Tree-Shaped Facility Location Problems and the Relationship with the Knapsack Problems

Maiko SHIGENO * and Akiyoshi SHIOURA *

(June, 1995)

Abstract: The tree-shaped facility location problem aims to locate a subtree on the center of a given tree network. This paper deals with four types of the problems in which the centrality for a subtree is measured by the eccentricity or the distance-sum, and a subtree may contain partial edges or not. We investigate the relationship between these problems and knapsack-type problems. The relation leads to a linear-time algorithm for a tree-shaped facility location problem which is known to be polynomial-time solvable, or an approximation algorithm for the one which is NP-hard.

1 Introduction

The problem of locating facilities on a network has been attracting many researchers from theoretical and practical points of view. The mathematical structure of this problem varies according to the type of the underlying network, the criterion of location, the shape and the number of facilities to locate, and so on. Consequently, there are many kinds of network location problems (see [9, 3]).

In this paper, we deal with the following network location problem: locate a tree-shaped facility on the center of a given tree network. This problem is referred to as the *tree-shaped facility location problem.* To measure the centrality of a facility, we use the distance between the facility and the farthest vertex from it, or the sum of the distances between the facility and all vertices. These two criteria are called as the "eccentricity" and the "distance-sum," respectively. The tree-shaped facility location problem is initiated by Minieka [6], who provided several properties and developed polynomial-time algorithms for locating a facility with *partial edges.* In case that a facility is allowed to contain only *full edges*, Hakimi, Schmeichel, and Labbé [3] presented a simple greedy algorithm for the problem which employs the eccentricity, and induced the NP-hardness of the problem with the distance-sum criterion by reducing the partition problem.

The main aim of this paper is to investigate the relationship between tree-shaped facility location problems and knapsack-type problems. Using the relation, we present a linear-time algorithm for a location problem which is known to be polynomial-time solvable, or an approximation algorithm for the one which is NP-hard.

*Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, 2-12-1 Oh-okayama, Meguro-ku, Tokyo 152, Japan. maiko@is.titech.ac.jp, shioura@is.titech.ac.jp

The *0-1 knapsack problem* maximizes the sum of values subject to a single linear inequality constraint. The *bottleneck knapsack problem,* which may be called the minimax knapsack problem, minimizes the maximum of values, instead of the sum of values. We observe that the 0-1 and the bottleneck knapsack problems are related to the tree-shaped facility location problems with the eccentricity and the distance-sum criteria, respectively. Although the 0-1 knapsack problem is NP-hard, the bottleneck knapsack problem can be solved in linear time. Associated with partial edges, we additionally introduce the continuous relaxation of each knapsack problem. These relaxation problems are also linear-time solvable. We derive the linear-time solvability of the location problems from that of the corresponding knapsack-type ones. It is the nice mathematical structure as well as the shape of the underlying network that provides the linear-time solvability of a tree-shaped facility location problem.

This paper is organized as follows: Section 2 introduces definitions and notation, and describes four types of tree-shaped facility location problems. Section 3 explains the 0-1 knapsack problem and its variations. Section 4 discusses the relationship between tree-shaped facility location problems and knapsack-type ones.

2 Preliminaries

2.1 Tree Networks

Let $T = (V, E)$ be a tree network with a vertex set V and an edge set E. In this paper, we assume that T is drawn in a plane, where each edge $(u, v) \in E$ has a length of $l(u, v)$, and regard T as a point set. For two points p and q on one edge, a *partial edge* $[p, q]$ stands for the set of points lying between p and q, including boundary points p, q. When the points p, q respectively lie on the both end vertices of an edge, the partial edge $[p, q]$ is called a *full edge.* We denote the length of a partial edge $[p, q]$ by $l(p, q)$.

A subset S of points in T is called a *continuous subgraph* of T or a *subgraph* in short, if S is a closed set. Let $V(S)$ denote the set of points in S corresponding to vertices of V. Any subgraph S is decomposed into several partial edges such that the intersection of any pair of distinct partial edges is empty or a point in $V(S)$. We denote such a set of partial edges by $E(S)$. Note that $E(T)$ indicates the set of all full edges in T. A subgraph S is *discrete* if $E(S)$ contains only full edges and points corresponding to vertices. We call any connected subgraph S a *subtree.* In contrast with S $(\subseteq T)$, the subgraph $T \backslash S$ is constituted by the points not in S and its boundary.

In a tree T, the *distance between two points* $p, q \in T$, denoted by $d(p, q)$, is the length of the unique path connecting p and q in T. The *distance between a point* p *and a subtree* S is given by $d(p, S) = \min\{d(p, q) : q \in S\}$. The *size of a subtree* S is defined as $\mathrm{size}(S) = \sum\{l(u, v) : [u, v] \in E(S)\}$.

A *rooted tree*, denoted by T_r, is a tree T which has a distinguished point r called a *root.* When one endpoint p of an partial edge $[p, q]$ is on the path between r and the other endpoint q, then we call the point q as a *child* of p in T_r, and

