$(r|p)$-Centroid Problems on Networks with Vertex and Edge Demand[†]

Dominik Kress[*], Erwin Pesch

Department of Management Information Science, University of Siegen, Hölderlinstraße 3, D-57068 Siegen, Germany

Abstract

This paper analyzes $(r|p)$-centroid problems on networks with vertex and edge demand under a binary choice rule. Bilevel programming models are presented for the discrete problem class. Furthermore, NP-hardness proofs for the discrete and continuous $(1|p)$-centroid problem on general networks with edge demand only are provided. Nevertheless, an efficient algorithm to determine a discrete $(1|p)$-centroid of a tree network with vertex and edge demand can be derived.

Keywords: location, competitive location, centroid, edge demand, bilevel programming

1. Introduction

Location problems are concerned with the location of (physical or nonphysical) resources in some given space. Competitive location models (dating back to Hotelling [1]) additionally incorporate the fact that location decisions have been or will be made by independent decision-makers who will subsequently compete with each other, e.g. for market share when we think of locating facilities such as gas stations or supermarkets. Several reviews and classifications have appeared in this field. They include Drezner [2], Eiselt et al. [3], Eiselt and Laporte [4], Kress and Pesch [5], Plastria [6], Serra and ReVelle [7].

The representation of the location space may be divided into different classes. We follow ReVelle and Eiselt [8] in differentiating between d-dimensional real space and network location problems, each of which further being subdivided into continuous and discrete problems. A discrete problem arises, when the set of candidate locations is assumed to be finite and known a priori. In a continuous problem, any point of the network or the d-dimensional space is a potential location site. By identifying finite sets of points that capture optimal locations (finite dominating sets, Hooker et al. [9]), some continuous problem classes can be transformed into equivalent discrete problem classes a posteriori. Customer behavior is represented by some kind of choice rule. A choice rule is said to be binary (or deterministic), when the total demand of a customer is served by a single

[*]Corresponding author, phone: +49 271 740 3402, fax: +49 271 740 2940

Email addresses: dominik.kress@uni-siegen.de (Dominik Kress), erwin.pesch@uni-siegen.de (Erwin Pesch)

[†] This is an Accepted Manuscript of an article published by Elsevier in **Computers & Operations Research** on 5 March 2012, available online: https://doi.org/10.1016/j.cor.2012.02.025
© 2012. This manuscript version is made available under the CC-BY-NC-ND 4.0 license http://creativecommons.org/licenses/by-nc-nd/4.0/

facility; it is said to be probabilistic (or proportional), if demand is split over multiple facilities. Other fundamental categories of competitive location theory are related to game theoretic aspects. Competition itself, for instance, may be static, dynamic, or competitors may enter in a simultaneous or sequential fashion. The latter mode of competition (rooted in the work of Hay [10] and Prescott and Visscher [11]) is characterized by two types of players: leaders, who choose locations at given instants, anticipating the subsequent actions of later entrants, and followers, who make their location decisions based on the past decisions of the leaders. The solution concept generally employed in sequential location problems is the Stackelberg equilibrium [12]: Assuming rational players, the location of each player is determined by backward induction.

Hakimi [13] formally introduced the terms $(r|X_p)$-medianoid problem and $(r|p)$-centroid problem for sequential games with one leader (L) and one follower (F) locating p and r facilities, respectively. Note that r and p are arbitrary input parameters. Knowing the p locations of L, denoted by $X_p = (x_1, ..., x_p)$, F faces the problem of optimally locating r facilities (with respect to some objective function): the $(r|X_p)$-medianoid problem. L's problem, the $(r|p)$-centroid problem, is to locate p facilities, anticipating F's subsequent behavior. In this paper we will consider the objective of maximizing market share for both, L and F. We will generally precede the terms $(r|p)$-centroid and $(r|X_p)$-medianoid problem with the terms discrete, continuous, binary or proportional throughout the rest of the paper. This clarifies the type of choice rule (binary or proportional) and location space (discrete or continuous problems on networks) under consideration.

Whereas competitive location models in $\mathbb{R}^1$ typically assume the demand to be continuously dispersed over the line segment, the majority of network models incorporate discrete demand, i.e. demand arising in the vertices of the network. Dasci et al. [14] and Okunuki and Okabe [15] were among the first to consider demand densities over the edges of competitive location problems on general networks. In an urban context, the motivation is as follows: Cities are typically modeled as networks. Edges correspond to streets and vertices represent intersections. The houses of a city are usually dispersed over the streets. Thus, demand is typically non-discrete. While Dasci et al. [14] and Okunuki and Okabe [15] consider $(r|X_p)$-medianoid problems, it is the aim of this paper to apply the concept of edge demand to $(r|p)$-centroid problems. We will show that the discrete and continuous, binary $(1|p)$-centroid problem are NP-hard on general networks with edge demand only. Furthermore, we will adapt the definition of ξ-bounding sets by Spoerhase and Wirth [16] to design an efficient algorithm to determine a discrete, binary $(1|p)$-centroid of a tree network with vertex and edge demand.

The remainder of this paper is organized as follows. The notation and definitions used throughout the paper are given in Section 2. Section 3 is devoted to the binary $(r|p)$-centroid problem with vertex and edge demand on general networks. We present bilevel programming formulations for the discrete problem class in Section 3.1. While the focus of the paper is on drawing the line between hard and easy problems, some computational results for the corresponding $(r|X_p)$-medianoid problem are given in Section 3.2 to illustrate the need for approximate solution procedures when designing heuristic algorithms for the leader's problem. The above-mentioned NP-hardness proofs are subject of Section 3.3. Section 3.4 is concerned with the identification of finite dominating sets. The efficient algorithm for the discrete, binary $(1|p)$-centroid problem on a tree network with vertex and edge demand is given in Section 4. We introduce the basic ideas by restricting our attention to chain networks in Section 4.1, before describing the algorithm itself in Section 4.2. The paper ends with a conclusion in Section 5.

2. Basic notation and definitions

We extend and use the notation of Bandelt [17] in this paper (cf. also [18]). A *network* $N = (V, E, \lambda)$ consists of a finite set V ($|V| = n$), a finite set E ($|E| = m$) of two-element subsets of V and a mapping $\lambda : E \to \mathbb{R}^+$. The pair (V, E) gives a *graph* in the usual sense (cf. [19]). The elements v of V are called *vertices* of the network. The elements e of E are the *edges*. Every edge joins two distinct vertices of N. If e is an edge joining u and v this is expressed by the shorthand $e = [u, v]$. We assume that all edges are undirected, hence $[u, v] = [v, u]$. The value $\lambda(e) = \lambda(uv)$ is the *length* of e. An edge $[u, u]$ is a *loop*. The sum of the number of edges that join a vertex $v \in V$ with other vertices of the network and twice the number of loops at v is the *degree* of v. The *points* x of N ($x \in N$) are the elements of the edges (including all vertices). Two points x and y on an edge e ($x, y \in e$) determine a *subedge* $[x, y]$ of e, the length of which is denoted by $\lambda([x, y])$. A *path* $P(x, y)$ joining two points $x \in [u, v]$ and $y \in [w, z]$ is either a subedge or a sequence of edges and (at most two) subedges passing at most once through each point, where $P(x, y)$ contains x and y but no proper connected subset of $P(x, y)$ does. The points x and y are the *end points* of $P(x, y)$. The length of $P(x, y)$ is equal to the sum of the lengths of the edges and subedges. If the length of $P(x, y)$ is minimum among all paths connecting x and y, then $P(x, y)$ is a shortest path; its length is the *distance* $d(x, y)$ between x and y. We define $D(p, Z) := \min\{d(p, z) | z \in Z\}$ for a point $p \in N$ and a set of points $Z \subseteq N$. A *cycle* consists of an edge e joining two vertices u and v and some path $P(u, v) \neq e$ connecting u and v. A network is connected if for any two points x and y there exists a path joining x and y. A connected network without cycles is a *tree network*. A tree network where every vertex is incident to at most two edges is a *chain network*. We assume that the networks considered in this paper are connected and that there are no multiple edges. Moreover, we assume that there are no loops at the vertices.

Let V' be a subset of the vertex set of N. The network $N' = (V', E', \lambda')$ is the *subnetwork* of N on the vertex set V', if E' is a subset of E such that each edge of E joining u and v belongs to E' if and only if u and v are in V'. The mapping λ' is the restriction of λ to E'.

Let a tree network $N = (V, E, \lambda)$ be rooted at some distinguished vertex $r \in V$. For each pair of vertices $i \in V$ and $j \in V$, we call i a *descendant* of j, if j is on the unique path that connects i to the root r. If i is a descendant of j, we call j an *ancestor* of i. A vertex $v \in V$ is a *common ancestor* of two vertices $x, y \in V$, if it is an ancestor of both, x and y. A common ancestor of two vertices $x, y \in V$ is the *nearest common ancestor*, $nca(x, y)$, of these very vertices, if its distance to the root is the largest among all common ancestors. If $i \in V$ is a descendant of $j \in V$ and $[i, j] \in E$, then i is said to be a *child* of j and j is called the *father* of i. A vertex without children is a *leaf* of the tree network. For any vertex $v \in V$ we denote the subnetwork (subtree) of N on the vertex set $V_{T_v} := \{v\} \cup \{i \in V | i \text{ is a descendant of } v\}$ by T_v and the subnetwork on the vertex set $V'_{T_v} := V_{T_v} \setminus \{v\}$ by T'_v.

We associate a (local) coordinate $x_{uv} \in [0, \lambda(uv)]$ with every edge $[u, v] \in E$ of a graph N. Thus, we are able to define any point of the graph. The direction of counting can be defined arbitrarily.

A finite number of users is located at the vertices of the network N (vertex demands, vertex customers). At each vertex there may be several users or none at all. Their demand is described by a weight function $\pi : V \to \mathbb{R}_0^+$. We define $\pi(x) := 0$ for all $x \notin V$. Additionally, we consider a mapping $\delta : E \to \mathbb{R}_0^+$ with every edge $e = [u, v] \in E$. The value $\delta(e) = \delta(uv)$ is the uniform demand density (edge demand, edge customer) of the edge $e = [u, v]$, i.e. the demand per unit of length (see Figure 1). δ and π may not be equal to the zero function at the same time.

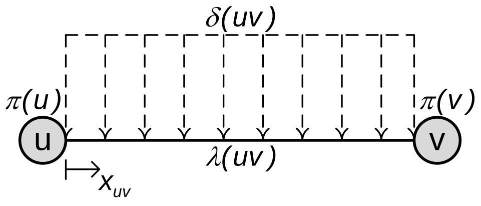


Figure 1: Vertex and edge demand.

For a subnetwork N' of N we denote by $\pi(V')$ the sum $\sum_{u \in V'} \pi(u)$ where V' is the vertex set of N'. Analogously, we denote by $\delta(E')$ the sum $\sum_{e \in E'} \delta(e) \cdot \lambda(e)$ where E' is the edge set of N'. Finally, we define $\xi(N') := \pi(V') + \delta(E')$. We denote the accumulated demand of the customers who accommodate their demand at F's facilities by $W_F(Y_r(X_p)|X_p)$, where $Y_r(X_p)$ corresponds to an arbitrary feasible set of F's locations, given the location decision X_p of L. The optimal follower's market share is denoted by $W_F^*(Y_r(X_p)|X_p)$. Analogously, we denote L's (optimal) market share by $W_L(r|p)$ ($W_L^*(r|p)$).

Let $N = (V, E, \lambda)$ be a tree network. Add an artificial vertex $n + 1$ with $\pi(n + 1) = 0$ as the root of N and connect it to an arbitrary vertex $s \in V$ by an artificial edge $[s, n+1]$ with $\lambda([s, n + 1]) = \infty$ and $\delta([s, n + 1]) = 0$.

Definition 1 (Spoerhase and Wirth [16]). Let $X \subseteq V \setminus \{n + 1\}$ be a vertex subset and $0 \leq \hat{\xi} \leq \xi(N)$. Set X is called $\hat{\xi}$-*bounding* if

1. $W_F^*(Y_1(X)|X) \leq \hat{\xi}$ and
2. $\forall\ x \in X$ with father x' we have $W_F^*(Y_1((X \setminus \{x\}) \cup \{x'\})|(X \setminus \{x\}) \cup \{x'\}) > \hat{\xi}$.

