identify $CRP(J_{10}|J_7, J_8)$:

*Step 1.* $\delta(J_{10}|J_7, J_8) = \min\{re(J_7) - d(J_7, J_{10}), re(J_8) - d(J_{10}, J_8)\} = \min\{5, 25\} = 5$

*Step 2.* Initialize an empty first-in first-out queue $Q$. Enqueue $J_{10}$.

*Step 3.*

*Step 3.1.* Dequeue $J_{10}$.

*Step 3.2.* $child(J_{10}) = J_9$.

*Step 3.3.* For $J_9$, perform the following steps:

*Step 3.3.1.* Compute $\delta(J_9|J_7, J_8) = \delta(J_{10}|J_7, J_8) - d(J_{10}, J_9) = 3$;

*Step 3.3.2.* Perform procedure c) since $\delta(J_9|J_7,J_8) > 0$ and $deg(J_9) \geq 2$ in the original tree network:

c) Enqueue $J_9$, set the $parent(J_9)$ as $J_{10}$.

Another iteration of *Step 3.*

*Step 3.1.* Dequeue $J_9$.

*Step 3.2.* $child(J_9) = \{J_5, J_6\}$.

*Step 3.3.* For $J_9$, perform the following steps:

For $J_5$:

*Step 3.3.1.* Compute $\delta(J_5|J_7,J_8) = \delta(J_9|J_7,J_8) - d(J_9,J_5) = -81$;

*Step 3.3.2.* Perform procedure a) since $\delta(J_5|J_7,J_8) < 0$:

a) $r_{CRP}$ is located on the arc $(J_9,J_5)$ at a distance 3 from $J_9$. Add $r_{CRP}$ into set $CRP(J_{10}|J_7,J_8)$.

For $J_6$:

*Step 3.3.1.* Compute $\delta(J_6|J_7,J_8) = \delta(J_9|J_7,J_8) - d(J_9,J_6) = -1$;

*Step 3.3.2.* Perform procedure a) since $\delta(J_6|J_7,J_8) < 0$:

a) $r_{CRP}$ is located on the arc $(J_9,J_6)$ at a distance 1 from $J_9$. Add $r_{CRP}$ into set $CRP(J_{10}|J_7,J_8)$.

*Terminate.*

Example 3 - Identify $CRP(J_{10}|J_8,J_9)$:

*Step 1.* $\delta(J_{10}|J_8,J_9) = \min\{\max\{re(J_8) - d(J_8,J_{10}), re_{J_8 \rightsquigarrow J_9}(J_{10})\}, re(J_9) - d(J_{10},J_9)\} =$

$\min\{\max\{25,49\},49\} = 49$

*Step 2.* Initialize an empty first-in first-out queue $Q$. Enqueue $J_{10}$.

*Step 3.*

*Step 3.1.* Dequeue $J_{10}$.

*Step 3.2.* $child(J_{10}) = J_7$.

*Step 3.3.* For $J_7$, perform the following steps:

*Step 3.3.1.* Compute $\delta(J_7|J_8,J_9) = \delta(J_{10}|J_8,J_9) - d(J_{10},J_7) = -3$;

*Step 3.3.2.* Perform procedure a) since $\delta(J_8|J_7,J_9) < 0$:

a) $r_{CRP}$ is located on the arc $(J_{10},J_7)$ at a distance 49 from $J_{10}$. Add $r_{CRP}$ into set $CRP(J_{10}|J_8,J_9)$.

Let us define the symmetric cycle $SC(v_{CSV}, r_{CRP}|v_i, v_j)$ as the segment that contains all RP locations that cover the detouring sub-path of $P(v_i, v_j)$ originating at cycle starting vertex $v_{CSV}$ and ending at cycle returning point $r_{CRP}$ in $CRP(v_{CSV}|v_i, v_j)$.

Now, we have found the localization tree for covering the traffic flow from $v_i$ to $v_j$. To determine the localization tree for covering the traffic flow that enters the star at any vertex $v$ and exits the star at any other vertex $u \neq v$, we need to found the corresponding localization subtree for each $P(v_i, v_j)$ and take the intersection of all of these subtrees.

**Algorithm (Localization Tree Detection Algorithm)**

---

*Step 1.* For each $P(v_i, v_j)$, determine its localization tree.

*Step 2.* Take the intersection of all the trees found in Step 1.