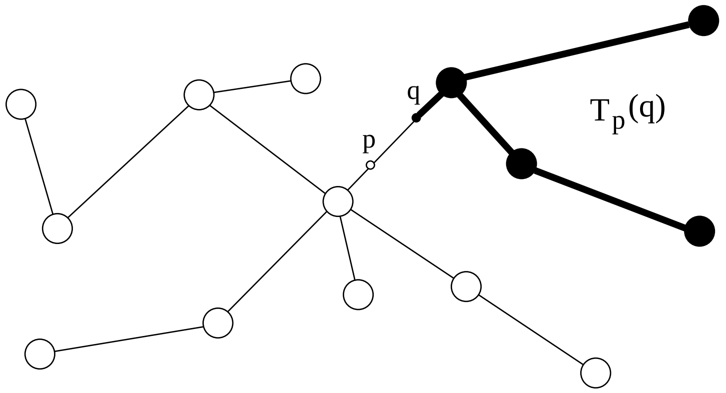

Figure 1: a subtree $T_p(q)$

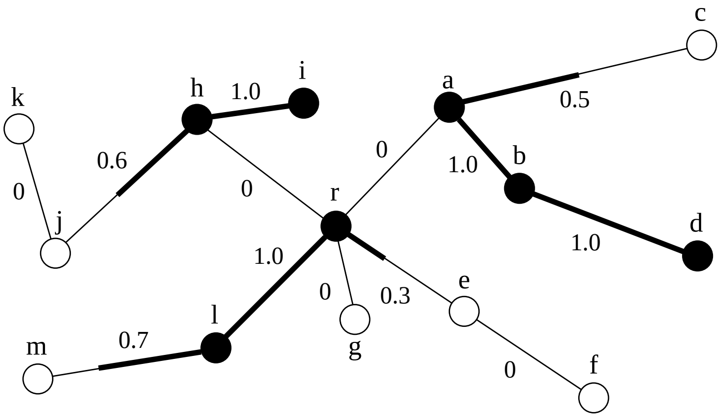

Figure 2: An example of the subgraph $S_r(\boldsymbol{x})$. Given a tree and a vector $\boldsymbol{x}$ such that $x(r,a) = 0, x(a,b) = 1.0, x(a,c) = 0.5, ...,$ the subgraph $S_r(\boldsymbol{x})$ is the one drawn by thick lines and black vertices.

denote this relation by $p \xrightarrow{r} q$. Especially, given a subtree S, if it holds that $p \xrightarrow{r} q$ for all $r \in S$, then we write $p \xrightarrow{S} q$. In the same way, if any two distinct full edges $[u,v], [v,w] \in E(T)$ satisfy $u \xrightarrow{r} v$ and $v \xrightarrow{r} w$, then $[v,w]$ is also called a *child* of $[u,v]$ in T_r, i.e., $[u,v] \xrightarrow{r} [v,w]$. When a point has no child other than itself in the rooted tree T_r, we call the point a *leaf* of T_r. Given two points $p, q \in T$, we define the subtree $T_p(q)$ as $\{x \in T : \ q$ is on the path between p and $x\,\}$ (see Figure 1).

For a point $r \in V(T)$ and a vector $\boldsymbol{x}$ with $0 \leq x(u,v) \leq 1$ for each $[u,v] \in E(T)$, we define the subgraph $S_r(\boldsymbol{x})$ of T as the point set

$$\{r\} \cup \bigcup \{\{q \in [u,v] : d(u,q) \leq l(u,v)\, x(u,v)\} : [u,v] \in E(T),\ x(u,v) > 0,\ u \xrightarrow{r} v\}.$$

Figure 2 shows the example of the subgraph $S_r(\boldsymbol{x})$. The subgraph $S_r(\boldsymbol{x})$ is a subtree if and only if it contains the path between r and u for any $[u,v] \in E(T)$ with $x(u,v) > 0$ and $u \xrightarrow{r} v$.

Lemma 2.1. *Let* r *be a point in* $V(T)$, *and* $\boldsymbol{x}$ *be any vector with* $0 \le x(u,v) \le 1$ *for all* $[u,v] \in E(T)$. *Then, the subgraph* $S_r(\boldsymbol{x})$ *is connected if and only if* $\boldsymbol{x}$ *satisfies*

$$\forall\ [u,v],[v,w] \in E(T),\ \textit{if}\ x(v,w) > 0\ \textit{and}\ [u,v] \xrightarrow{r} [v,w],\ \textit{then}\ x(u,v) = 1.$$

2.2 The Tree-Shaped Facility Location Problem

Given a tree network $T = (V,E)$ and a nonnegative number L, the *tree-shaped facility location problem* is formulated as follows:

$$\left|\begin{array}{ll} \text{Minimize} & f(S) \\ \text{subject to} & \text{size}(S) \le L, \\ & S \text{ is a subtree of } T. \end{array}\right.$$

The objective function $f(S)$ is the measure of "centrality" for a subtree S and is given by the *eccentricity*,

$$\text{ecc}(S) = \max\{d(w,S) : w \in V(T)\},$$

or the *distance-sum*,

$$\text{dis}(S) = \textstyle\sum\{d(w,S) : w \in V(T)\}.$$

The eccentricity $\text{ecc}(S)$ has the following properties.

Lemma 2.2. *Suppose* S *is a subtree of* T *with* $S \ne T$. *Then,*

$$\text{ecc}(S) = \max\{\max\{d(w,p) : w \in V(T_p(q))\} : [p,q] \in E(T \setminus S),\ p \xrightarrow{S} q\}.$$

Proof. When a partial edge $[u,v] \in E(T \setminus S)$ satisfies $u \in S$, we have $d(w,S) = d(w,u)$ for any point $w \in V(T_u(v))$. The relation

$$V(T) \setminus V(S) = \bigcup\{V(T_u(v)) : [u,v] \in E(T \setminus S),\ u \in S\}$$

implies

$$\begin{aligned} \text{ecc}(S) &= \max\{d(w,S) : w \in V(T) \setminus V(S)\} \\ &= \max\{\max\{d(w,u) : w \in V(T_u(v))\} : [u,v] \in E(T \setminus S),\ u \in S\}. \quad (2.1) \end{aligned}$$

Furthermore, for any partial edge $[p,q] \in E(T \setminus S)$ with $p \xrightarrow{S} q$ and $p \notin S$, there exists at least one partial edge $[u,v] \in E(T \setminus S)$ with $u \in S$ and $[p,q] \in E(T_u(v))$. Hence we have $\max\{d(w,u) : w \in V(T_u(v))\} > \max\{d(w,p) : w \in V(T_p(q))\}$, which, together with (2.1), establishes the lemma. ∎

Corollary 2.3. *When a subtree* S *contains the point* $r \in V(T)$ *and* $S \ne T$,

$$\text{ecc}(S) = \max\{l(p,v) + \max\{d(w,v) : w \in V(T_r(v))\} : [p,v] \in E(T \setminus S),\ p \xrightarrow{r} v\}.$$

With respect to the distance-sum $\text{dis}(S)$, we can prove the following easily.