Finally, we define:

Definition 2. Let $N = (V, E, \lambda)$ be a tree network. Furthermore, let $X \subseteq V$, $v \in V$ and u be the father of v, $\{u, v\} \cap X = \emptyset$. Then the X-*subtracted subtree rooted in* u is the subnetwork on the vertex set

$$V_{X_s} := \{u\} \cup \left\{ V_{T_v} \setminus \bigcup_{i \in \{X \cap V_{T_v}\}} V'_{T_i} \right\}.$$

3. The binary (r|p)-centroid problem with vertex and edge demand

In the remainder of this paper we will consider a binary choice rule, i.e. each customer chooses the closest facility to accommodate all of his demand. We additionally assume that ties are broken in favor of the leader. This is a common assumption in the field of competitive location problems; see, for example, Hakimi [20] (binary $(r|p)$-centroid problem with vertex demand only) or Hansen and Labbé [21], Hansen and Thisse [22] (Condorcet and Simpson points). As a direct consequence we may assume that $p+r \leq n$ for the discrete version of the centroid problem under consideration.

3.1. Bilevel programming models for the discrete, binary $(r|p)$-centroid problem with vertex and edge demand

Consider the discrete, binary $(r|p)$-centroid problem with vertex and edge demand. For the remainder of this section, we assume – without loss of generality – that the underlying network is complete. If this is not the case, we simply add missing edges with infinite edge lengths and zero demand densities.

We define the following variables:

$$x_{ij}^F := \begin{cases} 1 & \text{if the vertex customers located in vertex i are served} \\ & \text{by a follower's facility located in vertex j,} \\ 0 & \text{else,} \end{cases} \qquad \forall\, i,j \in V, \tag{1}$$

$$x_{ij}^L := \begin{cases} 1 & \text{if the vertex customers located in vertex i are served} \\ & \text{by a leader's facility located in vertex j,} \\ 0 & \text{else,} \end{cases} \qquad \forall\, i,j \in V, \tag{2}$$

$$y_j^F := \begin{cases} 1 & \text{if the follower locates in vertex j,} \\ 0 & \text{else,} \end{cases} \qquad \forall\, j \in V, \tag{3}$$

$$y_j^L := \begin{cases} 1 & \text{if the leader locates in vertex j,} \\ 0 & \text{else,} \end{cases} \qquad \forall\, j \in V. \tag{4}$$

For all $i,j = 1,...,n$, let C_{ij} and $\bar{C}_{ij}$ be the sets of vertices k with $d(k,i) < d(i,j)$ or $d(k,i) = d(i,j)$, respectively (cf. also Dobson and Karmarkar [23]). Furthermore, define $\hat{C}_{ij} := C_{ij} \cup \bar{C}_{ij}$. We can now state a binary bilevel nonlinear programming model (BBNP) for the problem under consideration (refer to Dempe [24] for an introduction to bilevel programming).

$$\max_{x^L, y^L} \quad \sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij}^L \left[\pi(i) + \sum_{k=1}^{n} \tfrac{1}{2}\delta(i,k)\left(\lambda(i,k) - d(i,j) + \sum_{l=1}^{n}(x_{kl}^L + x_{kl}^F)d(k,l)\right)\right] \tag{5}$$

$$\text{subject to} \quad \sum_{j=1}^{n} y_j^L = p, \tag{6}$$

$$x_{ij}^L \leq y_j^L \qquad \forall\, i,j = 1,...,n, \tag{7}$$

$$\sum_{j=1}^{n} x_{ij}^L \leq 1 \qquad \forall\, i = 1,...,n, \tag{8}$$

$$x_{ij}^L + y_k^L \leq 1 \qquad \forall\, i,j = 1,...,n,\ k \in C_{ij} \tag{9}$$

$$x_{ij}^L + y_k^F \leq 1 \qquad \forall\, i,j = 1,...,n,\ k \in C_{ij} \tag{10}$$

and x^F, y^F being a solution of

$$\max_{x^F,y^F} \quad \sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij}^F \Big[\pi(i) + \sum_{k=1}^{n} \tfrac{1}{2}\delta(i,k)\left(\lambda(i,k) - d(i,j) + \sum_{l=1}^{n}(x_{kl}^L + x_{kl}^F)d(k,l)\right)\Big] \tag{11}$$

$$\text{subject to} \qquad \sum_{j=1}^{n} y_j^F = r, \tag{12}$$

$$x_{ij}^F \le y_j^F \qquad \forall\, i,j = 1,...,n, \tag{13}$$

$$\sum_{j=1}^{n}\left(x_{ij}^F + x_{ij}^L\right) = 1 \qquad \forall\, i = 1,...,n, \tag{14}$$

$$x_{ij}^F + y_k^L \le 1 \qquad \forall\, i,j = 1,...,n,\ k \in \hat{C}_{ij} \tag{15}$$

$$x_{ij}^F + y_k^F \le 1 \qquad \forall\, i,j = 1,...,n,\ k \in C_{ij} \tag{16}$$

$$x_{ij}^L, x_{ij}^F \in \{0,1\} \quad \forall\, i,j = 1,...,n, \tag{17}$$

$$y_j^L, y_j^F \in \{0,1\} \quad \forall\, j = 1,...,n. \tag{18}$$

The objective functions (5) and (11) correspond to the maximization of the market share of each of the players, taking into account vertex and edge demand. Constraints (6) and (12) ensure that the players locate exactly p and r facilities, respectively. Conditions (7) and (13) guarantee that a (vertex) customer can only be served if the corresponding facility has been opened. Constraints (8) and (14) relate to the fact that the demand of each (vertex) customer must be served by exactly one facility, while conditions (9), (10), (15) and (16) guarantee that each (vertex) customer is served by its closest facility (cf. Dobson and Karmarkar [23]). Observe that edge customers are guaranteed to be served implicitly. Figure 2 depicts an example. Vertex i is served by a leader's facility in vertex j, while vertex k is served by a follower's facility in vertex l. Therefore, there must exist a point $x_{ik} = x$ on edge $[i,k]$, where $d(i,j) + x = \lambda(i,k) - x + d(k,l)$. Every edge customer located at $x_{ik} \le x$ ($x_{ik} > x$) will be served by the leader (follower).

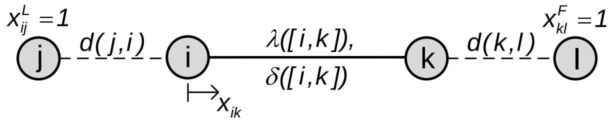


Figure 2: Serving of edge customers.

Observe that

$$\sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij}^L \left[\pi(i) + \sum_{k=1}^{n}\frac{1}{2}\delta(i,k)\left(\lambda(i,k) - d(i,j) + \sum_{l=1}^{n}(x_{kl}^L + x_{kl}^F)d(k,l)\right)\right]$$
$$= \sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij}^L\pi(i) + \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\frac{1}{2}\delta(i,k)x_{ij}^L\left(\lambda(i,k) - d(i,j)\right)$$
$$+ \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n}\frac{1}{2}\delta(i,k)x_{ij}^L x_{kl}^L d(k,l) + \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n}\frac{1}{2}\delta(i,k)x_{ij}^L x_{kl}^F d(k,l).$$

An analogous result holds for the follower's objective function. Thus, we may linearize BBNP by defining binary variables z_{ijkl}^{LL}, z_{ijkl}^{LF}, z_{ijkl}^{FL} and z_{ijkl}^{FF} for all $i, j, k, l = 1, ..., n$ and introducing additional constraints. This results in a binary bilevel linear programming model (BBLP).

$$\begin{aligned}
\max_{x^L, y^L} \quad & \sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij}^{L}\pi(i) + \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \tfrac{1}{2}\delta(i,k)x_{ij}^{L}\left(\lambda(i,k) - d(i,j)\right) \\
& + \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} \tfrac{1}{2}\delta(i,k)z_{ijkl}^{LL}d(k,l) + \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} \tfrac{1}{2}\delta(i,k)z_{ijkl}^{LF}d(k,l) \qquad (19)
\end{aligned}$$

subject to constraints (6)-(10)

$$z_{ijkl}^{LL} \le \tfrac{1}{2}(x_{ij}^{L} + x_{kl}^{L}) \quad \forall\, i,j,k,l = 1,...,n, \qquad (20)$$

$$z_{ijkl}^{LF} \le \tfrac{1}{2}(x_{ij}^{L} + x_{kl}^{F}) \quad \forall\, i,j,k,l = 1,...,n, \qquad (21)$$

and x^F, y^F being a solution of

$$\begin{aligned}
\max_{x^F, y^F} \quad & \sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij}^{F}\pi(i) + \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \tfrac{1}{2}\delta(i,k)x_{ij}^{F}\left(\lambda(i,k) - d(i,j)\right) \\
& + \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} \tfrac{1}{2}\delta(i,k)z_{ijkl}^{FL}d(k,l) \\
& + \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} \tfrac{1}{2}\delta(i,k)z_{ijkl}^{FF}d(k,l) \qquad (22)
\end{aligned}$$

subject to constraints (12)-(18)

$$z_{ijkl}^{FL} \le \tfrac{1}{2}(x_{ij}^{F} + x_{kl}^{L}) \quad \forall\, i,j,k,l = 1,...,n, \qquad (23)$$

$$z_{ijkl}^{FF} \le \tfrac{1}{2}(x_{ij}^{F} + x_{kl}^{F}) \quad \forall\, i,j,k,l = 1,...,n, \qquad (24)$$

$$z_{ijkl}^{LL},\ z_{ijkl}^{LF} \in \{0,1\} \quad \forall\, i,j,k,l = 1,...,n, \qquad (25)$$

$$z_{ijkl}^{FL},\ z_{ijkl}^{FF} \in \{0,1\} \quad \forall\, i,j,k,l = 1,...,n. \qquad (26)$$

Constraints (20), (21), (23) and (24) can easily be rearranged to receive a disaggregated model (BBLP2). Inequalities (20), for example, may be replaced by

$$z_{ijkl}^{LL} \le x_{ij}^{L} \quad \forall\, i,j,k,l = 1,...,n, \qquad (27)$$

$$z_{ijkl}^{LL} \le x_{kl}^{L} \quad \forall\, i,j,k,l = 1,...,n. \qquad (28)$$

3.2. Potential solution approaches for the discrete, binary $(r|p)$-centroid problem on general networks with vertex and edge demand

The focus in the remainder of this paper is on drawing the line between hard and easy $(r|p)$-centroid problems with vertex and edge demand. Nevertheless, we give some basic results concerning the design of solution approaches for the discrete, binary $(r|p)$-centroid problem on general networks with vertex and edge demand in this section.

Due to their hierarchical structure, bilevel programming problems are very hard to solve. The fact that even the linear bilevel programming problem in continuous variables is NP-hard in the strong sense [25, Chap. 5] suggests, that we are unlikely to be confronted with polynomially time solvable problems. Indeed, as we will see in the following section, the discrete, binary $(1|p)$-centroid problem on a general network is NP-hard if we consider vertex demand only, edge demand only or vertex and edge demand. Thus, the design of exact solution approaches does not seem to be very promising. Heuristic solution approaches will need to compute or approximate the follower's response to some given set of leader's locations. Since the discrete, binary $(r|X_1)$-medianoid problem is known to be NP-hard on general networks (again, for the case of vertex demand only, edge demand only or vertex and edge demand, cf. Dasci et al. [14], Hakimi [20]), calculating the follower's response is most likely too expensive with regard to computational time. To support this assumption, we have coded the model BBNP in C++ for fixed sets of leader's locations. The resulting quadratic programs were solved with CPLEX 12.2 with a time limit of 1800 seconds. We generated a total of 50 test networks; ten complete networks for $n = 5, 10, 15, 20, 25$, respectively. Integer edge lengths, edge demand densities and vertex demands with $\lambda([u, v]) \in [1, 5]$, $\delta([u, v]) \in [0, 10]$ and $\pi(u) \in [0, 10]$ for all $u, v \in V$ were generated randomly. Test instances were run on a Laptop with an Intel Core i7 CPU, 2.67 GHz, 4GB system memory, running under the 64bit Windows 7 Professional operating system. Figures 3 and 4 plot the average computational time for the 5 network sizes with $r = 1, 2, 3, 4$ and $p = 1$ or $p = 2$ random leader locations. Details are given in the appendix. For additional test instances on networks with $n \geq 40$ we frequently ran out of memory while generating or solving the models.

