TOP
https://doi.org/10.1007/s11750-018-0476-5

INVITED PAPER

Extensive facility location problems on networks: an updated review

Justo Puerto[1] · Federica Ricca[2] · Andrea Scozzari[3]

Abstract Location problems with extensive facilities represent a challenging field of research. According to the specialized literature, a facility is called extensive if, for purposes of location, it is too large in relation to its environment to be considered a point. There are many examples of this type of structures that appear in real-world applications both in the continuous space (straight lines, circles, strips) and in networks (paths, cycles, trees). There exists a recent literature review on the location of dimensional facilities on continuous space (Díaz-Báñez et al. in TOP 154:22–44, 2004; Schöbel in Location of dimensional facilities in a continuous space, 2015) that does not cover similar problems on networks. The goal of this paper is to review the location of dimensional facilities in networks. We mainly concentrate on the location of paths and trees considering the most common objective functions in the location literature, namely median and center. However, we also consider some other alterna-

This invited paper is discussed in the comments available at https://doi.org/10.1007/s11750-018-0473-8, https://doi.org/10.1007/s11750-018-0474-7, and https://doi.org/10.1007/s11750-018-0475-6.

This research has been partially supported by Spanish Ministry of Economía and Competitividad/FEDER grant number MTM2016-74983-C02-01.

✉ Justo Puerto
puerto@us.es

Federica Ricca
federica.ricca@uniroma1.it

Andrea Scozzari
andrea.scozzari@unicusano.it

1 IMUS and Dept. Estadística e Investigación Operativa, Universidad de Sevilla, Seville, Spain

2 Università di Roma, La Sapienza, Rome, Italy

3 Università degli Studi Niccolò Cusano, Rome, Italy

Published online: 09 April 2018

tive criteria generalizing them, as the ordered median objective function, or related to equity, reliability, and robustness. We include the basic tools and techniques that are applicable to develop algorithms for this kind of problems. Moreover, we present the best known complexity results for each of the considered problems. Finally, some suggestions are also made for possible directions of future research.

Keywords Path location problems · Tree location problems · Ordered median · Equity measures · Minimum loss criteria

Mathematics Subject Classification 90B-10 · 90C-35 · 68R-10

1 Introduction

The location of extensive facilities on graphs dates back to the early 1980s when the problem of locating a service facility with an extended structure rather than isolated points was formulated for the first time in Hedetniemi et al. (1981) and in Slater (1982). Location problems on networks were extended to path- and tree-shaped facilities that optimize the common measures of centrality in graphs, like the eccentricity and the sum of the distances. A first comprehensive review of extensive facilities location in networks appeared in Mesa and Boffey (1996). Since then, a number of papers have been published in the literature, both introducing new as well as efficient procedures for locating paths and trees on graphs, and presenting new centrality measures. Considering also the two variants of these problems, i.e., locating continuous or discrete extensive structures, scientific production in this field developed wide and diverse, and, perhaps, too fast and not in an organized way. This fact may explain the difficulty felt by many researchers entering this area in finding the appropriate literature. Actually, there are different volumes and survey papers, each focusing on different specific features of these problems. For example, the very recent book edited by Laporte et al. (2015) gathers the essential knowledge on what is called Modern Location Science. Unfortunately, the problem of locating extensive facilities on networks optimizing the common measures of centrality was left out from this book's contents.

In spite of this, growing attention has been paid by the specialized literature to the location of path-shaped and tree-shaped facilities on graphs. Besides the theoretical interest for this kind of problems, the impact of these results on real-life applications was immediately evident and soon became a strong motivation to push toward new results. The location of pipelines, of express lanes in a highway, and the design of public transportation routes are typical situations requiring the location of path-shaped facilities. On the other hand, problems related to the location of tree-shaped facilities can be found in every context in which the problem is finding a backbone structure (the subtree) of a given network. Examples can be found in the design of optical fiber networks, high-speed communication networks, hub-and-spoke air carrier networks and highway networks. For additional applications, the reader is referred to Mesa and Boffey (1996).