Lemma 2.4. *Let S be a subtree and $[p, q]$ be a partial edge in $E(T \setminus S)$ with $p \in S$. Then, it holds that*

$$\mathrm{dis}(S) - \mathrm{dis}(S \cup [p, q]) = |V(T_p(q))|\, l(p, q).$$

Proof. By addition of $[p, q]$ to S, the distance between the subtree and each point in $V(T_p(q))$ becomes shorter, and the one between the subtree and each point in $V(T) \setminus V(T_p(q))$ remains unchanged. ∎

Corollary 2.5. *For any subtree S containing the point $r \in V(T)$,*

$$\mathrm{dis}(r) - \mathrm{dis}(S) = \textstyle\sum\{|V(T_p(q))|\, l(p, q) : [p, q] \in E(S),\ p \xrightarrow{r} q\}.$$

We call optimal solutions of the tree-shaped facility location problems a *tree center* and a *tree median* when the objective functions are the eccentricity and the distance-sum, respectively. By the combination of objectives and types of subtrees such as discrete and continuous, we have four kinds of problems: *discrete tree center*, *continuous tree center*, *discrete tree median*, and *continuous tree median problems*. Without loss of generality, we assume that $L < \mathrm{size}(T)$. If $L = 0$ then the tree-shaped facility location is equivalent to the point location, and optimal solutions of the above four problems are nothing but a vertex center, absolute center, vertex median, and absolute median, respectively (see [9]).

3 The 0-1 Knapsack Problem and its Variations

This section explains the 0-1 knapsack problem and three variations of it. These knapsack-type problems are related to tree-shaped facility location problems.

Suppose we are given a set of items $N = \{1, 2, \cdots, n\}$ where the i-th item has a positive profit p_i and a positive weight w_i, and a positive bound W. The *0-1 knapsack problem*, described as

$$\text{KP}\left|\begin{array}{lll} \text{Maximize} & \sum\{p_i x_i : i \in N\} & \\ \text{subject to} & \sum\{w_i x_i : i \in N\} \le W, & \\ & x_i \in \{0,\ 1\} & (i \in N), \end{array}\right.$$

is one of the most popular integer programming problems and known to be NP-hard [5]. Without loss of generality, assume that $\max\{w_i : i \in N\} \le W < \sum\{w_i : i \in N\}$. We introduce the total order $\prec$ on N such that

$$\forall\, i, j \in N, \quad i \prec j \Longleftrightarrow p_i/w_i < p_j/w_j,\ \text{or}\ p_i/w_i = p_j/w_j \text{ and } i < j.$$

The following lemma provides an approximate solution of the KP.

Lemma 3.1. [5] *Suppose that $\boldsymbol{x}^*$ is an optimal solution, and $\boldsymbol{x}^1, \boldsymbol{x}^2 \in \{0,1\}^N$ are given by*

$$x_i^1 = \begin{cases} 1 & (i \succ s), \\ 0 & (i \preceq s), \end{cases} \qquad\qquad x_i^2 = \begin{cases} 1 & (i = s), \\ 0 & (i \neq s), \end{cases}$$

where $s \in N$ *is the item with*

$$\sum\{w_i : i \in N,\ i \succ s\} \le W < \sum\{w_i : i \in N,\ i \succeq s\}. \tag{3.1}$$

Then, both $\boldsymbol{x}^1$ *and* $\boldsymbol{x}^2$ *are feasible solutions satisfying*

$$\max\{\sum\{p_i x_i^1 : i \in N\},\ \sum\{p_i x_i^2 : i \in N\}\} \ge \frac{1}{2}\sum\{p_i x_i^* : i \in N\}.$$

We call the item s with the condition (3.1) the *critical item*, which can be found by the algorithm FIND_CRITICAL_ITEM, as shown below.

algorithm FIND_CRITICAL_ITEM
begin
  $sum := 0$; $I := N$;
  while $|I| > 1$ **do begin**
    select the $\lfloor(|I|+1)/2\rfloor$-th smallest item k of I with respect to the total order $\prec$;
    $sum_1 := \sum\{w_i : i \in I,\ i \succ k\}$;
    if $sum + sum_1 \le W$ **then**
      $sum := sum + sum_1$; $I := \{i \in I : i \preceq k\}$;
    else
      $I := \{i \in I : i \succ k\}$;
    endif ;
  end ;
  let s be a unique item in I;
end .

Since the selection of the $\lfloor(|I|+1)/2\rfloor$-th smallest item of I requires only $\mathrm{O}(|I|)$ time [1], FIND_CRITICAL_ITEM finds the critical item in $\mathrm{O}(n)$ time.

The continuous relaxation of the KP is said to be the *continuous knapsack problem* (CKP). It is well-known that the CKP is solvable in $\mathrm{O}(n)$ time [5]. An optimal solution is given by

$$x_i = \begin{cases} 1 & (i \succ s), \\ (W - \sum\{w_i : i \in N,\ i \succ s\})/w_s & (i = s), \\ 0 & (i \prec s), \end{cases}$$

where s is the critical item.

We now exchange the criterion for the choice of items. The following problem is called the *bottleneck knapsack problem.*

$$\text{BKP}\left|\begin{array}{lll} \text{Minimize} & \max\{p_i x_i : i \in N\} & \\ \text{subject to} & \sum\{w_i x_i : i \in N\} \ge W, & \\ & x_i \in \{0,1\} & (i \in N). \end{array}\right.$$

In the BKP, assume that $0 < W \le \sum\{w_i : i \in N\}$. Note that the 0-1 valued vector $\boldsymbol{x}^*$ given by

$$x_i^* = \begin{cases} 1 & (p_i \le p_t), \\ 0 & (p_i > p_t) \end{cases}$$

is optimal for the BKP, where $t \in N$ is an item with

$$\sum\{w_i : i \in N,\ p_i < p_t\} < W \le \sum\{w_i : i \in N,\ p_i \le p_t\}.$$

Hence, the following lemma holds.