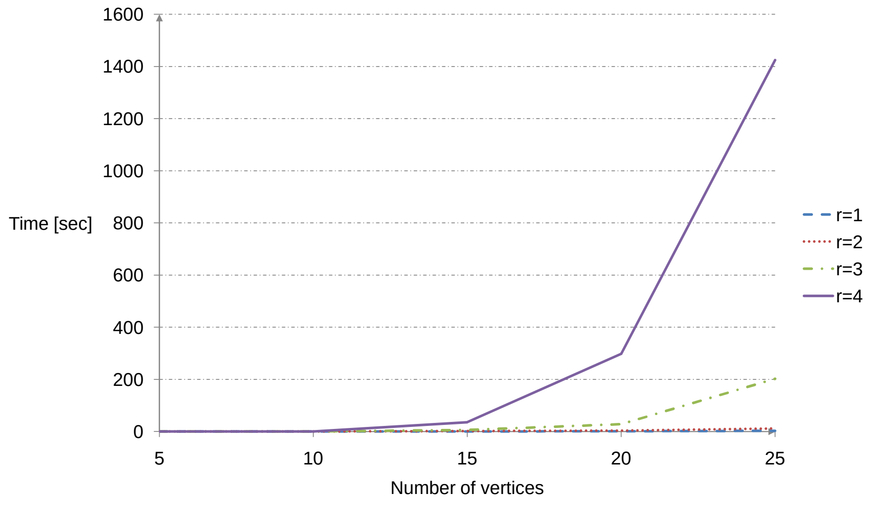


Figure 3: Computational results for the medianoid problem, $p = 1$.

Thus, since computational times are fairly large for test instances on relatively small networks, it seems to be reasonable to approximate the follower's response when designing heuristic algorithms for the discrete, binary $(r|p)$-centroid problem on networks with vertex and edge demand. Therefore, we refer to the tabu search algorithms by Benati and Laporte [26] for discrete, binary $(r|p)$-centroid and $(r|X_p)$-medianoid problems on networks with vertex demand only. These algorithms can be adapted to our case, since the objective function of the discrete, binary $(r|X_p)$-medianoid problem with vertex demand only remains submodular and non-decreasing when additionally incorporating edge demand.

Lemma 1. *The objective function of the discrete, binary $(r|X_p)$-medianoid problem with vertex and edge demand is submodular.*

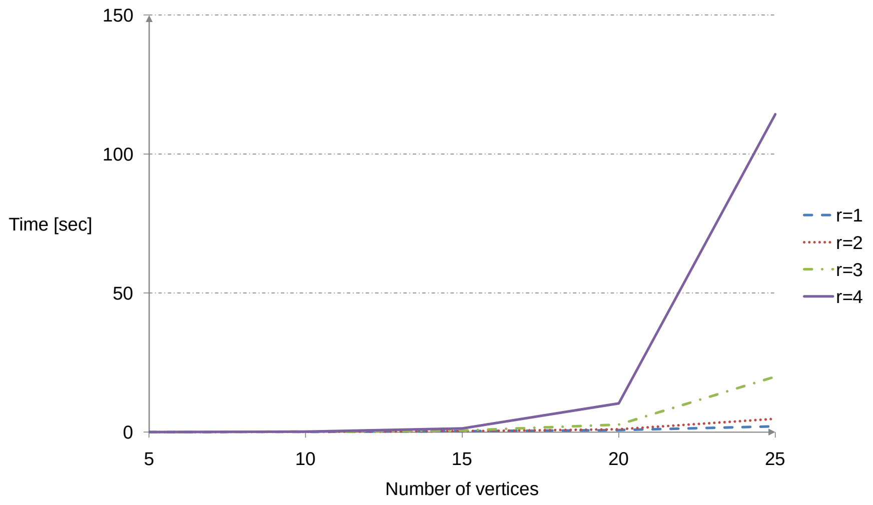


Figure 4: Computational results for the medianoid problem, $p = 2$.

Proof. Let Y_r be a solution to a discrete, binary $(r|X_p)$-medianoid problem with vertex and edge demand. Furthermore, let $S \subset T \subset Y_r$ and $k \in V \backslash T$. We want to show that $W_F(T \cup \{k\}|X_p) - W_F(T|X_p) \leq W_F(S \cup \{k\}|X_p) - W_F(S|X_p)$.

We define $\Delta_v(T,k)$ ($\Delta_v(S,k)$) to be the increase in the follower's market share that is induced by vertex customer $v \in V$ when augmenting T (S) with an additional facility in vertex k. Similarly, $\Delta_{[u,v]}(T,k)$ ($\Delta_{[u,v]}(S,k)$) is defined to be the increase in the follower's market share that is induced by the demand over edge $[u,v]$. Therefore, $W_F(T \cup \{k\}|X_p) - W_F(T|X_p) = \sum_{v \in V} \Delta_v(T,k) + \sum_{[u,v] \in E} \Delta_{[u,v]}(T,k)$ and $W_F(S \cup \{k\}|X_p) - W_F(S|X_p) = \sum_{v \in V} \Delta_v(S,k) + \sum_{[u,v] \in E} \Delta_{[u,v]}(S,k)$.

One of the following cases holds for every vertex $v \in V$:

1. $D(v,T) \geq D(v,X_p)$
   (a) and $d(v,k) \geq D(v,X_p)$,
   (b) and $d(v,k) < D(v,X_p)$,
2. $D(v,T) < D(v,X_p)$, $D(v,S) \geq D(v,X_p)$
   (a) and $d(v,k) \geq D(v,X_p)$,
   (b) and $d(v,k) < D(v,X_p)$, $d(v,k) \geq D(v,T)$,
   (c) and $d(v,k) < D(v,X_p)$, $d(v,k) < D(v,T)$,
3. $D(v,S) < D(v,X_p)$
   (a) and $d(v,k) \geq D(v,S)$,
   (b) and $d(v,k) < D(v,S)$, $D(v,S) = D(v,T)$,
   (c) and $d(v,k) < D(v,S)$, $D(v,S) > D(v,T)$.

Thus, for an arbitrary vertex $v \in V$, we have $\Delta_v(S,k) = \Delta_v(T,k) = 0$ for cases 1.a, 2.a, 3.a, 3.b and 3.c; $\Delta_v(S,k) = \Delta_v(T,k) = \pi(v)$ for case 1.b; $\Delta_v(S,k) = \pi(v) \geq$

$\Delta_v(T,k) = 0$ for cases 2.b and 2.c. Similarly, for an arbitrary edge $[u,v] \in E$, we have $\Delta_{[u,v]}(T,k) \leq \Delta_{[u,v]}(S,k)$. The related case differentiation is fairly simple but rather lengthy (36 possible combinations of cases 1.a – 3.c). Hence, for the sake of brevity, we leave it to the reader. $\square$

Obviously, if $S \subset T$, then $W_F(S|X_p) \leq W_F(T|X_p)$. This is a direct consequence of the non-negativity of the vertex demands and edge demand densities. Hence, analogously to Benati and Laporte [26], one may apply Theorem 9.3 of Nemhauser and Wolsey [27, Chap. III.3]. Therefore, $W_F^*(r|X_p) \leq (1 - [(r-1)/r]^r)^{-1} W_F^g(r|X_p)$, where $W_F^g(r|X_p)$ is the follower's market share obtained through a greedy heuristic.

Another promising approach might be the use of genetic algorithms, as the study by Jaramillo et al. [28] indicates (cf. also Arostegui Jr. et al. [29]).

3.3. Some complexity results

This section aims at extending some well known complexity results on $(r|p)$-centroid problems. Hakimi [20] provides a NP-hardness proof for the continuous, binary $(1|p)$-centroid problem on a general network with vertex demand only. Spoerhase and Wirth [16] have recently extended this result by showing that Hakimi's result remains true on pathwidth bounded graphs with vertex demand only.[1] Hence, we can immediately conclude:

Theorem 1. *The problem of finding a continuous, binary $(1|p)$-centroid of a network $N = (V, E, \lambda)$ with vertex and edge demand is NP-hard.*

Proof. The NP-hard [20] continuous, binary $(1|p)$-centroid problem with vertex demand only is a special case of the continuous, binary $(1|p)$-centroid problem with vertex and edge demand. $\square$

In what follows, we will show that the problem remains NP-hard even if we drop the vertex weights. This is in line with the results of Dasci et al. [14] for the continuous, binary $(r|X_1)$-medianoid problem with (vertex and) edge demand.

Theorem 2. *The problem of finding a continuous, binary $(1|p)$-centroid of a network $N = (V, E, \lambda)$ with edge demand only, i.e. $\pi(v) = 0$ for all $v \in V$, is NP-hard.*

Proof. The proof is in analogy to Hakimi [20] who considers networks with vertex demand only. The theorem is proven by reducing the *Vertex Cover* (VC) problem (cf. Garey and Johnson [30, p. 190]).

Vertex Cover Problem: Given a graph $G = (V, E)$ and an integer $p \leq |V|$, is there a subset $V' \subseteq V$ with $|V'| \leq p$ such that for each edge $e = [u, v] \in E$ at least one of u and v belongs to V'?

Consider an instance $I(VC)$ of the VC problem on graph $G = (V, E)$ and replace each edge $e_j = [u, v] \in E$ by the structure shown in Figure 5 ("diamond structure joining u and v" (Hakimi [20])) to construct a network $N_1 = (V_1, E_1)$ with the corresponding edge lengths λ and edge demand densities δ.

[1]See Spoerhase and Wirth [16] for a definition of pathwidth bounded graphs.

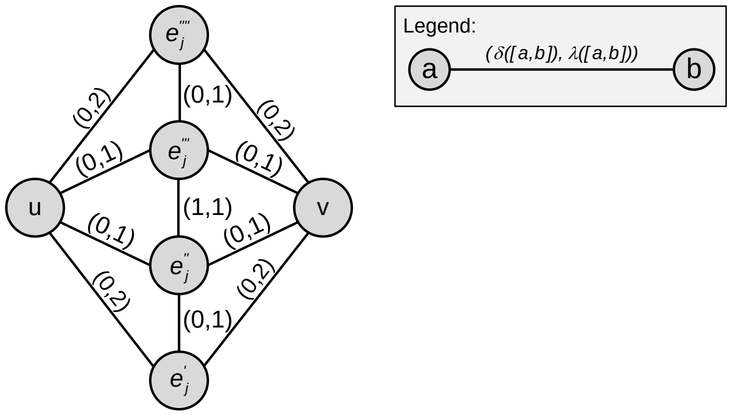


Figure 5: Diamond structure.

We will show that there exists a set of p points X_p on N_1 such that $W_F(Y_1(X_p)|X_p) \leq 1$ for every point $Y_1(X_p)$ on N_1 if and only if the VC instance has a solution. This will prove the theorem.

"If": Suppose V' is a solution of $I(VC)$ and let $X_p = V'$ on N_1. Then for any diamond structure joining u and v in N_1, either u, v or both belong to $V' = X_p$. Thus, we obviously have $W_F(Y_1(X_p)|X_p) \leq 1$ for any point $Y_1(X_p)$ on N_1 (recall that ties are broken in favor of the leader).

"Only if": Assume that a set of p points X_p on N_1 is such that $W_F(Y_1(X_p)|X_p) \leq 1$ for every point $Y_1(X_p)$ on N_1. If there exists at least one point of X_p on each diamond of N_1, then one can move these points to ($\leq p$) vertices $V' \subseteq V$ such that each diamond has at least one vertex in V'. Then V' is a solution to $I(VC)$. Therefore, we may assume that there exists a diamond structure in N_1 joining, say, u and v such that this diamond structure contains no point of X_p. Suppose, $\min\{D(u, X_p), D(v, X_p)\} > 2$. It is immediately implied that $W_F(u|X_p) \geq 2$ or $W_F(v|X_p) \geq 2$ so that we may assume $0 < \min\{D(u, X_p), D(v, X_p)\} \leq 2$. Without loss of generality, let $\min\{D(u, X_p), D(v, X_p)\} = D(u, X_p)$. Let us say there exists another diamond in N_1 that corresponds to edge $\hat{e} = [u, v'] \in E$ and thus joins u and v', where $v \neq v'$. If there is exactly one point $x_p \in X_p$ on this diamond, then we can always select $Y_1(X_p)$ to lie on an edge of this diamond incident to u such that $W_F(Y_1(X_p)|X_p) > 1$:

1. If x_p is a point on edge $[u, \hat{e}'''']$ (or, analogously, on edge $[u, \hat{e}']$) with $0 < d(u, x_p) < 2$, then we select $Y_1(X_p) = u$.

2. If x_p is a point on edge $[u, \hat{e}''']$ (or, analogously, on edge $[u, \hat{e}'']$) with $0 < d(u, x_p) < 1$, then we choose $Y_1(X_p) = u$ as well.