In continuation of the work by Mesa and Boffey (1996), the aim of the present survey is to update the state of the art of the literature on extensive facility location, reviewing papers on this topic published since 1996. It configures as a comprehensive and structured review of this literature focusing on network location of paths and trees, both continuous and discrete, and illustrating the wide variety of objectives that were introduced for these problems, from the classical centrality measures to objective functions related to robust and reliable optimization, up to the recently ordered median objectives introduced in Nickel and Puerto (2005) and Puerto and Tamir (2005).

In this survey, we intentionally left out the problem of locating extensive facilities in a continuous space, since there already exists a very recent comprehensive review on this subject by Díaz-Báñez et al. (2004) and Schöbel (2015). Similarly, in our paper, we consider only the location of paths and trees on networks, excluding other possible extensive structures, such as cycles, that, in fact, were studied in the literature, but relying on techniques that are quite different from the ones analyzed here. Actually, since this problem can be formulated only on graphs with cycles, the solution techniques considered in the literature are similar to those applied for the Hamiltonian cycle problem on general networks. Obviously, techniques for trees do not apply in this case. However, the interested reader may refer to Labbé et al. (2005) and Laporte and Martín (2007). Finally, in this paper we do not consider obnoxious facility location problems. We believe that this is a very particular research topic, strictly related to the specific applications that motivated the theoretical study of these problems. This produced a wide variety of contributions in the literature on this subject that, in our opinion, would require another, separated, comprehensive review. Therefore, to maintain the focus of our paper, we decided to leave out this topic.

For each considered problem, the paper surveys the computational complexity results, and for problems with polynomial time complexity (typically those defined on trees) it recalls the idea of the solution algorithms provided by the different authors. Since all the studied problems fall into the class of difficult NP-hard problems when the underlying network is a general graph, the survey naturally dedicates more space to results for tree networks. In addition, the paper provides a guide through the most common techniques introduced in the literature to design polynomial algorithms for these problems. Actually, in optimization problems on tree networks, one can rely on a variety of techniques exploiting the structural properties of this special class of graphs. Such techniques are related to preprocessing procedures, tree decomposition, special data structures and structural properties for trees. They have been repeatedly used in the literature and have become basic available tools for developing efficient solution algorithms. The collection of these techniques can be viewed as a toolbox for researchers who intend to start studying this subject.

The paper is organized as follows. Section 2 provides notation and definitions that constitute the essential background of location problems. Section 3 is devoted to reviewing the general techniques applicable to problems on trees mentioned above. Sections 4 and 5, which are related to path and tree location problems, respectively, share the same structure. For both, we first survey the results presented in the literature about the two classical centrality measures (center and median criterion), and then those related to the center–median objective, which is a convex combination of the previous two. For paths, two additional sections are devoted to new centrality measures that have been introduced in the recent literature on facility location. One class of new objectives are equity measures that are designed to control dispersion of the clients around the service structure. Another interesting criterion for location is based on measuring some kind of loss. Different problems can be formulated in this framework, depending on the meaning of the considered loss. In path facility location, one of these criteria measures a loss in terms of client satisfaction, leading to problems that are known as reliable location. A second possible loss is given by the regret function that measures the worst case opportunity loss in an optimization framework under uncertain scenarios on the service demand, and that at times is called robust location. A separated section is concerned with a special objective function known as ordered median objective that encompasses some classical objective functions in location analysis in an unified framework. In the concluding section, we provide some final thoughts and suggest some lines of research that, in our opinion, are promising ways to obtain new results in the field of location theory.

2 Notation and definitions

Consider a finite, connected and undirected graph with no self-loops $G = (V, E)$. We denote by $V(G)$ and $E(G)$ the vertex and edge set, respectively, with $|V(G)| = n$ and $|E(G)| = m$. An edge $e \in E(G)$ is identified by a pair of vertices (u, v), with $v, u \in V(G)$. Suppose that a non-negative weight $w(v)$ is associated with each vertex $v \in V(G)$, and a non-negative real length $\ell(e)$ is assigned to each edge $e \in E(G)$. Each edge in E is assumed to be rectifiable. $A(G)$ is the continuum set of points on the edges of G. We refer to interior points on an edge by their distances (along the edge) from the two end vertices of the edge. For any pair of points $x, y \in A(G)$, we let $d(x, y)$ denote the length of a shortest path in $A(G)$ connecting x and y. We refer to $A(G)$ as the metric space induced by G and the edge lengths. Consider a closed subset F of $A(G)$; the distance $d(v, F)$ between a vertex v and F is defined as the smallest of the distances from v to the points in F, that is:

$$d(v, F) = \min\{d(v, u)|u \in F\}.$$

In facility location, a weight assigned to a vertex of G represents population or demand for a service provided by a facility to be located. Therefore, in the more general case, the problem considers a weighted distance from v to F that is defined as follows:

$$w(v)d(v, F) = w(v)\min\{d(v, u)|u \in F\}.$$

In this survey, we consider closed subsets F that are connected sets of points forming a path or a tree in G. In particular, paths or trees to be located on a given graph G are classified w.r.t. the following criteria: (i) discrete, that is, the tips of the facility are all vertices of G; (ii) continuous, at least one tip of the facility belongs to the interior of an edge. Given a connected set F of G, a classical objective function considered in the literature is the eccentricity $E(F)$ of F which is the maximum weighted distance between F and a vertex of G:

$$E(F) = \max_{v \in V(G)} w(v)d(v, F). \tag{1}$$

Another typical objective in location theory is the distsum (or median) $D(F)$ of F given by the sum of the weighted distances between F and the vertices of G, that is:

$$D(F) = \sum_{v \in V(G)} w(v)d(v, F). \tag{2}$$

In some applications, it is necessary to locate facilities subject to a budget constraint. Depending on the particular structure of F, this budget constraint can be related to the length of the facility or to its monetary cost. In any case, for discrete paths or trees F the total cost is defined as follows:

$$\text{Cost}(F) = \sum_{e \in E(F)} \ell(e). \tag{3}$$

The extension of the cost function to continuous structures is similar accounting for the length of partial edges whenever it is necessary.

In the following, we review some algorithmic techniques that are useful for handling the problem of locating an extensive facility F on a graph G. We present preprocessing and decomposition techniques, as well as, special data structures and properties that may help in the design of efficient solution algorithms.

3 General techniques

This section is devoted to illustrate some of the most useful techniques adopted in the literature for developing algorithms for extensive facility location of paths and trees. Dynamic programming (DP) is the most commonly used method employed for finding an optimal facility F^*. In particular, it has been extensively used for solving location problems on tree networks, and, in the last two decades, efforts have been made to provide efficient, or even linear time, DP algorithms on trees. All the procedures proposed in the literature exploit a preprocessing phase and special data structures to collect all the information needed to efficiently compute the optimal value of an objective function. In the following, we start by reviewing algorithmic techniques for finding F^* in a tree minimizing either the eccentricity, $E(\cdot)$, or the distsum, $D(\cdot)$.

3.1 Preprocessing phase in recursive procedures

Given a tree $T = (V, E)$, with $|V(T)| = n$, let $\ell(e) = \ell(v, u)$ be a positive weight representing the length of the edge $e = (v, u)$. Suppose also that a non-negative weight $w(v)$ is assigned to each vertex v. In facility location problems on trees, it is customary to provide procedures that are based on recursive formulas computed at each vertex of T. A preprocessing phase is typically performed to obtain some quantities necessary to compute the recursive formulas. This is done in $O(nM)$ time through a single visit of the tree rooted at some vertex, with M being the time complexity of the operations performed at each vertex. Let a vertex $r \in V(T)$ and root T at r are given, and denote by T_r the resulting rooted tree. For any vertex v, let T_v be the subtree of T_r rooted at vertex v, $S(v)$ the set of children of v in T_r, and $p(v)$ the parent of v in T_r. Clearly, a vertex v is a leaf of T_r if and only if $|S(v)| = 0$. In the sequel, in a rooted tree T_r an edge denoted by (v, u) means that vertex v is on the unique path connecting u to r, or, in other words, that v is closer to r than u.