Lemma 3.2. *There exists an optimal solution* $\boldsymbol{x}^*$ *satisfying the condition*

$$\forall i, j \in N, \quad \textit{if } p_i > p_j \textit{ and } x_i^* = 1, \ \textit{then } x_j^* = 1. \tag{3.2}$$

Finding the item t in the similar way as the algorithm FIND_CRITICAL_ITEM, we can obtain an optimal solution satisfying the condition (3.2) in $\mathrm{O}(n)$ time.

At last, we consider the *continuous bottleneck knapsack problem* (CBKP). In addition to p_i and w_i, suppose that the i-th item has a fixed profit of q_i (≥ 0). The CBKP is formulated as follows:

$$\text{CBKP}\left|\begin{array}{lll} \text{Minimize} & \max\{p_i x_i + q_i : i \in N,\ x_i > 0\} & \\ \text{subject to} & \sum\{w_i x_i : i \in N\} \ge W, & \\ & 0 \le x_i \le 1 & (i \in N). \end{array}\right.$$

We also assume that $0 < W \le \sum\{w_i : i \in N\}$. For each $i \in N$, let us define the function $\mathrm{tw}_i : \boldsymbol{R} \to \boldsymbol{R}$ as

$$\mathrm{tw}_i(z) = \begin{cases} w_i & (z > p_i + q_i), \\ w_i(z - q_i)/p_i & (q_i \le z \le p_i + q_i), \\ 0 & (z < q_i), \end{cases}$$

and $\mathrm{tw}(z) = \sum\{\mathrm{tw}_i(z) : i \in N\}$. Here $\boldsymbol{R}$ denotes a set of real numbers. It is clear that the function $\mathrm{tw} : \boldsymbol{R} \to \boldsymbol{R}$ is the nondecreasing and piecewise linear, in which breaking points are given by $p_i + q_i$ and q_i for all $i \in N$. The function tw helps us to find an optimal solution of the CBKP.

Lemma 3.3. *Let* $lp = \min\{y \in \boldsymbol{R} : \mathrm{tw}(y) \ge W\}$. *An optimal solution of the CBKP is uniquely determined by* $\boldsymbol{x}^*$ *such that*

$$x_i^* = \begin{cases} 1 & (p_i + q_i \le lp), \\ (lp - q_i)/p_i & (q_i < lp \le p_i + q_i), \\ 0 & (q_i > lp). \end{cases} \tag{3.3}$$

The number lp can be found by the algorithm FIND_LEAST_PROFIT, which is a modification of FIND_CRITICAL_ITEM. In the algorithm, the next lemma is necessary for finding lp efficiently.

Lemma 3.4. *For any two numbers* y, z *with* $y \le z$, *it holds that*

$$\begin{aligned} \mathrm{tw}(z) - \mathrm{tw}(y) \ = \ & (z - y) \sum\{w_i/p_i : i \in N,\ y \le q_i < z \textit{ or } q_i < y < p_i + q_i\} \\ & - \sum\{w_i(q_i - y)/p_i : i \in N,\ y < q_i \le z\} \\ & - \sum\{w_i(z - p_i - q_i)/p_i : i \in N,\ y < p_i + q_i \le z\}. \end{aligned}$$

Set $a_i = q_i,\ a_{n+i} = p_i + q_i$ for each $i \in N$, and $N' = \{n+1, \cdots, 2n\}$. In the description of the algorithm, we use a total order $\prec$ on $N \cup N'$ such that

$$\forall\, i, j \in N \cup N', \quad i \prec j \Longleftrightarrow a_i < a_j, \text{ or } a_i = a_j \text{ and } i < j.$$

algorithm FIND_LEAST_PROFIT;
begin
$\quad sum := 0;\ lwr := 0;\ rate := \sum\{w_i/p_i : i \in N\};\ I := N \cup N';$
$\quad$ **while** $|I| > 1$ **do begin**
$\qquad$ select the $\lfloor (|I|+1)/2 \rfloor$-th smallest index k of I with respect to the total order $\prec$;
$\qquad rate_1 := rate - \sum\{w_i/p_i : i \in I \cap N,\ i \succ k\};$
$\qquad sum_1 := (a_k - lwr)rate_1 - \sum\{w_i(a_i - lwr)/p_i : i \in I \cap N,\ i \preceq k\}$
$\qquad\qquad\qquad - \sum\{w_{i-n}(a_k - a_i)/p_{i-n} : i \in I \cap N',\ i \preceq k\};$
$\qquad$ **if** $sum + sum_1 < W$ **then**
$\qquad\quad sum := sum + sum_1;\ lwr := a_k;$
$\qquad\quad rate := rate - \sum\{w_{i-n}/p_{i-n} : i \in I \cap N',\ i \preceq k\};\ I := \{i \in I : i \succ k\};$
$\qquad$ **else**
$\qquad\quad rate := rate_1;\ I := \{i \in I : i \preceq k\};$
$\qquad$ **endif** ;
$\quad$ **end** ;
$\quad lp := lwr + (W - sum)/rate;$
end .

Since FIND_LEAST_PROFIT also runs in $\mathrm{O}(n)$ time, Lemma 3.3 implies the linear-time solvability of the CBKP.

4 Relations between Tree-Shaped Facility Locations and Knapsack Problems

4.1 The Discrete Tree Center Problem

This subsection shows that the discrete tree center problem,

$$\text{DTCP}\left|\begin{array}{ll} \text{Minimize} & \mathrm{ecc}(S) \\ \text{subject to} & \mathrm{size}(S) \le L, \\ & S \text{ is a discrete subtree of } T, \end{array}\right.$$

can be reduced to the bottleneck knapsack problem. Let α denote a vertex center in T.

Lemma 4.1. *There exists a discrete tree center containing* α.