3. If x_p is a point on edge $[v', \hat{e}''']$ or $[\hat{e}''', \hat{e}'''']$ (including the vertices v', $\hat{e}'''$ and $\hat{e}''''$), then we select $Y_1(X_p)$ to lie on edge $[u, \hat{e}'']$ such that $d(u, Y_1(X_p)) < D(v, X_p)$. The case where x_p is a point on edge $[v', \hat{e}'']$ or $[\hat{e}', \hat{e}'']$ is treated analogously (symmetry of the diamond).

4. If x_p is a point on edge $[\hat{e}'', \hat{e}''']$ with $x_p \neq \hat{e}''$ and $x_p \neq \hat{e}'''$, then we select $Y_1(X_p) = \hat{e}''$ or $Y_1(X_p) = \hat{e}'''$.

Therefore, there must exist at least two points of X_p on the diamond structure joining u and v'. This implies that, if there exist no points of X_p on an arbitrary diamond structure of N_1, then there are at least two points of X_p on one "adjacent" diamond structure of N_1. Therefore, there are enough points to be moved to all diamonds of N_1 so that the existence of a solution to VC is established. □

The hardness results that have been subject of this section carry over to the discrete versions of the related $(1|p)$-centroid problems. Spoerhase and Wirth [16] explicitly show this for the discrete, binary $(1|p)$-centroid problem with vertex demand only on a pathwidth bounded graph so that, analogously to Theorem 1, we can conclude:

Theorem 3. *The problem of finding a discrete, binary* $(1|p)$*-centroid of a network* $N = (V, E, \lambda)$ *with vertex and edge demand is NP-hard.*

Moreover, we note that the proof of Theorem 2 can easily be adapted for the discrete, binary $(1|p)$-centroid problem with edge demand only. Therefore, we have:

Theorem 4. *The problem of finding a discrete, binary* $(1|p)$*-centroid of a network* $N = (V, E, \lambda)$ *with edge demand only is NP-hard.*

Proof. We augment the diamond structure of Figure 5 by inserting vertices e_j^1, e_j^2, e_j^3 and e_j^4 at the midpoints of the edges $[u, e_j^{''}]$, $[u, e_j^{'''}]$, $[v, e_j^{''}]$ and $[v, e_j^{'''}]$.

The proof of Theorem 2 remains unchanged apart from the case differentiation: If there lies exactly one point $x_p \in X_p$ on the diamond joining u and v', then we can always select $Y_1(X_p)$ to lie on a vertex of this diamond such that $W_F(Y_1(X_p)|X_p) > 1$:

1. If $x_p = v'$, $x_p = \hat{e}^1$, $x_p = \hat{e}^2$, $x_p = \hat{e}^3$, $x_p = \hat{e}^4$, $x_p = \hat{e}^{'}$ or $x_p = \hat{e}^{''''}$ then we select $Y_1(X_p) = u$.
2. If $x_p = \hat{e}^{''}$ then we select $Y_1(X_p) = \hat{e}^4$.
3. If $x_p = \hat{e}^{'''}$ then we select $Y_1(X_p) = \hat{e}^3$.

The conclusion of the proof of Theorem 2 is therefore still true. □

Table 1 summarizes the complexity results.

Table 1: NP-hard binary $(1|p)$-centroid problems.

type of demand	continuous	discrete
vertex demand only	NP-hard on general networks [20]	NP-hard on general networks [20]
	NP-hard on pathwidth bounded graphs [16]	NP-hard on pathwidth bounded graphs [16]
vertex and edge demand	NP-hard on general networks / pathwidth bounded graphs (implied)	
edge demand only	NP-hard on general networks (Theorem 2)	NP-hard on general networks (Theorem 4)

Polynomial time algorithms are due to Hansen and Labbé [21] (continuous, binary $(1|1)$-centroid problem on general networks with vertex demand only) and Spoerhase and Wirth [16] (discrete and continuous, binary $(1|p)$-centroid problems on tree networks with vertex demand only). We extend these results in the following section by providing an efficient algorithm for the discrete, binary $(1|p)$-centroid problem on tree networks with vertex and edge demand.

3.4. Finite dominating sets

A finite dominating set for the continuous, binary $(r|X_p)$-medianoid problem with vertex demand only has been introduced by Megiddo et al. [31]. The authors prove that there always exists an optimal solution at a set of so called boundary points (cardinality $O(nm)$) of the network. Dasci et al. [14] extend this result to the case of vertex and edge demand. They show that an instance of the continuous, binary $(r|X_p)$-medianoid problem with vertex and edge demand does not always possess an optimal solution. Therefore, they seek ϵ-optimal solutions, i.e. solutions that guarantee an objective function value at most ϵ units away from a known upper bound, and establish a finite dominating set of cardinality $O(nm)$ by presenting a simple augmentation of the boundary point set. One needs to additionally consider the set of all γ-points, i.e. points located γ units away from a vertex, where γ has to be sufficiently small.

Discretization results concerning $(r|p)$-centroid problems are rather limited in the literature (cf. Kress and Pesch [5]). An optimal solution to the continuous, binary $(r|1)$-centroid problem of a general network with vertex demand only is always a vertex for $r \geq 2$, while this is not the case for $r = 1$. If we restrict the network to be a tree, however, there always exists a vertex that is a $(1|1)$-centroid (see Hakimi [20] and the references therein). It is easy to see that the former result carries over to the case of vertex and edge demand, while the latter does not, i.e. there exist tree networks with vertex and edge demand without a $(1|1)$-centroid located in a vertex. Consider, for example, Figure 1 with $\pi(u) = \pi(v) = 0$ and $\lambda(uv) = \delta(uv) = 1$.

Proposition 1. *Let $r \geq 2$. Then there always exists a vertex $v \in V$ that is an optimal solution to the continuous, binary $(r|1)$-centroid problem with vertex and edge demand, defined on a network $N = (V, E, \lambda)$.*

The proof is straight forward. Assume that the leader locates in an arbitrary point $x \notin V$ of the network. Then there exists an (ϵ-) optimal solution to the corresponding medianoid problem with two follower's facilities located infinitesimally close to x such that the leader's gain is (arbitrarily close to) 0. Locating the leader's facility in a vertex $v \in V$, in contrast, will enforce the corresponding vertex customers to accommodate their demand at the leader's facility.

Spoerhase and Wirth [16] derive a finite dominating set for the continuous, binary $(1|p)$-centroid problem on tree networks with vertex demand only. They restrict $\lambda : E \to \mathbb{Q}^+$ and $\pi : V \to \mathbb{Q}_0^+$. Thus, they may assume without loss of generality that the edge lengths and vertex demands are integer numbers (cf. Section 4). The authors show that there always exists an optimal solution to the leader's problem, such that the distance from any of the leader's facilities to any vertex is an integer number or an integer number divided by two. Note that this discretization result is not polynomial. While, as described above, Dasci et al. [14] are able to show that the discretization result by Megiddo et al. [31] almost directly applies when additionally considering edge demand, the same is implausible for the finite dominating set just described for the continuous, binary $(1|p)$-centroid problem on tree networks. Consider, for instance, a chain network $N = (V, E, \lambda)$ with $V = \{1, 2, 3\}$, $E = \{[1, 2], [2, 3]\}$, $\delta([1, 2]) = 1$, $\delta([2, 3]) = 2$, $\lambda([1, 2]) = \lambda([2, 3]) = 1$ and $\pi(1) = \pi(2) = \pi(3) = 0$. It is easy to see that the unique $(1|1)$-centroid x is located on edge $[2, 3]$ with $d(1, x) = 1.25$. Moreover, Spoerhase and Wirth [16] show that, on a chain network, the continuous version of the binary $(r|p)$-centroid problem with vertex demand only is NP-hard, while the discrete version is not. Hence, they conjecture that a polynomial finite dominating set is unlikely to exist for the continuous, binary $(r|p)$-centroid problem on general networks with vertex demand only. This holds for the case of vertex and edge demand as well. Hence, we leave the question of whether or not there exists a more general discretization result than the one given in Proposition 1 for future research and turn our attention to the discrete problem class.

4. An algorithm for the discrete, binary (1|p)-centroid problem on tree networks with vertex and edge demand

Before describing the algorithm itself in Section 4.2, we will introduce the basic ideas by restricting our attention to chain networks in Section 4.1. We impose an additional assumption on the networks under consideration, i.e. we assume $\lambda : E \to \mathbb{Q}^+$, $\delta : E \to \mathbb{Q}_0^+$ and $\pi : V \to \mathbb{Q}_0^+$. As a direct consequence, we may assume without loss of generality that the edge lengths, demand densities and vertex demands are integer numbers for the remainder of this paper: Let all edge lengths, demand densities and vertex demands be expressed as fractions of two integers and define c to be the least common multiple of their denominators. Then we can transform N into a network $N' = (V, E, \lambda')$ with $\lambda'(uv) = c\lambda(uv)$, $\delta'(uv) = c\delta(uv)$ for all $[u, v] \in E$ and $\pi'(u) = c^2\pi(u)$ for all $u \in V$ without changing the ratio of the market shares of leader and follower for any feasible location setting.

4.1. Chain networks

Consider a chain network $N = (V, E, \lambda)$ with vertex set $V = \{u_1, ..., u_n\}$ and edge set $E = \{[u_i, u_{i+1}]|i = 1, ..., n - 1\}$ (see Figure 6). Global variables x and t allow the definition of any point of the chain network. For the sake of notational convenience, we will denote the vertices by natural numbers, i.e. i instead of u_i, whenever possible. The distinction of vertex numbers and distinct values of the global variables x and t will, in general, become clear from the context.

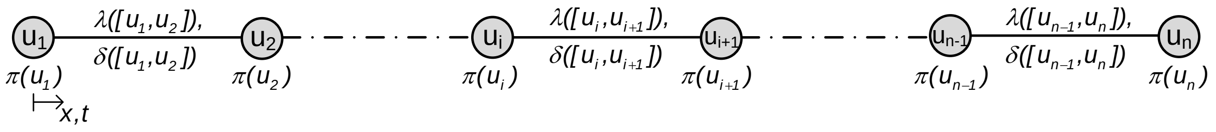


Figure 6: Chain network.

We define

$$f(x) := \sum_{i \in V, d(1,i) < x} \pi(i) + \int_0^x \delta(t)dt, \tag{29}$$

where $\delta(t) := 0$ for all $t < 0$ and $t > d(1, n)$.

This results in

$$f(x) = \begin{cases} 0 & \text{if } x \leq 0, \\ \pi(1) + \delta([1,2])x & \text{if } 0 < x \leq d(1,2), \\ \vdots & \vdots \\ \displaystyle\sum_{i=1}^{u-2} [\delta([i,i+1])\lambda([i,i+1]) + \pi(i)] & \\ \qquad + \pi(u-1) + \delta([u-1,u])(x - d(1,u-1)) & \text{if } d(1,u-1) < x \leq d(1,u), \\ \vdots & \vdots \\ \displaystyle\sum_{i=1}^{n-2} [\delta([i,i+1])\lambda([i,i+1]) + \pi(i)] & \\ \qquad + \pi(n-1) + \delta([n-1,n])(x - d(1,n-1)) & \text{if } d(1,n-1) < x \leq d(1,n), \\ \sum_{i=1}^{n-1} [\delta([i,i+1])\lambda([i,i+1]) + \pi(i)] + \pi(n) & \text{if } x > d(1,n). \end{cases} \tag{30}$$

Note that, after applying a simple preprocessing procedure of linear time complexity, one can evaluate $f(x)$ in constant time for any x. Figure 7 gives an example.

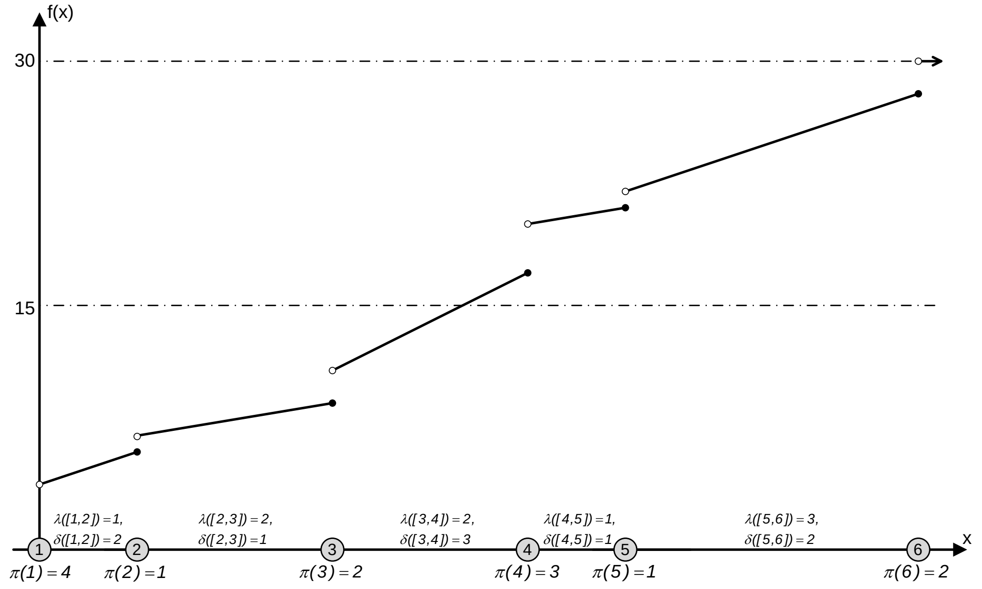


Figure 7: Function $f(x)$.