---

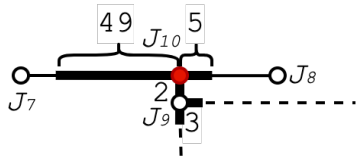



**Figure 5.12** Localization tree for the star network in Figure 5.11

Next, let us describe the outward searching procedure in order to determine all localization trees for the original tree network.

Step Two --- Outward Searching Procedure

In the last step of the inward searching procedure, we have found the localization tree for the star network. The endpoints of that localization tree indicate the boundary points defining the segments containing all RP locations. These boundary points are then used to initiate the outward searching procedure.

Recall the Breadth First Search (BFS) traversing approach used to traverse graphs. BFS is an algorithm where you should start traversing from a selected vertex (source) and traverse the graph layer wise thus exploring the neighbor vertices (i.e., vertices which are directly connected to the source). Then you must move towards the next-level neighbor vertices. As the name BFS suggests, we are required to traverse the graph breadthwise as follows: First move horizontally and visit all the vertices of the current layer; then move to the next layer.

In this subsection, we use the idea from BFS to identify the set of boundary points of all localization trees for the original tree network.

---

*Step 1.* Let $S$ be an empty set.

*Step 2.* Decompose the original tree network using the endpoints of the expanded localization tree that we have found for the star in the inward searching procedure.

*Step 3.* For each endpoint $s$ in *Step 2*, perform the following steps:

*Step 3.1.* Let $Q$ be a first-in first-out queue.

*Step 3.2.* Insert $s$ in $Q$ *(enqueue* $s$*).*

*Step 3.3.* Mark $s$ as visited.

*Step 3.4.* While $Q$ is not empty, perform the following procedures:

*Step 3.4.1.* Remove vertex $v$ from $Q$, whose neighbor will be visited now.

*Step 3.4.2.* Update the remaining travel distance at $v$, if $v$ has a child vertex $w$ such that $deg(w) = 1$ and $w$ has an RP. To update: $re(v) = \max\{re(v), \max\{r - d(v,w) | deg(w) = 1, and\ w\ has\ an\ RP\}\}$.

*Step 3.4.3.* For all neighbors $w$ of $v$ in subtree $T$:

*Step 3.4.3.1.* If $w$ has not been visited:

If $d(v,w) \geq re(w)$: systematically locate RPs at distances $re(w)$, $re(w) + r$, $re(w) + 2r$, ..., from vertex $v$; add these RP locations into set $\mathcal{S}$; compute the remaining travel distance at $w$.

Else: compute the remaining travel distance at $w$.

*Step 3.4.3.2.* Store $w$ in $Q$ to further visit its neighbor vertices.

*Step 3.4.3.3.* Mark $w$ as visited.

---

After the Inward Searching procedure and the Outward Searching procedure, we have identified the minimum number of RPs that are needed to be located on the tree network in order to serve all one-way transportation needs, and we have also identified the localization tree for each RP.

For the tree network in Figure 5.5, we found 8 location trees by applying the inward searching and outward searching procedure.

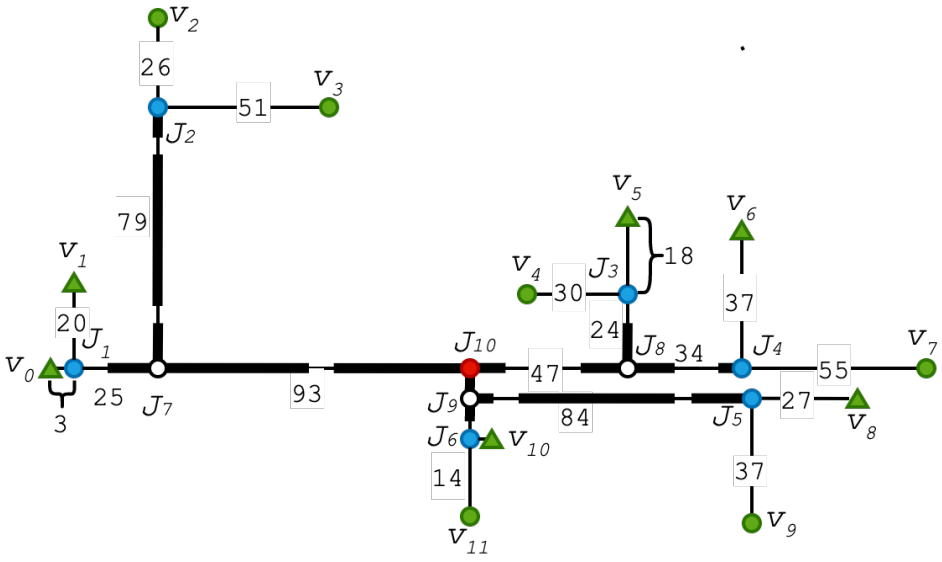



**Figure 5.13** Localization trees for the tree network in Figure 5.5

### 5.3.2. Set of breakpoints

Again, we denote $\mathcal{B}$ as the set of breakpoints, and $\mathcal{B}$ is consisting the following four parts:

- The set of endpoints of each localization tree;
- The set of internal junction vertices. A junction vertex is called an internal junction if it is within a localization tree.
- The set of XNRD sites for the points that are either internal junction vertices or boundary points of some localization trees.
- The set of RP locations such that for any two RPs in the above three subsets, with this fourth subset of RP locations we are able to guarantee that there exists a feasible refueling walk between these two RPs.

By identifying the set of breakpoints $\boldsymbol{\mathcal{B}}$, each localization segment can be further divided into several indivisible line segments, where a segment is called to be indivisible if it does not contain any breakpoint as its interior point.

Next, let us discuss some major differences between the localization segments of a comb network and the localization subtrees of a general tree network, which will lead to a difference as we identify the breakpoints.

Consider a comb network of which at least three RPs should be located on the comb span. Let $S_{i-1}$, $S_i$, $S_{i+1}$ be three consecutive localization segments, and let $\alpha \in S_{i-1}$, $\beta \in S_{i+1}$ be two points that we take from the segments $S_{i-1}$ and $S_{i+1}$. If there exists a feasible refueling walk from $\alpha$ to $\beta$, it is trivial for us to note that a portion of segment $S_i$ or the whole segment $S_i$ must belong to that refueling walk from $\alpha$ to $\beta$. Therefore, when we try to identify the set of breakpoints for the comb network, we can do it in a linear wise manner as follows: first we perform a rightward screening process, then we perform a leftward screening process. Whereas for a tree network, this may not be the case.

Therefore, when we determine the breakpoints for a tree network, we may perform several iterations before identifying all the breakpoints for some adjacent localization subtrees.

### 5.4. Solution method

For the line network problem as well as the comb tree network problem, we formulate the original problem as a shortest path problem on an acyclic network constructed on the layers of breakpoints. More specifically, this acyclic network is a multistage graph, in which the vertices can be divided into a set of stages such that

all edges from a stage to next stage only, in other words, there is no edge between vertices of the same stage and from a vertex of current stage to previous stage. Each path from source node to sink node corresponds to a feasible combination of RPs for serving all O-D pairs, and the cost of the path is equal to the total refueling detour distances with all RPs in the combination are open. By finding the shortest path in this constructed network, we also able to find the optimal solution to our original problem.

While for the general tree network problem, we are not able to construct such a multistage graph.

We then present a mixed-integer linear programming formulation.

**Formulation of the problem**

Minimize $\sum_{q\in Q} f_q \sum_H \delta_{qh} v_h$ (5.5)

Subject to $\sum_{h\in H} v_h = 1$ (5.6)

$x_k \geq v_h, \quad \forall h \in H, k \in K_h$ (5.7)

$\sum_{k\in K} x_k = m$ (5.8)

$x_k \in \{0,1\},, \quad \forall k \in K$ (5.9)

$v_h \in \{0,1\},, \quad \forall h \in H$ (5.10)

where:

$q$: a particular O-D pair

$Q$: set of all O-D pairs

$h$: a particular combination of RPs

$H$: set of all potential RP combinations

$k$: a potential RP location

$K$: set of all potential RP locations

$K_h$: set of RPs that are in combination $h$

Parameters

$f_q$: flow between O-D pair $q$

$\delta_{qh}$: minimum refueling detour distance for O-D pair $q$, if all RPs in combination $h$ are open and at least one refueling walk for the O-D pair $q$ exists

$m$: the number of RPs to be located

Decision variables

$x_k = 1$ if there is an RP at location $k$, $x_k = 0$ otherwise

$v_h = 1$ if all RPs in combination $h$ are open, $v_h = 0$ otherwise

The objective function (5.5) minimizes the total weighted refueling detour distances. Constraint (5.6) ensures exactly one combination of RPs is open. Constraint (5.7) ensures all RPs in combination $h$ are open before $v_h$ becomes one. Constraint (5.8) specify the number of RPs to be located. Constraints (5.9) and (5.10) are the integrality constraints for the binary decision variables.