Proof. Suppose that a discrete tree center S does not contain the vertex center α. We denote by u the nearest point in $V(S)$ from α. For any $w \in V(T)$, if $w \in V(T_\alpha(u))$ then $d(w,u) < d(w,\alpha) \le \mathrm{ecc}(\alpha)$, otherwise $d(w,u) = d(w,S) \le \mathrm{ecc}(S) \le \mathrm{ecc}(\alpha)$. Thus, $\mathrm{ecc}(u) \le \mathrm{ecc}(\alpha)$, and u is also a vertex center. Since $\mathrm{ecc}(S) = \mathrm{ecc}(u) = \mathrm{ecc}(\alpha)$, the vertex center α is also a discrete tree center. ∎

In view of Lemma 4.1, it suffices to consider discrete subtrees containing α. Set $a(u,v) = \max\{d(w,u) : w \in V(T_\alpha(v))\}$ for each $[u,v] \in E(T)$ with $u \overset{\alpha}{\to} v$.

Lemma 4.2. *Let S be any discrete subtree containing α. If $S \neq T$ then*

$$\mathrm{ecc}(S) = \max\{a(u,v) : [u,v] \in E(T \setminus S)\}.$$

Proof. The claim immediately follows from Corollary 2.3. ∎

By using a 0-1 valued vector $\boldsymbol{y}$, we can reformulate the DTCP to

$$\text{DTCP1}\left|\begin{array}{lll} \text{Minimize} & \max\{a(u,v)\,(1-y(u,v)) : [u,v] \in E(T)\} & \\ \text{subject to} & \sum\{l(u,v)\,y(u,v) : [u,v] \in E(T)\} \leq L, & \\ & S_\alpha(\boldsymbol{y}) \text{ is connected}, & \\ & y(u,v) \in \{0,1\} & ([u,v] \in E(T)). \end{array}\right.$$

Putting $L' = \sum\{l(u,v) : [u,v] \in E(T)\} - L$ and $\boldsymbol{x} = \mathbf{1} - \boldsymbol{y}$, we obtain

$$\text{DTCP2}\left|\begin{array}{llll} \text{Minimize} & \max\{a(u,v)\,x(u,v) : [u,v] \in E(T)\} & & \\ \text{subject to} & \sum\{l(u,v)\,x(u,v) : [u,v] \in E(T)\} \geq L', & & (4.1) \\ & S_\alpha(\mathbf{1}-\boldsymbol{x}) \text{ is connected}, & & (4.2) \\ & x(u,v) \in \{0,1\} & ([u,v] \in E(T)). & (4.3) \end{array}\right.$$

Lemma 4.3. *There exists an optimal solution of the DTCP2 with the condition*

$$\forall\, [u,v],[w,z] \in E(T),\ \textit{if}\ a(u,v) \geq a(w,z)\ \textit{and}\ x(u,v) = 1\ \textit{then}\ x(w,z) = 1. \quad (4.4)$$

Proof. Let $\boldsymbol{x}^*$ be an optimal solution of the DTCP2 and $A = \max\{a(u,v) : [u,v] \in E(T),\ x^*(u,v) = 1\}$. We define the vector $\hat{\boldsymbol{x}} \in \{0,1\}^{E(T)}$ such that for each $[u,v] \in E(T)$, if $a(u,v) \leq A$ then $\hat{x}(u,v) = 1$, otherwise $\hat{x}(u,v) = 0$. Clearly, the condition (4.4) holds with $\hat{\boldsymbol{x}}$. We obtain $a(u,v) \geq a(v,w) + l(u,v) > a(v,w)$ for any two full edges $[u,v],[v,w] \in E(T)$ with $[u,v] \overset{\alpha}{\to} [v,w]$, which, together with the condition (4.4) and Lemma 2.1, provides that $S_\alpha(\mathbf{1} - \hat{\boldsymbol{x}})$ is connected. It is obvious that $\hat{\boldsymbol{x}}$ satisfies (4.1) and has the same objective value as $\boldsymbol{x}^*$. Therefore, $\hat{\boldsymbol{x}}$ is also optimal for the DTCP2. ∎

The proof of the previous lemma yields the next corollary.

Corollary 4.4. *If $\boldsymbol{x} \in \{0,1\}^{E(T)}$ satisfies the condition* (4.4), *then it also satisfies the condition* (4.2).

Thus, any optimal solution of

$$\text{DTCP3}\left|\begin{array}{ll} \text{Minimize} & \max\{a(u,v)\,x(u,v) : [u,v] \in E(T)\} \\ \text{subject to} & (4.1),\ (4.3),\ (4.4) \end{array}\right.$$

is also optimal for the DTCP2. By removing the condition (4.4) from the DTCP3, we obtain a bottleneck knapsack problem. For such a problem, we stated in Section 3 that an optimal solution with the condition (4.4) can be found in $\mathrm{O}(|E(T)|) = \mathrm{O}(|V|)$ time.

The selection of a vertex center α can be done in $\mathrm{O}(|V|)$ time [8]. By the recursive equation

$$a(u,v) = \begin{cases} l(u,v) & (v \text{ is a leaf of } T_\alpha), \\ l(u,v) + \max\{a(v,w) : [v,w] \in E(T),\ [u,v] \xrightarrow{\alpha} [v,w]\} & (\text{otherwise}), \end{cases}$$

we can compute $a(u,v)$ for all $[u,v] \in E(T)$ in $\mathrm{O}(|V|)$ time. The following theorem consequently holds.

Theorem 4.5. *The discrete tree center problem is reducible to the bottleneck knapsack problem in* $\mathrm{O}(|V|)$ *time, and solvable in* $\mathrm{O}(|V|)$ *time.*

4.2 The Continuous Tree Center Problem

In this subsection we provide the reducibility of the continuous tree center problem,

$$\text{CTCP}\ \left|\ \begin{array}{ll} \text{Minimize} & \mathrm{ecc}(S) \\ \text{subject to} & \mathrm{size}(S) \le L, \\ & S \text{ is a continuous subtree of } T, \end{array}\right.$$

to the continuous bottleneck knapsack problem. We define β as the absolute center of T.