Let the leader's locations be the vertices j and k with $j \leq k$ and let the follower's location be vertex $i \neq j,k$. Then we can easily determine an open interval $]a,b[$ with $a < b$ and $a,b \in [0, d(1,n)]$ of customers, who accommodate their demand at the follower's facility:

$$a = \begin{cases} 0 & \text{if } i < j, \\ \frac{1}{2}(d(1,j) + d(1,i)) & \text{if } j < i < k, \\ \frac{1}{2}(d(1,k) + d(1,i)) & \text{if } i > k, \end{cases} \tag{31}$$

$$b = \begin{cases} \frac{1}{2}(d(1,j) + d(1,i)) & \text{if } i < j, \\ \frac{1}{2}(d(1,k) + d(1,i)) & \text{if } j < i < k, \\ d(1,n) & \text{if } i > k. \end{cases} \tag{32}$$

The customers located at a and b have to be considered separately, so that the follower's market share can be calculated as follows:

$$W_F(i|j,k) = \begin{cases} f(b) - f(a) & \text{if } i < j, \\ f(b) - f(a) - \pi(a) & \text{if } j < i < k, \\ f(b) - f(a) - \pi(a) + \pi(n) & \text{if } i > k. \end{cases} \tag{33}$$

Thus, we can state:

Lemma 2. *The problem of finding a discrete, binary $(1|X_2)$-medianoid of a chain network $N = (V, E, \lambda)$ with vertex and edge demand can be solved in linear time $O(n)$.*

Proof. An algorithm is as follows. It determines (not necessarily all) $(1|X_2)$-medianoids, stored in Y, as well as the corresponding market share of the follower, ξ_s^*.

Algorithm 4.1 discrete, binary $(1|X_2)$-medianoid of chain network with vertex and edge demand

```
1: $Y := \emptyset$
2: $\xi_s := 0$, $\xi_s^* := 0$
3: for $i = j - 1$ to $k + 1$, $i \neq j$, $i \neq k$, $i \neq 0$, $i \neq n + 1$ do
4:    $\xi_s = W_F(i|j,k)$ (Equations (31)-(33))
5:    if $\xi_s > \xi_s^*$ then
6:       $Y = \{i\}$
7:       $\xi_s^* = \xi_s$
8:    else if $\xi_s = \xi_s^*$ then
9:       $Y = Y \cup \{i\}$
10:   end if
11: end for
12: if $Y = \emptyset$ then
13:    $Y = Y \cup \{1\}$
14: end if
```

□

We are now able to derive an efficient algorithm for the centroid problem by combining Algorithm 4.1 and the definition of $\hat{\xi}$-bounding sets, presented by Spoerhase and Wirth [16] for the case of vertex demand only (see section 2). To this end we add an artificial vertex $n+1$ with $\pi(n+1) = 0$ and an artificial edge $[n, n+1]$ with $\lambda([n, n+1]) = \infty$ and $\delta([n, n+1]) = 0$ to the chain network.

Lemma 3. *Let $0 < p \leq n$ with $p \in \mathbb{N}$ and $0 \leq \hat{\xi} \leq \xi(N)$ with $\hat{\xi} \in \mathbb{R}$. If, for a chain network $N = (V, E, \lambda)$ with vertex and edge demand, there exists a $\hat{\xi}$-bounding set with at most p elements, we can determine one such set (or decide that there is no such set) in $O(n^2)$ time.*

Proof. Observe that the cases $\hat{\xi} = 0$ and $\hat{\xi} = \xi(N)$ are trivial, i.e. we can easily decide if a $\hat{\xi}$-bounding set exists and if so, it is a simple matter to find such a set. Hence, we assume $0 < \hat{\xi} < \xi(N)$ in the following.

Algorithm 4.2 determines a set X of at most p elements that form a $\hat{\xi}$-bounding set in the proposed running time. If there is no such set, the algorithm terminates with $X = \emptyset$.

Algorithm 4.2 $\hat{\xi}$-bounding set X with $|X| \le p$ of chain network with vertex and edge demand

```
 1: $X := \emptyset$
 2: $\xi_s := \pi(1) + \frac{1}{2}\delta([1,2])\lambda([1,2])$
 3: $i := 2$
 4: while $\xi_s \le \hat{\xi}$ and $i \le n-1$ do
 5:    $i = i + 1$
 6:    $\xi_s = \xi_s + \frac{1}{2}\delta([i-2,i-1])\lambda([i-2,i-1]) + \pi(i-1) + \frac{1}{2}\delta([i-1,i])\lambda([i-1,i])$
 7: end while
 8: if $i = n$ and $\xi_s \le \hat{\xi}$ then
 9:    $X := X \cup \{i\}$
10: else
11:    $X = X \cup \{i-1\}$
12: end if
13: while $|X| \le p$ and $i \le n$ do
14:    $\xi_s = 0$
15:    while $\xi_s \le \hat{\xi}$ and $i \le n$ do
16:       $i = i + 1$
17:       Call Algorithm 4.1 on subnetwork $N'$ with vertex set $V' = \{\max\{l|l \in X\}, ..., i\}$
          and $j = \max\{l|l \in X\}$, $k = i$. It results in $\xi_s^*$ and a set $Y$.
18:       $\xi_s = \xi_s^*$
19:    end while
20:    if $\xi_s > \hat{\xi}$ then
21:       $X = X \cup \{i-1\}$
22:    end if
23: end while
24: if $|X| = p + 1$ then
25:    $X := \emptyset$
26: end if
```

□

Obviously, the set of all $(1|p)$-centroids of a chain network with follower gain $\hat{\xi}$ contains all $\hat{\xi}$-bounding sets with at most p elements. Thus, we may state the main result of this section:

Theorem 5. *The problem of finding a discrete, binary $(1|p)$-centroid of a chain network $N = (V, E, \lambda)$ with vertex and edge demand can be solved with time complexity $O(n^2 \log \xi(N))$.*

Proof. We apply bisection on the interval $[0, \xi(N)]$ to find the smallest value $\hat{\xi}$ such that a $\hat{\xi}$-bounding set with at most p elements exists. Observe that (due to the additional assumption in Section 4) we can terminate the bisection method when $|\xi_{best} - \xi_{nobs}| < 0.5$, where ξ_{best} corresponds to the follower's market share in the best feasible solution found so far, and ξ_{nobs} corresponds to the largest demand level in an iteration of the bisection method that resulted in the nonexistence of a ξ_{nobs}-bounding set with at most p elements. This results in the proposed running time. □

4.2. General tree networks

Suppose that the given network $N = (V, E, \lambda)$ is a tree network with vertex set $V = \{u_1, ..., u_n\}$ and edge set E. We will denote the vertices by natural numbers, i.e. i instead of u_i, whenever possible. As in the case of a chain network, the set of all $(1|p)$-centroids of a tree network with follower gain $\hat{\xi}$ contains all $\hat{\xi}$-bounding sets with at most p elements. As we will show in the following, we can extend the ideas presented in Section 4.1 to determine a discrete, binary $(1|p)$-centroid of a tree network.

Add an artificial vertex $n+1$ with $\pi(n+1) = 0$ as the root of N and connect it to an arbitrary vertex $s \in V$ by an artificial edge $[s, n+1]$ with $\lambda([s, n+1]) = \infty$ and $\delta([s, n+1]) = 0$. A $\hat{\xi}$-bounding set with at most p elements can be determined by performing a depth first search traversal of the tree network (cf. Spoerhase and Wirth [16]).

Algorithm 4.3 $\hat{\xi}$-bounding set X with $|X| \leq p$ of tree network with vertex and edge demand

```
1: $X := \emptyset$
2: Perform a depth first search traversal of the tree network, starting at the artificial
   vertex $n+1$. Whenever the search tracks back from vertex $v$ to vertex $u$, solve a $(1|X_{\hat{p}})$-
   medianoid problem on the $X$-subtracted subtree rooted in $u$ with $X_{\hat{p}} = (X \cap V_{X_s}) \cup \{u\}$.
   If the follower's optimal market share exceeds $\hat{\xi}$, set $X = X \cup \{v\}$.
3: if $|X| > p$ then
4:     $X = \emptyset$
5: end if
```

Suppose for now that we have an appropriate algorithm to solve discrete, binary $(1|X_p)$-medianoid problems on tree networks at hand. Then the main idea of how to find a discrete, binary $(1|p)$-centroid is in analogy to Section 4.1, i.e. we apply bisection on the interval $[0, \xi(N)]$ to find the smallest value $\hat{\xi}$ such that a $\hat{\xi}$-bounding set with at most p elements exists. The following example illustrates this basic idea. Suppose we want to solve a $(1|2)$-centroid problem on the tree network depicted in Figure 8, with all edge lengths, demand densities and vertex demands equal to one. We arbitrarily choose $s = 1$. Figure 9 highlights and numbers all the subtrees that we will have to consider in the solution process. Shaded vertices represent leader's facilities. Table 2 lists corresponding optimal solutions to the medianoid problems.

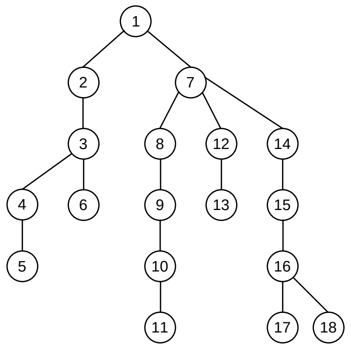


Figure 8: Example - tree network.

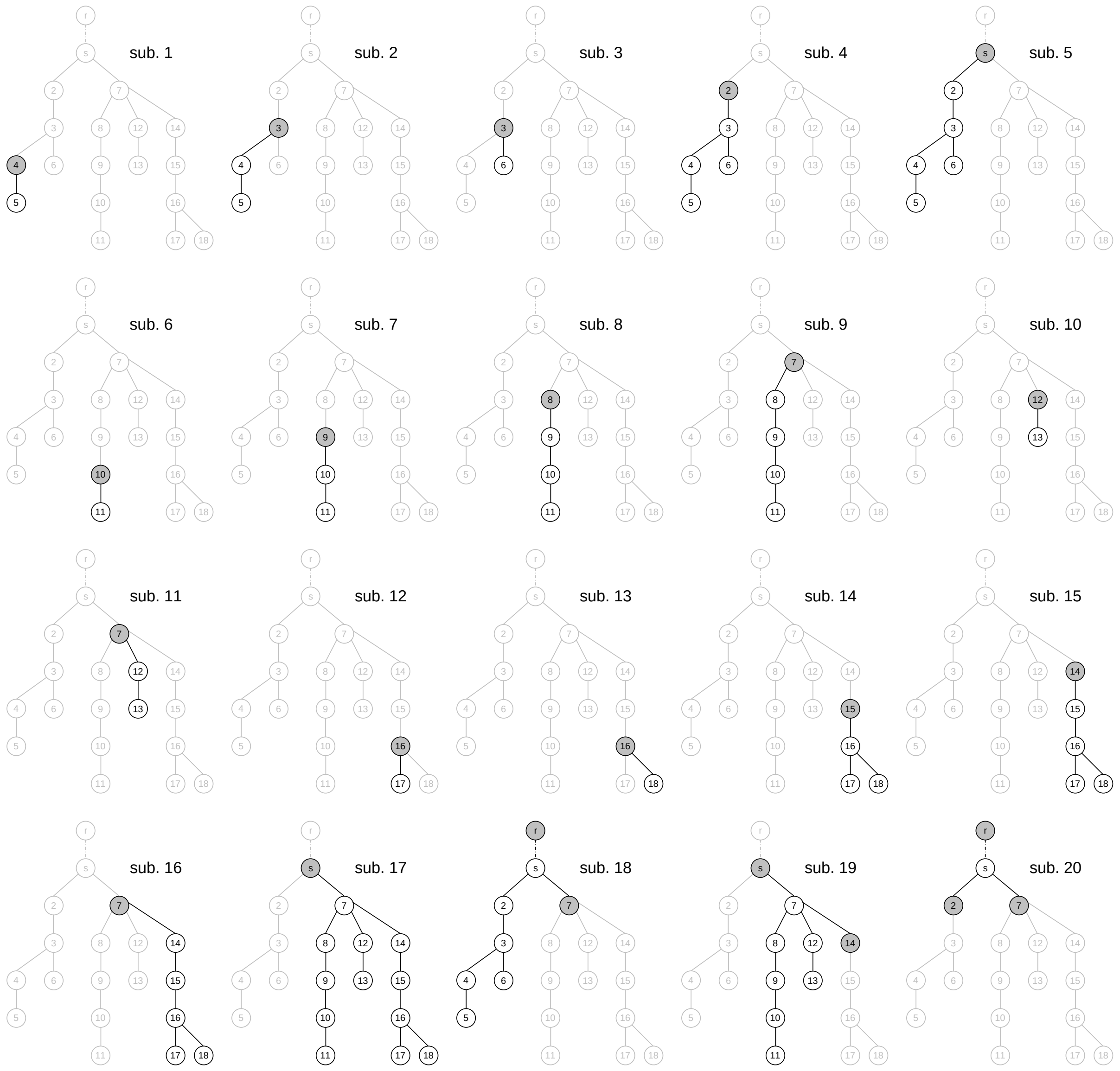


Figure 9: Example - subtrees.