Preprocessing

In this subsection, we present a preprocessing procedure that reduces the number of combinations of RPs, while preserving the optimal solution. We are hoping that this preprocessing procedure will reduce the problem size and decrease the solution time in practice.

As we have mentioned before, we probably are not able to construct a multistage graph (where stages element should be connected consecutively) on the

breakpoints in a general tree network problem. But, for the set of breakpoints of some RPs, we may construct a multistage graph.

Let $T_{i-1}$, $T_i$ and $T_{i+1}$ be such three set of breakpoints and let $T_{i+1}$ be the cluster which is much closer to the central vertex of the original tree network. Then, for each breakpoint in $T_{i+1}$, we can use dynamic programming to find the shortest path from $T_{i-1}$ to $T_{i+1}$. We can also compute the refueling detour distance as we fix RPs locations. Then, we can reduce the total number of combinations.

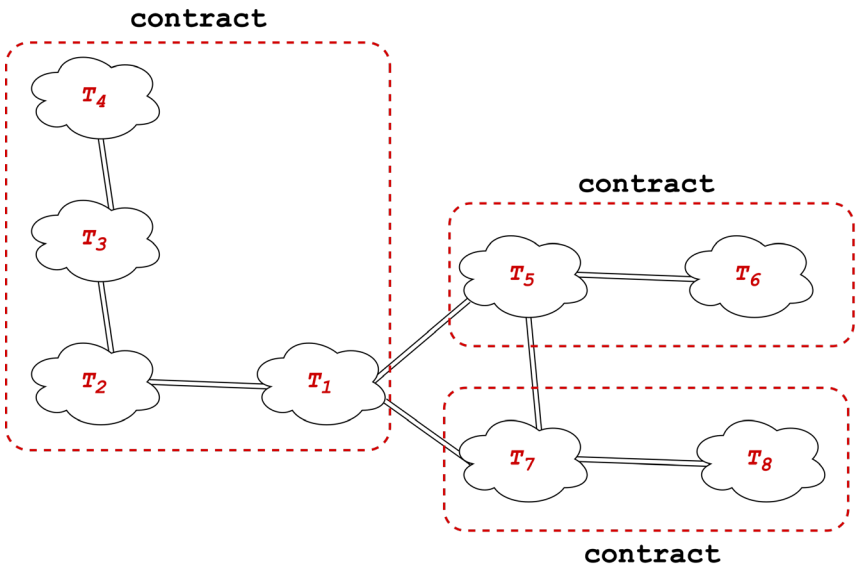



**Figure 5.14** A contraction preprocessing idea

## 5.5. Conclusion

In this chapter, we studied the continuous location problem related to locating RPs on general tree networks. To find the fewest number of RPs needed to serve all one-way O-D pairs, we proposed a 2-step greedy method --- an inward searching and an outward searching. Then, we identified the set of breakpoints for the tree network. In order for find the optimal RP locations, we formulated the

problem as a mixed integer linear programming and proposed some preprocessing steps to reduce the problem size.

Beyond the scope of current study, there are several issues worth of a further investigation. For instance, it is our interest to see if there is a way to construct a multi-stage network on the breakpoints and solve the original problem as a shortest path problem, like what we did for both line network problems and comb tree network problems. It is also worth to develop a heuristic  to solve the problem, for instance, decompose the tree network into a set of comb trees and utilize the algorithm designed for the comb tree case to solve the problem.

# CHAPTER 6

## CONCLUSIONS AND FUTURE WORK

In this dissertation several decision problems relating to infrastructure design , which arise when switching from gasoline vehicles to ones using alternative clean fuels or electricity, were formulated, solved and analyzed.

In Chapter 2, we studied the continuous location problem related to locating RPs on line networks, where finding the minimum number of RPs needed to refuel all O-D flows is considered as the first objective. Given this minimum number, our goal is to locate this number of RPs to minimize weighted sum of the travelling distance for all O-D flows. The one-way scenario is rather simple. For the round-trip scenario, an integer program with linear constraints and quartic objective function is formulated, and the problem can be solved using OPTI toolbox in Matlab. We have also identified a finite dominating set to the problem, and based on the existence of finite dominating set, the problem is formulated as a shortest path problem.