Lemma 4.6. [6] *The continuous tree center contains the absolute center* β.

Therefore, it is sufficient to consider continuous subtrees containing β. From now on, let us assume that β is a point in $V(T)$. If not, put a new vertex on it and construct a new tree. Addition of a new vertex makes no matter with this problem.

For each full edge $[u,v] \in E(T)$ with $u \xrightarrow{\beta} v$, let $b(u,v) = \max\{d(w,v) : w \in V(T_\beta(v))\}$. Corollary 2.3 can be rewritten as follows.

Lemma 4.7. *Suppose that* S *is a continuous subtree containing* β *with* $S \ne T$. *Then, it holds that*

$$\mathrm{ecc}(S) = \max\{l(p,v) + b(u,v) : [p,v] \in E(T \setminus S),\ p \xrightarrow{\beta} v,\ [p,v] \subseteq [u,v] \in E(T)\}.$$

Using a vector $\boldsymbol{y}$ indexed by $E(T)$, we can restate the CTCP as

$$\text{CTCP1}\ \left|\ \begin{array}{lll} \text{Minimize} & \max\{l(u,v)\,(1-y(u,v)) + b(u,v) : [u,v] \in E(T),\ y(u,v) < 1\} & \\ \text{subject to} & \sum\{l(u,v)\,y(u,v) : [u,v] \in E(T)\} \le L, & \\ & S_\beta(\boldsymbol{y}) \text{ is connected}, & \\ & 0 \le y(u,v) \le 1 & ([u,v] \in E(T)), \end{array}\right.$$

which is equivalent to

$$\text{CTCP2}\ \left|\ \begin{array}{llr} \text{Minimize} & \max\{l(u,v)\,x(u,v) + b(u,v) : [u,v] \in E(T),\ x(u,v) > 0\} & \\ \text{subject to} & \sum\{l(u,v)\,x(u,v) : [u,v] \in E(T)\} \ge L', & (4.5) \\ & S_\beta(\mathbf{1} - \boldsymbol{x}) \text{ is connected}, & (4.6) \\ & 0 \le x(u,v) \le 1 \qquad ([u,v] \in E(T)). & (4.7) \end{array}\right.$$

Recall that $L' = \sum\{l(u,v) : [u,v] \in E(T)\} - L$. Elimination of the condition (4.6) from the CTCP2 yields a continuous bottleneck knapsack problem, referred to as the CTCP3.

Lemma 4.8. *Let* $\boldsymbol{x}^*$ *be the optimal solution of the CTCP3. Then,* $S_\beta(\mathbf{1} - \boldsymbol{x}^*)$ *is a continuous tree center.*

Proof. From Lemma 3.3, there exists a real number lp such that

$$x^*(u,v) = \begin{cases} 1 & (l(u,v) + b(u,v) \leq lp), \\ (lp - b(u,v))/l(u,v) & (b(u,v) \leq lp < l(u,v) + b(u,v)), \\ 0 & (b(u,v) > lp). \end{cases}$$

For any two edges $[u,v],[v,w] \in E(T)$ with $[u,v] \xrightarrow{\beta} [v,w]$, we have $b(u,v) \geq b(v,w) + l(v,w)$, and consequently $x^*(u,v) = 0$ if $x^*(v,w) < 1$. It follows from Lemma 2.1 that $\boldsymbol{x}^*$ satisfies the condition (4.6). ∎

It takes $\mathrm{O}(|V|)$ time to find the optimal solution for the CTCP3 and the absolute center β [4]. Using the recursive equation

$$b(u,v) = \begin{cases} 0 & (v \text{ is a leaf of } T_\beta), \\ \max\{l(v,w) + b(v,w) : [v,w] \in E(T),\ [u,v] \xrightarrow{\beta} [v,w]\} & (\text{ otherwise }), \end{cases}$$

we can compute $b(u,v)$ for all $[u,v] \in E(T)$ in $\mathrm{O}(|V|)$ time.

Theorem 4.9. *The continuous tree center problem is reducible to the continuous bottleneck knapsack problem in* $\mathrm{O}(|V|)$ *time, and solvable in* $\mathrm{O}(|V|)$ *time.*

4.3 The Discrete Tree Median Problem

The discrete tree median problem,

$$\text{DTMP}\left|\begin{array}{ll} \text{Minimize} & \text{dis}(S) \\ \text{subject to} & \text{size}(S) \leq L, \\ & S \text{ is a discrete subtree of } T, \end{array}\right.$$

which is known to be NP-hard [3], can be solved by a modification of the pseudo-polynomial-time algorithm by Rabinovitch and Tamir for another kind of tree-shaped facility location problem [7]. Their algorithm is based on the dynamic programming, and uses a technique called "centroid decomposition" to decrease the time complexity.

Theorem 4.10. *The discrete tree median problem is solvable in* $\mathrm{O}(L^2|V|\log|V|)$ *time.*

We show that the DTMP is related to the 0-1 knapsack problem, and present an approximation algorithm for the DTMP, which is a modified version of that for the 0-1 knapsack problem.