Table 2: Example - optimal solutions to the medianoid problems.

Subnetwork	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Opt. solution	$\{5\}$	$\{4\}$	$\{6\}$	$\{3\}$	$\{2\}$	$\{11\}$	$\{10\}$	$\{9\}$	$\{8\}$	$\{13\}$	$\{12\}$	$\{17\}$	$\{18\}$	$\{16\}$	$\{15\}$	$\{14\}$	$\{7\}$	$\{s\}$	$\{7\}$	$\{s\}$
Follower gain	1.5	3.5	1.5	7.5	9.5	1.5	3.5	5.5	7.5	1.5	3.5	1.5	1.5	5.5	7.5	9.5	23.5	11.5	14	1

Table 3 walks through the solution process as described above. The termination criterion, $\hat{\xi}_7 - \hat{\xi}_6 < 0.5$, is applicable due to the integrality of the edge lengths, vertex demands and demand densities (see below).

Table 3: Example - solving the centroid problem.

Bisection method	Algorithm 4.3: subnetworks under consideration	Bounding set
$\hat{\xi}_1 = 0.5\xi(N) = 17.5$	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17 $\rightarrow X = \{7\}$ 18	$X = \{7\}$
$\hat{\xi}_2 = 8.75$	1, 2, 3, 4, 5 $\rightarrow X = \{2\}$ 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16 $\rightarrow X = \{2, 14\}$ 19 $\rightarrow X = \{2, 7, 14\}$ 20 $\rightarrow \lvert X\rvert > 2$	$X = \emptyset$
$\hat{\xi}_3 = 13.125$	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17 $\rightarrow X = \{7\}$ 18	$X = \{7\}$
$\hat{\xi}_4 = 10.9375$	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17 $\rightarrow X = \{7\}$ 18 $\rightarrow X = \{1, 7\}$	$X = \{1, 7\}$
$\hat{\xi}_5 = 9.84375$	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17 $\rightarrow X = \{7\}$ 18 $\rightarrow X = \{1, 7\}$	$X = \{1, 7\}$
$\hat{\xi}_6 = 9.296875$	1, 2, 3, 4, 5 $\rightarrow X = \{2\}$ 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16 $\rightarrow X = \{2, 14\}$ 19 $\rightarrow X = \{2, 7, 14\}$ 20 $\rightarrow \lvert X\rvert > 2$	$X = \emptyset$
$\hat{\xi}_7 = 9.5703125$	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17 $\rightarrow X = \{7\}$ 18 $\rightarrow X = \{1, 7\}$	$X = \{1, 7\}$
	STOP ($\hat{\xi}_7 - \hat{\xi}_6 < 0.5$) $X_2 = \{1, 7\}$ is a $(1\|2)$-centroid	

Hence, the medianoid problems to be solved are of a special structure (Figure 9): The leader's facilities are solely located in leaves and roots of subtrees of the original network. We will now describe an algorithm to optimally solve those special structure $(1|X_{\hat{p}})$-medianoid problems. It results in the following lemma.

Lemma 4. *Let $\hat{N} = (\hat{V}, \hat{E}, \hat{\lambda})$, $|\hat{V}| = \hat{n}$, be a rooted tree network with vertex and edge demand. Moreover, let $X_{\hat{p}} \subseteq \hat{L} \cup \{\hat{r}\}$, $X_{\hat{p}} \cap \{\hat{r}\} \neq \emptyset$, where $\hat{L}$ denotes the set of leaves and*

$\hat{r}$ corresponds to the root of the tree network with degree 1. Then the problem of finding a discrete, binary $(1|X_{\hat{p}})$-medianoid can be solved with time complexity $O(\hat{n}\hat{p})$.

The algorithm is composed of six stages.

1. Collapse, $O(\hat{n})$
2. Label, $O(\hat{n})$
3. Rearrange, $O(\hat{p}^2)$
4. Construct chain functions, $O(\hat{n}\hat{p})$
5. Calculate distances of facilities in X_p, $O(\hat{p}^2)$
6. Evaluate, $O(\hat{n}\hat{p})$

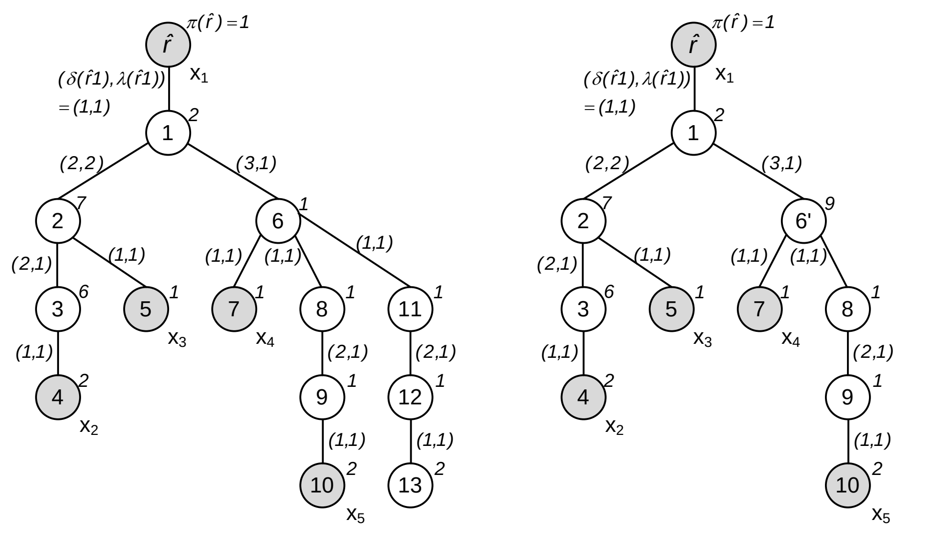
In the following we will describe each of the six stages in detail. We denote the elements of the set $X_{\hat{p}}$ by x_i, $i = 1, ..., \hat{p}$, where, without loss of generality, x_1 corresponds to the leader's facility located in the root $\hat{r}$.

Collapse: Consider a leaf $x \notin X_{\hat{p}}$ and let $x' \neq \hat{r}$ be the father of x. It is easy to see that $W_F(x|X_{\hat{p}}) \leq W_F(x'|X_{\hat{p}})$. Therefore, we may construct an auxiliary tree network N' by merging x and x' to define a vertex $v_{x,x'}$ with $\pi(v_{x,x'}) = \pi(x) + \pi(x') + \lambda([x', x])\delta([x', x])$ to replace x'. Any $(1|X_{\hat{p}})$-medianoid on N' corresponds to a $(1|X_{\hat{p}})$-medianoid on N. The transformation is straight forward: Let $v_{x,x'}$ be an optimal solution to the $(1|X_{\hat{p}})$-medianoid problem on N'; then x' is the corresponding $(1|X_{\hat{p}})$-medianoid on N. This idea of merging vertices can be repeated until every leaf of the resulting auxiliary tree network is either a child of $\hat{r}$ or an element of $X_{\hat{p}}$. This can be achieved by performing a depth first search traversal of the tree network of time complexity $O(\hat{n})$. An example with $X_{\hat{p}} = \{x_1, x_2, x_3, x_4, x_5\}$ is given in Figure 10. Since vertex 13 is a leaf without being a leader's location, we may merge vertices 13 and 12. The resulting vertex can subsequently be merged with vertices 11 and 6.

Observe that for $\hat{p} \leq 2$ we are left with a chain network (see Section 4.1). Therefore, in what follows, we will restrict our attention to the nontrivial case $\hat{p} > 2$. Moreover, to ease notation, we will omit the superscript $'$ when referring to the collapsed tree network and its vertex set, i.e. we will keep using the notation that has been introduced in Lemma 4.

Label: It is well known that the vertices of a tree network can be labeled in linear time such that hereafter the nearest common ancestor of any pair of vertices as well as their distance can be computed in constant time. See, for example, Alstrup et al. [32] and the references therein.

Rearrange: This stage aims at finding a permutation $[x_{\mu_1}, ..., x_{\mu_{\hat{p}-1}}]$ of the elements x_i, $i = 2, ...\hat{p}$. Roughly speaking, this permutation is such that the customers located on the unique path that connects a superordinated vertex of the permutation to the root are guaranteed not to accommodate their demand at a leader's facility located at a subordinated element of the permutation (or to be indifferent about accommodating their demand at one of the two corresponding leader's facilities). Additionally, in combination with the subsequent stages of the algorithm, the permutation eventually guarantees that each customer accommodates his demand at a single facility by grouping the leader's facilities according to their nearest common ancestors. The permutation is being determined by applying a three stage algorithm:



(a) Original tree network N. (b) Auxiliary tree network N'.

Figure 10: Collapsing stage.

1. Calculate $d(\hat{r}, x_i)$ for all $i = 2, \ldots \hat{p}$ by use of the vertex labels (Stage 2) (time complexity $O(\hat{p})$).
2. Sort the elements x_i, $i = 2, \ldots \hat{p}$, in the order of increasing distances $d(\hat{r}, x_i)$. We may, for example, apply mergesort (time complexity $O(\hat{p} \log \hat{p})$, cf. Cormen et al. [33]). The sorting algorithm results in an array L of length $\hat{p} - 1$ with the smallest element in the first position, denoted by $L[1]$.
3. Call Algorithm 4.4 to determine the desired permutation P. We denote the i-th position of the permutation P by $P[i]$. Using a linked list data structure, the algorithm has time complexity $O(\hat{p}^2)$.

Algorithm 4.4 Generate rearranged facilities.

```
initialize $P$
$pos := 0$, $lastpos := 2$, $depth := 0$
$P[1] = L[1]$, $P[2] = L[2]$
**for** $i = 3$ to $\hat{p} - 1$ **do**
   $pos = 0$
   $depth = d(\hat{r}, nca(L[i], P[1]))$
   **for** $j = 2$ to $lastpos$ **do**
      **if** $d(\hat{r}, nca(L[i], P[j])) > depth$ **then**
         $depth = d(\hat{r}, nca(L[i], P[j]))$
         $pos = j + 1$
      **end if**
   **end for**
   **if** $pos = 0$ **then**
      **if** $d(\hat{r}, nca(L[i], P[1])) > d(\hat{r}, nca(L[i], P[2]))$ **then**
         $pos = 2$
      **else**
```

```
17:         $pos = lastpos + 1$
18:       **end if**
19:     **end if**
20:     $P[pos] = L[i]$
21:     $lastpos = lastpos + 1$
22: **end for**
```

Let us denote the unique path that connects x_{μ_j} to the root $\hat{r}$ by c_{μ_j} for each $j = 1, ..., \hat{p} - 1$. Then the three stage rearranging process is such that for any pair $k, l \in \{1, ..., \hat{p} - 1\}$ with $k < l$ there exists a μ_m, $m \le k$ such that $d(x_{\mu_m}, z) \le d(x_{\mu_l}, z)$ for all points z on c_{μ_k}, even if $d(x_{\mu_l}, \hat{r}) < d(x_{\mu_k}, \hat{r})$.