In Chapter 3, we studied the continuous location problem related to locating RPs on comb tree networks. To find the fewest number of RPs needed to serve all one-way O-D pairs, we proposed a 2-step greedy method. Then, we proposed a math programming formulation, based on which we proved the existence of a finite dominating set to the comb tree problem. Then we formulated the problem as a shortest path problem whereby the shortest path of the constructed network gives us an optimal set of RP locations.

In Chapter 4, we extended the models under the probabilistic scenario.

In Chapter 5, we studied the continuous location problem related to locating RPs on general tree networks. To find the fewest number of RPs needed to serve all one-way O-D pairs, we proposed a 2-step greedy method, called an inward search and outward search algorithm. Then, we proposed a math programming formulation, based on which we proved the existence of a finite dominating set to the tree problem. Then we formulated the problem as a mixed integer linear program for finding the optimal set of RP locations.

There are various possible areas for future work on this topic. In this dissertation, we gave an exact solution for the location problem with each underlying transportation network topology (line, comb tree and general TREE networks), however, we proposed no heuristic. Future work could consider developing a heuristic for the problem which effectively select the locations of the RPs, especially for the location problem on a general network.

We assumed that each RP can serve an infinite number of EVs (i.e., incapacitated RPs), however, this is unlikely to be the case in the real world. It will be worth initiating the study of the capacity constrained location problem for this continuous location problem.

We also assumed that the underlying network topology is deterministic, while since almost all real-world networks evolve over time, either by adding or removing nodes or links over time, an interesting and challenging research problem could be studying this continuous location problem where the population is moving in network space over time, or where the transportation network is evolving over time.

## REFERENCES


Adler, J. D., & Mirchandani, P. B. (2014). Online routing and battery reservations for electric vehicles with swappable batteries. Transportation Research Part B: Methodological, 70, 285-302.

Berman, O., Larson, R. C., & Fouska, N. (1992). Optimal location of discretionary service facilities. Transportation Science, 26(3), 201-211.

Cabral, E. A., Erkut, E., Laporte, G., & Patterson, R. A. (2007). The network design problem with relays. European Journal of Operational Research,180(2), 834-844.

Capar, I., & Kuby, M. (2012). An efficient formulation of the flow refueling location model for alternative-fuel stations. IIE Transactions, 44(8), 622-636.

Capar, I., Kuby, M., Leon, V. J., & Tsai, Y. J. (2013). An arc cover–path-cover formulation and strategic analysis of alternative-fuel station locations. European Journal of Operational Research, 227(1), 142-151.

Handler, G. Y., & Mirchandani, P. B. (1979). Location on networks: theory and algorithms (Vol. 979). Cambridge, MA: MIT press.

Hodgson, M. J. (1990). A Flow-Capturing Location-Allocation Model. Geographical Analysis, 22(3), 270-279.

Kim, J. G., & Kuby, M. (2012). The deviation-flow refueling location model for optimizing a network of refueling stations. international journal of hydrogen energy, 37(6), 5406-5420.

Konak, A. (2012). Network design problem with relays: A genetic algorithm with a path-based crossover and a set covering formulation. European Journal of Operational Research, 218(3), 829-837.

Kuby, M., & Lim, S. (2005). The flow-refueling location problem for alternative-fuel vehicles. Socio-Economic Planning Sciences, 39(2), 125-145.

Kuby, M., & Lim, S. (2007). Location of alternative-fuel stations using the flow-refueling location model and dispersion of candidate sites on arcs. Networks and Spatial Economics, 7(2), 129-152.

Lim, S., & Kuby, M. (2010). Heuristic algorithms for siting alternative-fuel stations using the flow-refueling location model. European Journal of Operational Research, 204(1), 51-61.

Mirchandani, P. B., Rebello, R., & Agnetis, A. (1995). The Inspection Station Location Problem in Hazardous Material Transportation - Some Heuristics and Bounds. Infor, 33(2), 100-113.

Upchurch, C., Kuby, M., & Lim, S. (2009). A Model for Location of Capacitated Alternative-Fuel Stations. Geographical Analysis, 41(1), 85-106.

Kweon, S. J., Hwang, S.W., & Ventura, J.A. (2017). A Continuous Deviation-Flow Location Problem for an Alternative-Fuel Refueling Station on a Tree-Like Transportation Network. Journal of Advanced Transportation. 2017. 1-20.