For each point $r \in V(T)$, we deal with the subproblem

$$\text{DTMP}(r)\left|\begin{array}{ll} \text{Minimize} & \text{dis}(S) \\ \text{subject to} & \text{size}(S) \leq L, \\ & S \text{ is a discrete subtree of } T,\ r \in S. \end{array}\right.$$

Solving the DTMP(r) for all $r \in V(T)$, we can obtain a discrete tree median. Connected with each full edge $[u,v] \in E(T)$ with $u \xrightarrow{r} v$, define

$$c(u,v) = |V(T_r(v))|. \tag{4.8}$$

From Corollary 2.5, the DTMP(r) is rewritten by using a 0-1 vector $\boldsymbol{x}$ as

$$\text{DTMP1}(r) \left| \begin{array}{lll} \text{Maximize} & \sum\{c(u,v)\, l(u,v)\, x(u,v) : [u,v] \in E(T)\} & \\ \text{subject to} & \sum\{l(u,v)\, x(u,v) : [u,v] \in E(T)\} \le L, & (4.9) \\ & S_r(\boldsymbol{x}) \text{ is connected}, & (4.10) \\ & x(u,v) \in \{0,1\} \qquad ([u,v] \in E(T)). & (4.11) \end{array} \right.$$

Except for the constraint (4.10), this problem is a 0-1 knapsack problem. Unfortunately, there does not always exist an optimal solution of this 0-1 knapsack problem which is also optimal for the DTMP1(r). Nevertheless, an approximation algorithm of the 0-1 knapsack problem also applies for the DTMP1(r) with a slight modification.

Note that if $d(r,v) > L$ for a full edge $[u,v] \in E(T)$ with $u \xrightarrow{r} v$, then any feasible subtree S of the DTMP(r) does not contain $[u,v]$. Thus, the following 0-1 knapsack problem arises from the DTMP1(r):

$$\text{DTMP2}(r) \left| \begin{array}{ll} \text{Maximize} & \sum\{c(u,v)\, l(u,v)\, x(u,v) : [u,v] \in E(T)\} \\ \text{subject to} & (4.9),\ (4.11), \\ & x(u,v) = 0 \quad ([u,v] \in E(T) \setminus E_r(T)), \end{array} \right.$$

where

$$E_r(T) = \{[u,v] \in E(T) : u \xrightarrow{r} v,\ d(r,v) \le L\}.$$

When $\sum\{l(u,v) : [u,v] \in E_r(T)\} \le L$,

$$x(u,v) = \begin{cases} 1 & ([u,v] \in E_r(T)), \\ 0 & (\text{otherwise}) \end{cases}$$

is optimal for both the DTMP1(r) and the DTMP2(r). We now assume that $\sum\{l(u,v) : [u,v] \in E_r(T)\} > L$. Suppose we are given a total order $\prec$ on $E(T)$ satisfying

$$\forall\ [u,v],[w,z] \in E(T),\ c(u,v) < c(w,z) \Longrightarrow [u,v] \prec [w,z]. \tag{4.12}$$

Let $[p,q] \in E_r(T)$ be the full edge with

$$\sum\{l(u,v) : [u,v] \in E_r(T),\ [u,v] \succ [p,q]\} \le L < \sum\{l(u,v) : [u,v] \in E_r(T),\ [u,v] \succeq [p,q]\}, \tag{4.13}$$

and $\boldsymbol{x}^1,\ \boldsymbol{x}^2 \in \{0,1\}^{E(T)}$ be the vectors given by

$$\begin{aligned} x^1(u,v) &= \begin{cases} 1 & ([u,v] \in E_r(T),\ [u,v] \succ [p,q]), \\ 0 & (\text{otherwise}), \end{cases} \\ x^2(u,v) &= \begin{cases} 1 & ([u,v] \text{ is on the path connecting } r \text{ and } [p,q]), \\ 0 & (\text{otherwise}). \end{cases} \end{aligned}$$

Since two full edges $[u,v],[v,w]\in E(T)$ with $[u,v]\xrightarrow{r}[v,w]$ satisfy $c(u,v)>c(v,w)$, Lemma 2.1 implies that the condition (4.10) holds with respect to $\boldsymbol{x}^1$. The vector $\boldsymbol{x}^2$ also feasible for the DTMP1(r) because $[p,q]\in E_r(T)$. We denote by $\boldsymbol{x}^*$ and $\hat{\boldsymbol{x}}$ optimal solutions of the DTMP1(r) and the DTMP2(r), respectively. It follows from Corollary 2.5, Lemma 3.1 and the inequality

$$\sum\{c(u,v)\,l(u,v)\,\hat{x}(u,v):[u,v]\in E(T)\}\geq\sum\{c(u,v)\,l(u,v)\,x^*(u,v):[u,v]\in E(T)\}$$

that

$$\begin{aligned}
&\operatorname{dis}(r)-\min\{\operatorname{dis}(S_r(\boldsymbol{x}^1)),\operatorname{dis}(S_r(\boldsymbol{x}^2))\}\\
=\;&\max\{\textstyle\sum\{c(u,v)\,l(u,v)\,x^1(u,v):[u,v]\in E(T)\},\\
&\qquad\qquad\qquad\textstyle\sum\{c(u,v)\,l(u,v)\,x^2(u,v):[u,v]\in E(T)\}\}\\
\geq\;&(1/2)\textstyle\sum\{c(u,v)\,l(u,v)\,x^*(u,v):[u,v]\in E(T)\}\\
=\;&(1/2)\{\operatorname{dis}(r)-\operatorname{dis}(S_r(\boldsymbol{x}^*))\}.
\end{aligned}$$

That is, we have

$$\operatorname{dis}(S)\leq\{\operatorname{dis}(S_r(\boldsymbol{x}^*))+\operatorname{dis}(r)\}/2,$$

where S is one of two discrete subtrees $S_r(\boldsymbol{x}^1)$ and $S_r(\boldsymbol{x}^2)$ with the smaller distance-sum.

As mentioned in Section 3, the algorithm FIND_CRITICAL_ITEM finds the full edge $[p,q]\in E_r(T)$ with the condition (4.13) in $\mathrm{O}(|E_r(T)|)=\mathrm{O}(|V|)$ time. The recursive equation

$$c(u,v)=\begin{cases}1 & (v\text{ is a leaf of }T_r),\\ 1+\sum\{c(v,w):[v,w]\in E(T),\ [u,v]\xrightarrow{r}[v,w]\} & (\text{otherwise}),\end{cases}$$

provides the values $c(u,v)$ for all $[u,v]\in E(T)$ in $\mathrm{O}(|V|)$ time. Hence, we can find an approximate solution of the DTMP1(r) in linear time.