First, let us consider the eccentricity criterion. For the sake of simplicity, we assume that the vertex weights are $w(v) = 1$, for all $v \in V(T)$. Given a vertex v, the goal is to evaluate the maximum distance from v to the vertices of T. Let $E_{V(T_v)}(v)$ be the eccentricity of a vertex v w.r.t. the vertices in the subtree T_v, and let $E_{V\setminus V(T_v)}(v)$ be the maximum distance from v to the vertices not in $V(T_v)$. In a bottom-up visit of T_r, proceeding level by level from the leaves to the root r, it is possible to recursively compute $E_{V(T_v)}(v)$, for all $v \in V(T_r)$ as follows (see, e.g., Becker et al. 2007):

$$E_{V(T_v)}(v) = \begin{cases} 0, & \text{if } v \text{ is a leaf of } T_r \\ \max\limits_{u \in S(v)} \left[E_{V(T_u)}(u) + \ell(v,u)\right], & \text{otherwise.} \end{cases} \tag{4}$$

For a vertex v, let $u_{\max}(v)$ be the child of v that gives $E_{V(T_v)}(v)$. In a bottom-up visit of the tree it is also necessary to record the second maximum distance from v to the vertices in $V(T_v)$. We denote such distance by $E^2_{V(T_v)}(v)$ provided that the child that gives the second maximum distance to the vertex v is different from $u_{\max}(v)$. Proceeding top-down from the root r toward its leaves, we distinguish:

(i) if $v = r$, then $E_{V\setminus V(T_v)}(v) = 0$;
(ii) if v is adjacent to the root r:

$$E_{V\setminus V(T_v)}(v) = \begin{cases} E_{V(T_r)}(r) + \ell(r,v), & \text{if } v \neq u_{\max}(r) \\ E^2_{V(T_r)}(r) + \ell(r,v), & \text{if } v = u_{\max}(r) \end{cases}. \tag{5}$$

(iii) if v is not adjacent to the root r:

$$E_{V\setminus V(T_v)}(v) = \begin{cases} \max\{E_{V\setminus V(T_{p(v)})}(p(v)) + \ell(p(v),v),\, E_{V(T_{p(v)})}(p(v)) + \ell(p(v),v)\}, \\ \text{if } v \neq u_{\max}(p(v)) \\ \max\{E_{V\setminus V(T_{p(v)})}(p(v)) + \ell(p(v),v),\, E^2_{V(T_{p(v)})}(p(v)) + \ell(p(v),v)\}, \\ \text{if } v = u_{\max}(p(v)) \end{cases}. \tag{6}$$

All the above formulas can be computed in $O(n)$ time providing the eccentricity of each vertex of the tree. Unfortunately, the same approach is not viable if we assume that arbitrary non-negative weights are assigned to the vertices of T. Tamir (2004) suggests efficient algorithms to calculate $d_v = \max\{w(u)d(u,v) | u \in V(T)\}$, for all $v \in V(T)$. The procedure is based on a centroid decomposition of T (the reader is referred to Sect. 3.2 for a detailed explanation of this technique). Assume w.l.o.g. that T is binary. Let c be a centroid of T, and let $V^1(T)$ and $V^2(T)$ be the two subsets of vertices of the connected components obtained from T when removing c.

Recursively, for each $v \in V^1(T)$ compute $d_v^1 = \max\{w(u)d(u, v)|u \in V^1(T)\}$ and for each $v \in V^2(T)$ compute $d_v^2 = \max\{w(u)d(u, v)|u \in V^2(T)\}$. Then, for each $v \in V^1(T)$, define $b_v^1 = \max\{w(u)d(u, v)|u \in V^2(T)\}$, and for each $v \in V^2(T)$, define $b_v^2 = \max\{w(u)d(u, v)|u \in V^1(T)\}$. Hence, for each $v \in V^1(T)$, $d_v = \max\{d_v^1, b_v^1\}$, and for each $v \in V^2(T)$, $d_v = \max\{d_v^2, b_v^2\}$. The difficult task is the computation of b_v^1 (and b_v^2) for all $v \in V^1(T)$ ($v \in V^2(T)$). In Tamir (2004), it is shown that using the machinery provided in Frederickson and Johnson (1983), $O(n \log n)$ time is required to compute all the quantities b_v^1, $v \in V^1(T)$ (b_v^2, $v \in V^2(T)$). Therefore, the total time spent for computing the eccentricity of each vertex $v \in V(T)$ is $O(n \log n)$. We notice that the $O(n \log n)$ algorithm presented in Tamir (2004) can be replaced by a simple $O(n)$ algorithm when $w(v) = 1$, for all $v \in V(T)$.