We conclude by revisiting the example of Figure 10b in Table 4.

Table 4: Rearranging stage for the example in Figure 10b.

Steps 1 and 2:	distances:	sorted distances:
	i: 2, 3, 4, 5	i: 4, 3, 2, 5
	$d(\hat{r}, x_i)$: 5, 4, 3, 5	$d(\hat{r}, x_i)$: 3, 4, 5, 5
Alg. 4.4, initialization:	$L = [x_4, x_3, x_2, x_5]$	$P = [x_4, x_3]$
Alg. 4.4, $i = 3$:	insert x_2:	
	$nca(x_4, x_2) = 1$, $d(\hat{r}, 1) = 1$	
	$nca(x_3, x_2) = 2$, $d(\hat{r}, 2) = 3$	$P = [x_4, x_3, x_2]$
Alg. 4.4, $i = 4$:	insert x_5:	
	$nca(x_4, x_5) = 6'$, $d(\hat{r}, 6') = 2$	
	$nca(x_3, x_5) = 1$, $d(\hat{r}, 1) = 1$	
	$nca(x_2, x_5) = 1$, $d(\hat{r}, 1) = 1$	$P = [x_4, x_5, x_3, x_2]$

When inserting x_2, the algorithm compares the "depth" of the nearest common ancestors of x_2 and the facilities x_4 and x_3 in the tree network. Since the latter nearest common ancestor is located deeper in the network, x_2 is inserted after x_3. Similarly, x_5 needs to be paired with facility x_4. Thus, the procedure terminates with $x_{\mu_2} = x_5$ and $x_{\mu_3} = x_3$ although $d(\hat{r}, x_3) = 4 < d(\hat{r}, x_5) = 5$ because x_4 will always be closer to any point on chain $[\hat{r}, 1, 6', 8, 9, 10]$ than x_3.

Construct chain functions: As in the case of chain networks (Section 4.1), we can now define "chain functions" $f_{\mu_i}(y_{\mu_i})$ for all $i = 1, ..., \hat{p} - 1$, where y_{μ_i} are local variables that correspond to the distance of a point on chain c_{μ_i} to the root $\hat{r}$ of the collapsed tree network. In the following we will denote the vertices of a chain c_{μ_i} in the sequence of increasing distance to the root $\hat{r}$ by $v^1_{\mu_i}, ..., v^{l_{\mu_i}+1}_{\mu_i}$, where l_{μ_i} is the number of edges on the unique path from $\hat{r}$ to x_{μ_i}. In contrast to Section 4.1 we will now have to assign each vertex/edge customer to a unique chain. We do so by considering the permutation P that results from the rearranging stage. For any $i > 1$, we denote the vertex $nca(x_{\mu_{i-1}}, x_{\mu_i})$ by

$v^s_{\mu_i}$, where $1 \leq s \leq l_{\mu_i}$, and we define

$$f_{\mu_i}(y_{\mu_i}) := \begin{cases} 0 & \text{if } y_{\mu_i} \leq d(\hat{r}, v^s_{\mu_i}), \\ \delta([v^s_{\mu_i}, v^{s+1}_{\mu_i}])(y_{\mu_i} - d(\hat{r}, v^s_{\mu_i})) & \text{if } d(\hat{r}, v^s_{\mu_i}) < y_{\mu_i} \leq d(\hat{r}, v^{s+1}_{\mu_i}), \\ \vdots & \vdots \\ \begin{aligned} &\sum_{j=s}^{u-1} [\delta([v^j_{\mu_i}, v^{j+1}_{\mu_i}])\lambda([v^j_{\mu_i}, v^{j+1}_{\mu_i}]) + \pi(v^j_{\mu_i})] \\ &\quad - \pi(v^s_{\mu_i}) + \pi(v^u_{\mu_i}) \\ &\quad + \delta([v^u_{\mu_i}, v^{u+1}_{\mu_i}])(y_{\mu_i} - d(\hat{r}, v^u_{\mu_i})) \end{aligned} & \text{if } d(\hat{r}, v^u_{\mu_i}) < y_{\mu_i} \leq d(\hat{r}, v^{u+1}_{\mu_i}), \\ \vdots & \vdots \\ \begin{aligned} &\sum_{j=s}^{l_{\mu_i}-1} [\delta([v^j_{\mu_i}, v^{j+1}_{\mu_i}])\lambda([v^j_{\mu_i}, v^{j+1}_{\mu_i}]) + \pi(v^j_{\mu_i})] \\ &\quad - \pi(v^s_{\mu_i}) + \pi(v^{l_{\mu_i}}_{\mu_i}) \\ &\quad + \delta([v^{l_{\mu_i}}_{\mu_i}, v^{l_{\mu_i}+1}_{\mu_i}])(y_{\mu_i} - d(\hat{r}, v^{l_{\mu_i}}_{\mu_i})) \end{aligned} & \text{if } d(\hat{r}, v^{l_{\mu_i}}_{\mu_i}) < y_{\mu_i} \leq d(\hat{r}, v^{l_{\mu_i}+1}_{\mu_i}), \\ \begin{aligned} &\sum_{j=s}^{l_{\mu_i}} [\delta([v^j_{\mu_i}, v^{j+1}_{\mu_i}])\lambda([v^j_{\mu_i}, v^{j+1}_{\mu_i}]) + \pi(v^j_{\mu_i})] \\ &\quad - \pi(v^s_{\mu_i}) + \pi(v^{l_{\mu_i}+1}_{\mu_i}) \end{aligned} & \text{if } d(\hat{r}, v^{l_{\mu_i}+1}_{\mu_i}) < y_{\mu_i}. \end{cases} \tag{34}$$

For $i = 1$ we define $v^s_{\mu_1} := \hat{r}$ and $f_{\mu_1}(y_{\mu_1})$ in analogy to equation (30).

A simple $O(\hat{n}\hat{p})$ procedure to construct the chain functions determines the chains c_{μ_i} for all $i = 1, ..., \hat{p} - 1$ by performing a depth first search traversal of the collapsed tree network and, afterwards, generates the needed coefficients by considering all edges of the collapsed tree network at most $\hat{p}$ times. After having constructed a chain function c_{μ_i}, we can evaluate it in constant time for a given y_{μ_i}.

For our example, the chain functions $f_{\mu_1}(y_{\mu_1}), ..., f_{\mu_4}(y_{\mu_4})$ for $P = [x_4, x_5, x_3, x_2]$ are depicted in Figure 11. Since, for instance, $nca(x_4, x_5) = 6'$, we have $f_{x_5}(y_{x_5}) = 0$ for all $y_{x_5} \leq d(\hat{r}, 6')$. Hence, the vertex demand of the vertices $\hat{r}$ and 1 as well as the edge demand of the edges $[\hat{r}, 1]$ and $[1, 6']$ is uniquely assigned to chain c_{x_4}.

Calculate distances of facilities in $\mathbf{X_p}$: Determine

$$D(x_{\mu_i}) := \min\{d(\hat{r}, x_{\mu_i}), \min\{d(x_{\mu_i}, x_{\mu_j}) | j < i\}\} \tag{35}$$

for all $i = 2, ..., \hat{p} - 1$ and denote one of the corresponding vertices by $V(D(x_{\mu_i}))$. This can obviously be achieved in $O(\hat{p}^2)$ time. Moreover, we equivalently define $D(x_{\mu_1}) := d(\hat{r}, x_{\mu_1})$ and $V(D(x_{\mu_1})) := \hat{r}$. The example is revisited in Table 5.

Evaluate: Consider a vertex $v \in V \setminus \{X_{\hat{p}}\}$ of the collapsed tree network as a potential follower's location and let $k = \min\{i | v \text{ is a vertex on chain } c_{\mu_i}, i \in \{1, ..., \hat{p} - 1\}\}$. Then we can easily determine an open interval $]a_{c_{\mu_j}}, b_{c_{\mu_j}}[$ with $a_{c_{\mu_j}} \leq b_{c_{\mu_j}}$ of customers who accommodate their demand at v for each chain c_{μ_j}, $j = 1, ..., \hat{p} - 1$ in analogy to Section 4.1.

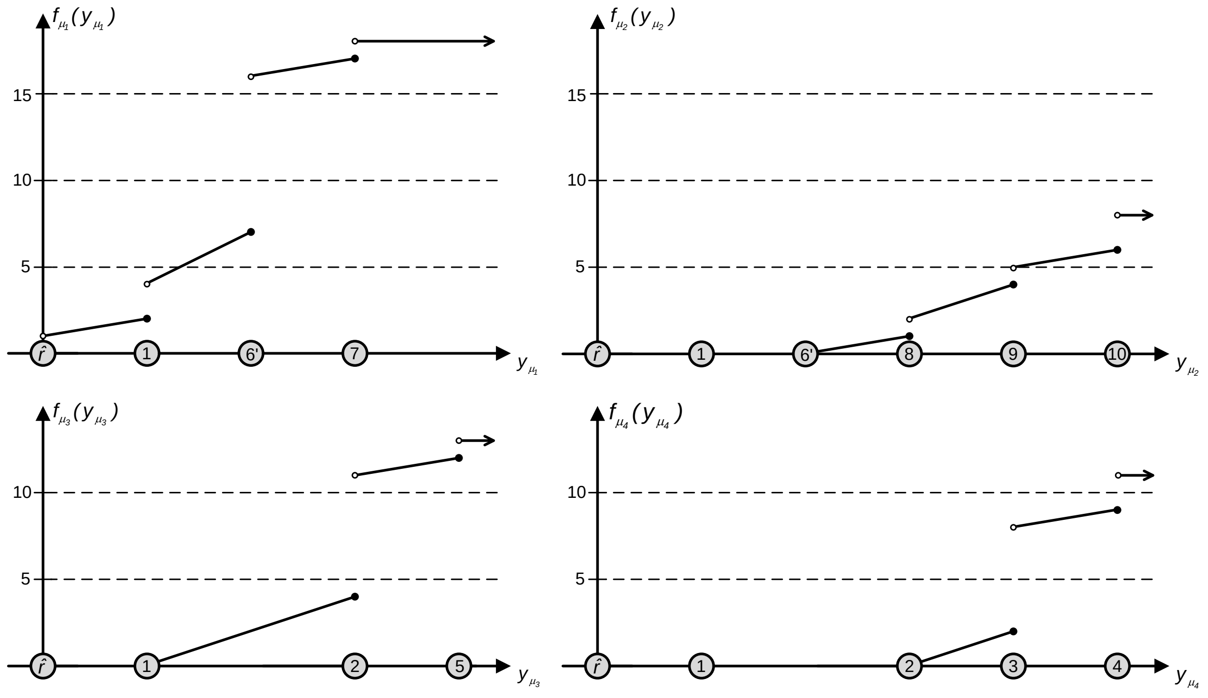


Figure 11: Functions $f_{\mu_i}(y_{\mu_i})$, $i = 1, ..., 4$, for the example in Figure 10b.