Finding an approximate solution of the DTMP1(r) for each $r\in V(T)$, we obtain the following theorem.

Theorem 4.11. *Suppose that S^* is a discrete tree median and that v is any point in $V(S^*)$. Then, we can find a discrete subtree S' with*

$$\operatorname{dis}(S')\leq\{\operatorname{dis}(S^*)+\operatorname{dis}(v)\}/2$$

in $\mathrm{O}(|V|^2)$ time.

4.4 The Continuous Tree Median Problem

This subsection discusses that the continuous tree median problem,

$$\text{CTMP}\left|\begin{array}{ll}\text{Minimize} & \operatorname{dis}(S)\\ \text{subject to} & \operatorname{size}(S)\leq L,\\ & S\text{ is a continuous subtree of }T,\end{array}\right.$$

is reducible to the continuous knapsack problem. We denote by γ an absolute median which is also a vertex median. It is known that such a point γ necessarily exists [2]. The correctness of the algorithm by Minieka [6] implies the next lemma.

Lemma 4.12. *There exists a continuous tree median containing* γ.

Thus, we have only to consider continuous subtrees containing γ. The CTMP can be stated as

$$\text{CTMP1}\left|\begin{array}{lll} \text{Maximize} & \sum\{c(u,v)\,l(u,v)\,x(u,v) : [u,v] \in E(T)\} & \\ \text{subject to} & \sum\{l(u,v)\,x(u,v) : [u,v] \in E(T)\} \leq L, & (4.14) \\ & S_\gamma(\boldsymbol{x}) \text{ is connected}, & (4.15) \\ & 0 \leq x(u,v) \leq 1 \qquad ([u,v] \in E(T)). & (4.16) \end{array}\right.$$

where each $c(u,v)$ is defined by (4.8) when $r = \gamma$. If the condition (4.15) is ignored in the CTMP1, we obtain a continuous knapsack problem. From Section 3, an optimal solution of this continuous knapsack problem is given by

$$x^*(u,v) = \begin{cases} 1 & ([u,v] \succ [p,q]), \\ (L - \sum\{l(u,v) : [u,v] \in E(T),\ [u,v] \succ [p,q]\})/l(p,q) & ([u,v] = [p,q]), \\ 0 & ([u,v] \prec [p,q]), \end{cases}$$

where $\prec$ stands for a total order on $E(T)$ satisfying the condition (4.12), and $[p,q]$ is the full edge with

$$\sum\{l(u,v) : [u,v] \in E(T),\ [u,v] \succ [p,q]\} \leq L < \sum\{l(u,v) : [u,v] \in E(T),\ [u,v] \succeq [p,q]\}.$$

Since $\boldsymbol{x}^*$ satisfies (4.15), $\boldsymbol{x}^*$ is also optimal for the CTMP1.

We can obtain in $\mathrm{O}(|V|)$ time an optimal solution of the above continuous knapsack problem, an absolute median γ [8], and the values $c(u,v)$, respectively. Therefore, we derive the next theorem.

Theorem 4.13. *The continuous tree median problem is reducible in* $\mathrm{O}(|V|)$ *time to the continuous knapsack problem, and solvable in* $\mathrm{O}(|V|)$ *time.*

5 Conclusion

We have discussed four types of tree-shaped facility location problems, and induced the relationship with knapsack-type problems. The discrete tree center, continuous tree center, and continuous tree median problems are reducible in linear time to the bottleneck knapsack, continuous bottleneck knapsack, and continuous knapsack problems, respectively. Linear-time algorithms are presented for the above three location problems by using the linear-time solvability of the corresponding knapsack-type problems. The discrete tree median problem, which is known to be NP-hard, can be related to the 0-1 knapsack problem, and this relation gives an approximation algorithm for the discrete tree median problem.

In the discussion so far, we have used the length $l(u,v)$ of each edge $(u,v) \in E$ to define both the size of a subtree and the distance. Instead of this, suppose that we are additionally given a positive weight $w(u,v)$ of each edge $(u,v) \in E$, and newly define the size of a subtree S as $\mathrm{size}(S) = \sum\{w(u,v) : [u,v] \in E(S)\}$. In this situation, our arguement still holds with the discrete and continuous tree center problems. With respect to the tree median problems, however, the structure of the problems becomes more complicated, and the NP-hardness can be proved for not only the discrete version but also the continuous one.

Acknowledgement

We are grateful to Professor Yasutoshi Yajima and Mr. Takeaki Uno of Tokyo Institute of Technology for their valuable comments.

References

[1] M. Blum, R. W. Floyd, V. Pratt and R. L. Rivest. Time Bounds for Selection. *J. Comput. System Sci.* **7** (1973) 448–461.

[2] S. L. Hakimi. Optimal Locations of Switching Centers and the Absolute Centers and Medians of a Graph. *Operations Research* **12** (1964) 450–459.

[3] S. L. Hakimi, E. F. Schmeichel, and M. Labbé. On Locating Path- or Tree-Shaped Facilities on Networks. *Networks* **23** (1993) 543–555.

[4] G. Y. Handler. Minimax Location of a Facility in an Undirected Tree Graph. *Transportation Sci.* **7** (1973) 287–293.

[5] S. Martello and P. Toth. *Knapsack Problems: Algorithms and Computer Implementations.* John Wiley & Sons, Chichester (1990).

[6] E. Minieka. The Optimal Location of a Path or Tree in a Tree Network. *Networks* **15** (1985) 309–321.

[7] R. Rabinovitch and A. Tamir. On a Tree-Shaped Facility Location Problem of Minieka. *Networks* **22** (1992) 515–522.

[8] A. Rosenthal and J. A. Pino. A Generalized Algorithm for Centrality Problem on Trees. *J. Assoc. Compute. Mach.* **36** (1989) 349–361.

[9] B. C.Tansel, R. L. Francis and T. J. Lowe. Location on Networks: A Survey, Parts I and II. *Management Sci.* **29** (1983) 482–511.