In Puerto and Ricca (2011) and Ye (2017), the following quantity is introduced. For each $(v, u) \in E$ denote by $E_{V(T_u)}(v)$ the maximum weighted distance from v to the vertices in T_u. These quantities are useful in solving the center and centdian regret path location problems. Applying the idea of Tamir (2004), $E_{V(T_u)}(v)$ can be computed in $O(n \log n)$ time for each edge $(v, u) \in E$.

When the objective function is the distsum, another preprocessing phase can be applied for computing the sum of the weighted distances of each vertex $v \in V(T)$ to all the other vertices of T. Again, consider T_r and denote by $D_{V(T_v)}(v)$ the sum of the distances of a vertex v to all the vertices $u \in V(T_v)$. This quantity can be recursively computed in a bottom-up approach as follows:

$$D_{V(T_v)}(v) = \begin{cases} 0, & \text{if } v \text{ is a leaf of } T_r \\ \sum\limits_{u \in S(v)} \left[D_{V(T_u)}(u) + W_{V(T_u)}(u)\ell(v, u)\right], & \text{otherwise,} \end{cases} \tag{7}$$

where $W_{V(T_v)}(v)$ is the sum of the weights of all the vertices in $u \in V(T_v)$ recursively computed as

$$W_{V(T_v)}(v) = \begin{cases} w(v), & \text{if } v \text{ is a leaf of } T_r \\ w(v) + \sum\limits_{u \in S(v)} W_{V(T_u)}(u), & \text{otherwise.} \end{cases} \tag{8}$$

To compute the sum of the distances, $D(v)$, for all the vertices v in T_r, we proceed from the root r to the leaves, and, given that $D(r) = D_{V(T_r)}(r)$, we have:

$$D(v) = D(p(v)) + \ell(p(v), v)[W - 2W_{V(T_v)}(v)], \tag{9}$$

where $W = \sum_{v \in V(T_r)} w(v)$. The computation of all formulas (7)–(9) can be done once in $O(n)$ time. In the problem of locating extensive facilities, there is an additional quantity that must be computed when adopting the distsum as a centrality measure. It is the distance saving of a discrete path P, which was defined in Morgan and Slater (1980) as follows:

given a path P_{vu} and a path P_{uz} with edges disjoint from P_{vu}, the distance saving of P_{uz} with respect to P_{vu}, is the reduction in the sum of the distances obtained by adding P_{uz} to P_{vu}, i.e.,:

$$\mathrm{sav}(P_{vu}, P_{uz}) = D(P_{vz}) - D(P_{vu}).$$

For a given vertex v in $V(T_r)$, one can compute the distance saving of a path starting from v and ending at a leaf $z \in V(T_v)$ in linear time, $\mathrm{sav}(v, P_{vz})$, by recursively computing the following formula:

$$\mathrm{sav}(v, P_{vz}) = \mathrm{sav}(v, P_{vp(z)}) + \ell(p(z), z)W_{V(T_z)}(z). \tag{10}$$

Another useful tool for computing the objective function value of two (disjoint) facilities is the notion of bisector of two vertices. Given u and v, the bisector of u and v is an edge that contains the midpoint of the path from u to v. For any pair u and v, the bisector allows to partition a tree T into two subtrees: one contains vertices closer to u and the other those closer to v. This tree bipartition strategy, together with a preprocessing phase, has been used in point facility location (Nickel and Puerto 1999; Wang et al. 2001), as well as in the location of extensive facilities (see, e.g., Puerto and Ricca 2011; Wang 2002; Ye 2017).

3.2 Centroid decomposition

Centroid decomposition (CD) is a widely used technique for solving combinatorial optimization problems on trees. Let T be the given tree with $|V(T)| = n$ vertices. A centroid of T is a vertex characterized by the property that each of the connected components obtained by its removal contains at most $\frac{n}{2}$ vertices. A centroid of a tree T can be found in $O(n)$ time since it corresponds to a 1-median of the unweighted tree T (Slater 1982). In extensive facility location, centroid decomposition is used in particular when there is a constraint on the length of the facility F. The key idea is based on the fact that, given a tree T and a vertex $v \in V(T)$, only one of the two cases may occur:

1. the optimal facility F^* contains v;
2. the optimal facility F^* is fully contained in one of the subtrees obtained by removing v from T.