Table 5: $D(x_{\mu_i})$ and $V(D(x_{\mu_i}))$, $i = 1, ..., 4$, for the example in Figure 10b.

i	1	2	3	4
x_{μ_i}	x_4	x_5	x_3	x_2
$D(x_{\mu_i})$	3	4	4	3
$V(D(x_{\mu_i}))$	$\hat{r}$	7	$\hat{r}$	5

- For all c_{μ_j} with $j > k$:

$$a_{c_{\mu_j}} = 0, \tag{36}$$

$$b_{c_{\mu_j}} = \begin{cases} d(\hat{r}, x_{\mu_j}) - \frac{d(v, x_{\mu_j})}{2} & \text{if } d(v, x_{\mu_j}) < D(x_{\mu_j}), \\ 0 & \text{else.} \end{cases} \tag{37}$$

- For all c_{μ_j} with $j < k$, $j \neq 1$:

$$a_{c_{\mu_j}} = \begin{cases} d(\hat{r}, v) - \frac{d(V(D(x_{\mu_j})), v)}{2} & \text{if } d(\hat{r}, v) < d(\hat{r}, x_{\mu_j}) \text{ and } d(x_{\mu_j}, v) < D(x_{\mu_j}), \\ 0 & \text{else,} \end{cases} \tag{38}$$

$$b_{c_{\mu_j}} = \begin{cases} d(\hat{r}, x_{\mu_j}) - \frac{d(v, x_{\mu_j})}{2} & \text{if } d(\hat{r}, v) < d(\hat{r}, x_{\mu_j}) \text{ and } d(x_{\mu_j}, v) < D(x_{\mu_j}), \\ 0 & \text{else.} \end{cases} \tag{39}$$

- For c_{μ_k}:

$$a_{c_{\mu_k}} = d(\hat{r}, v) - \frac{d(V(D(x_{\mu_k})), v)}{2}, \tag{40}$$

$$b_{c_{\mu_k}} = \frac{d(\hat{r}, x_{\mu_k}) + d(\hat{r}, v)}{2}. \tag{41}$$

- For c_{μ_1}, if $k \neq 1$:

$$a_{c_{\mu_1}} = \begin{cases} \frac{d(\hat{r}, v)}{2} & \text{if } d(\hat{r}, v) < d(\hat{r}, x_{\mu_1}) > d(v, x_{\mu_1}), \\ 0 & \text{else,} \end{cases} \tag{42}$$

$$b_{c_{\mu_1}} = \begin{cases} d(\hat{r}, x_{\mu_1}) - \frac{d(v, x_{\mu_1})}{2} & \text{if } d(\hat{r}, v) < d(\hat{r}, x_{\mu_1}) > d(v, x_{\mu_1}), \\ 0 & \text{else.} \end{cases} \tag{43}$$

Observe that there might exist points of the collapsed tree network that are contained in multiple intervals $]a_{c_{\mu_j}}, b_{c_{\mu_j}}[$, $j \in \{1, ..., \hat{p} - 1\}$. However, by definition of equation (34) and $f_{\mu_1}(y_{\mu_1})$, only one of the corresponding chain functions $f_{\mu_j}(y_{\mu_j})$ may have a value larger than zero at each of these points, so that the related demand will only be considered once in the subsequent calculation of the actual follower's market share $W_F(v|X_{\hat{p}})$. In analogy to Section 4.1 we get

$$W_F(v|X_{\hat{p}}) = \sum_{j=1}^{\hat{p}-1} W_F^{c_{\mu_j}}(v|X_{\hat{p}}), \tag{44}$$

where

$$W_F^{c_{\mu_j}}(v|X_{\hat{p}}) = \begin{cases} f_{\mu_j}(b_{c_{\mu_j}}) - f_{\mu_j}(a_{c_{\mu_j}}) & \text{if } d(\hat{r}, v_{\mu_j}^s) \geq a_{c_{\mu_j}}, \\ f_{\mu_j}(b_{c_{\mu_j}}) - f_{\mu_j}(a_{c_{\mu_j}}) - \pi(a_{c_{\mu_j}}) & \text{else,} \end{cases} \tag{45}$$

corresponds to the portion of the total follower's market share that is induced by chain c_{μ_j} for any $j \in \{1, ..., \hat{p} - 1\}$.

Equations (36)-(45) can be evaluated in constant time for a given v, so that we are now able to determine a (special structure) discrete, binary $(1|X_{\hat{p}})$-medianoid with time complexity $O(\hat{n}\hat{p})$ as claimed in Lemma 4. The following algorithm stores $(1|X_{\hat{p}})$-medianoids in the set Y; the corresponding follower's market share equals W^*.

Algorithm 4.5 Evaluate.

```
 1: $Y := \emptyset$, $W := 0$, $W^* := 0$
 2: for all $v \in V \setminus \{X_{\hat{p}}\}$ do
 3:    $W = 0$
 4:    for $j = 1$ to $\hat{p} - 1$ do
 5:       $W = W + W_F^{c_{\mu_j}}(v|X_{\hat{p}})$ (Equations (36)-(45))
 6:    end for
 7:    if $W > W^*$ then
 8:       $Y = \{v\}$
 9:       $W^* = W$
10:    else if $W = W^*$ then
11:       $Y = Y \cup \{v\}$
12:    end if
13: end for
14: if $Y = \emptyset$ then
15:    $Y = \{\hat{r}\}$
16: end if
```

Let us consider our example with the follower's location in vertex $v = 6'$, i.e. $k = 1$. We get $a_{c_{\mu_1}} = 2 - 2/2 = 1$, $b_{c_{\mu_1}} = (3 + 2)/2 = 2.5$, $a_{c_{\mu_2}} = 0$, $b_{c_{\mu_2}} = 5 - 3/2 = 3.5$, $a_{c_{\mu_3}} = 0$, $b_{c_{\mu_3}} = 0$, $a_{c_{\mu_4}} = 0$, $b_{c_{\mu_4}} = 0$ and $W_F(6'|X_{\hat{p}}) = 15.5$. Analogously, one computes the follower's market share when locating in vertex 1, 2, 3, 8 or 9. By comparison of these values we find vertex $6'$ to be the $(1|X_5)$-medianoid.

We may now conclude this section with the main result in Theorem 6.

Theorem 6. *The problem of finding a discrete, binary $(1|p)$-centroid of a tree network $N = (V, E, \lambda)$ with vertex and edge demand can be solved with time complexity $O(n^2 p \log \xi(N))$.*

Proof. We apply bisection on the interval $[0, \xi(N)]$ to find the smallest value $\hat{\xi}$ such that a $\hat{\xi}$-bounding set with at most p elements exists. The termination criterion described in the proof of Theorem 5 can be adapted to the case at hand. □

5. Conclusion

In this paper we have analyzed a classical sequential location problem on networks, the $(r|p)$-centroid problem under a binary choice rule. Differing from the majority of previous publications, we have considered networks with both, vertex and edge demand. The corresponding follower problem is known to be NP-hard even on networks with edge demand only [14]. We have proven that this remains true for the $(r|p)$-centroid problem. On tree networks, however, an efficient algorithm to determine a $(1|p)$-centroid has been derived, after having restricted the location sites to the vertex set of the underlying network (discrete problem class). Bilevel programming models have been presented for the discrete $(r|p)$-centroid problem on general networks with vertex and edge demand. Computational results based on these models indicate, that heuristics will need to approximate the follower's response to a given set of leader's locations. An upper bound on the follower's optimal market share has previously been introduced for the case of vertex demand only. We have shown that this bound can be adapted to the case of vertex and edge demand. Future research topics (cf. also Section 3.4) include the application of non-uniform demand densities or different types of choice rules, as, for example, proportional or partially binary choice rules.

Appendix A. Computational results

Note: "-" marks trivial instances with $r + p \geq n$.

Table A.6: Computational results - Model BBNP with fixed set of leader's locations, $p = 1$.

		avg. comp. time	instances not solved to optimality	
r	# of vertices	[sec.]	#	average gap [%]
1	5	0.0188	0	0
	10	0.1062	0	0
	15	0.3963	0	0
	20	1.0048	0	0
	25	2.6973	0	0
2	5	0.0967	0	0
	10	0.4306	0	0
	15	1.5272	0	0
	20	3.9342	0	0
	25	11.8092	0	0
3	5	0.0361	0	0
	10	0.2915	0	0
	15	6.0265	0	0
	20	28.1928	0	0
	25	202.638	0	0
4	5	-	-	-
	10	0.4084	0	0
	15	35.2106	0	0
	20	297.308	0	0
	25	1424.61	4	29.05

Table A.7: Computational results - Model BBNP with fixed set of leader's locations, $p = 2$.

		avg. comp. time	instances not solved to optimality	
r	# of vertices	[sec.]	#	average gap [%]
1	5	0.0157	0	0
	10	0.0875	0	0
	15	0.2605	0	0
	20	0.6396	0	0
	25	2.0358	0	0
2	5	0.0188	0	0
	10	0.0828	0	0
	15	0.3199	0	0
	20	1.0142	0	0
	25	4.7645	0	0
3	5	-	-	-
	10	0.1014	0	0

	15	0.5302	0	0
	20	2.672	0	0
	25	19.9147	0	0
4	5	-	-	-
	10	0.1329	0	0
	15	1.2654	0	0
	20	10.329	0	0
	25	114.313	0	0

References

[1] Hotelling H. Stability in competition. The Economic Journal 1929;39(153):41–57.

[2] Drezner T. Competitive facility location in the plane. In: Drezner Z, editor. Facility Location - A Survey of Applications and Methods. New York: Springer; 1995, p. 285–300.

[3] Eiselt HA, Laporte G, Thisse JF. Competitive location models: A framework and bibliography. Transportation Science 1993;27(1):44–54.

[4] Eiselt HA, Laporte G. Sequential location problems. European Journal of Operational Research 1996;96(2):217–31.

[5] Kress D, Pesch E. Sequential competitive location on networks. European Journal of Operational Research 2012;217(3):483–99.

[6] Plastria F. Static competitive facility location: An overview of optimisation approaches. European Journal of Operational Research 2001;129(3):461–70.

[7] Serra D, ReVelle C. Competitive location in discrete space. In: Drezner Z, editor. Facility Location - A Survey of Applications and Methods. New York: Springer; 1995, p. 367–86.

[8] ReVelle CS, Eiselt HA. Location analysis: A synthesis and survey. European Journal of Operational Research 2005;165(1):1–19.

[9] Hooker JN, Garfinkel RS, Chen CK. Finite dominating sets for network location problems. Operations Research 1991;39(1):100–18.

[10] Hay DA. Sequential entry and entry-deterring strategies in spatial competition. Oxford Economic Papers 1976;28(2):240–57.

[11] Prescott EC, Visscher M. Sequential location among firms with foresight. The Bell Journal of Economics 1977;8(2):378–93.

[12] von Stackelberg H. Marktform und Gleichgewicht. Vienna: Springer; 1934.

[13] Hakimi SL. On locating new facilities in a competitive environment. European Journal of Operational Research 1983;12(1):29–35.

[14] Dasci A, Eiselt H, Laporte G. On the (r, X_p)-medianoid problem on a network with vertex and edge demands. Annals of Operations Research 2002;111(1):271–8.

[15] Okunuki KI, Okabe A. Solving the Huff-based competitive location model on a network with link-based demand. Annals of Operations Research 2002;111(1-4):239–52.

[16] Spoerhase J, Wirth HC. (r,p)-centroid problems on paths and trees. Theoretical Computer Science 2009;410(47-49):5128–37.

[17] Bandelt HJ. Networks with Condorcet solutions. European Journal of Operational Research 1985;20(3):314–26.

[18] Bauer A, Domschke W, Pesch E. Competitive location on a network. European Journal of Operational Research 1993;66(3):372–91.

[19] Swamy MNS, Thulasiraman K. Graphs, Networks, and Algorithms. New York: Wiley; 1981.

[20] Hakimi SL. Locations with spatial interactions: Competitive locations and games. In: Mirchandani PB, Francis RL, editors. Discrete Location Theory. New York: Wiley; 1990, p. 439–78.

[21] Hansen P, Labbé M. Algorithms for voting and competitive location on a network. Transportation Science 1988;22(4):278–88.

[22] Hansen P, Thisse JF. Outcomes of voting and planning: Condorcet, Weber and Rawls locations. Journal of Public Economics 1981;16(1):1–15.

[23] Dobson G, Karmarkar US. Competitive location on a network. Operations Research 1987;35(4):565–74.

[24] Dempe S. Foundations of Bilevel Programming. Kluwer Academic Publishers; 2002.

[25] Bard JF. Practical Bilevel Optimization - Algorithms and Applications. Dordrecht: Kluwer; 1998.

[26] Benati S, Laporte G. Tabu search algorithms for the $(r|X_p)$-medianoid and $(r|p)$-centroid problems. Location Science 1994;2(4):193–204.

[27] Nemhauser GL, Wolsey LA. Integer and Combinatorial Ooptimization. New York: Wiley; 1999.

[28] Jaramillo JH, Bhadury J, Batta R. On the use of genetic algorithms to solve location problems. Computers & Operations Research 2002;29(6):761–79.

[29] Arostegui Jr. MA, Kadipasaoglu SN, Khumawala BM. An empirical comparison of tabu search, simulated annealing, and genetic algorithms for facilities location problems. International Journal of Production Economics 2006;103(2):742–54.

[30] Garey MR, Johnson DS. Computers and Intractability - A Guide to the Theory of NP-Completeness. New York: Freeman; 1979.

[31] Megiddo N, Zemel E, Hakimi SL. The maximum coverage location problem. SIAM Journal on Algebraic and Discrete Methods 1983;4(2):253–61.

[32] Alstrup S, Gavoille C, Kaplan H, Rauhe T. Nearest common ancestors: A survey and a new algorithm for a distributed environment. Theory of Computing Systems 2004;37(3):441–56.

[33] Cormen TH, Leiserson CE, Rivest RL, Stein C. Introduction to Algorithms. Cambridge: The MIT press; 2nd ed.; 2001.