Therefore, a general implementation of the centroid decomposition technique for finding a path with length at most L that optimizes a given objective function is provided by the following recursive procedure.

```
BestPath
1. Find a centroid $c$ of $T$.
2. Let $T_c^1, \dots, T_c^k$ be the $k$ subtrees obtained from $T$ by removing $c$.
3. Find (if it exists) a facility $\bar{F}$ in $T$ with length at most $L$ passing through $c$ that optimizes a given
   objective function.
4. For each subtree $T_c^1, \dots, T_c^k$ do
       apply BestPath.
```

The time complexity for finding F^* in T depends on both the depth of the recursion and the time spent in step 3. For a tree with $|V(T)| = n$ vertices, the depth of the recursion is $O(\log n)$, so that the total effort needed to find F^* is:

$$\text{Time}(|V(T)|) = M(n) + [\text{Time}(|V(T_c^1)|) + \cdots + \text{Time}(|V(T_c^k)|)] \leq M(n) \cdot O(\log n),$$

where $|V(T_c^1)| + \cdots + |V(T_c^k)| = O(n)$, $|V(T_c^i)| \leq \frac{n}{2}$, $i = 1, \ldots, k$, and $M(n)$ is the time for computing a facility in T with length at most L passing through c that optimizes a given objective function.

3.3 Special data structures and properties

3.3.1 Top trees

Top trees have been introduced by Alstrup et al. (2001, 2005) as an efficient data structure for designing divide-and-conquer algorithms for problems on trees. To define a top tree τ of T, we need to introduce the following notions. For any subtree X of T, call a boundary vertex a vertex in X having a neighbor in T outside X. A cluster of T is a subtree having at most two boundary vertices. For any cluster Cl, $\beta(Cl)$ denotes the set of boundary vertices and $\pi(Cl)$ the path connecting the two boundary vertices, if they exist. If, on the contrary, Cl has only one boundary vertex, $\pi(Cl)$ simply denotes such vertex. Two clusters A and B can be merged if they intersect in a single vertex and $A \cup B$ is still a cluster. There are five different possibilities of merging [see also Wang et al. (2008) for the algorithmic details].

A top tree is a binary search tree with the following properties:

1. each vertex of τ represents a cluster of T;
2. the leaves of τ represent the edges of T;
3. each internal vertex of τ represents a cluster Cl merged from the two clusters represented by its children;
4. the root of τ represents T;
5. the height of τ is $O(\log n)$.

A top tree τ identifies a way to recursively decompose the original tree into subtrees, where the leaves of τ represent single edges of T. It was shown in Alstrup et al. (2005) that a top tree τ of T can be constructed in $O(n)$ time. When using top trees, some information must be also computed such as the minimum cost of a path located in a cluster Cl. This calculation is done bottom-up level by level in the top tree, so that, when computing the information for a cluster, the information for its children is already computed. For instance, for any subtree (cluster) Cl of T, one computes $\text{Cost}(Cl) = \min\{D(H)\}$, where H is a path of length ℓ in Cl, which represents the distsum of a best path of length ℓ contained in Cl. Then, in a bottom-up approach, it is possible to compute $\text{Cost}(Cl)$ for all clusters Cl by using information from the bottom level. In particular, in path location problems, the computation at the root of τ produces the distsum of a median path. As we will see in the sequel, this data structure can be exploited to derive efficient solution algorithms.

It is worth mentioning that the way of representing a tree T by a top tree τ is similar to the tree representation of a series-parallel graph. Actually, it is well known that a series-parallel multigraph G can be represented recursively starting from its edges. This representation identifies a binary decomposition tree that can be exploited to show that various optimization problems that are NP-hard on general graphs are, indeed, polynomially solvable in series-parallel graphs (see, e.g., Hassin and Tamir 1986; Takamizawa et al. 1982). Since a tree T is a special case of a series-parallel graph, a top tree τ is, in fact, a binary decomposition tree of T that allows to implement a DP algorithm for solving extensive facility location problems efficiently.

3.3.2 Binary trees

An alternative representation of a tree is by means of a transformation of the original tree T into a binary tree where each vertex that is not a leaf has exactly two children. Here, we report the transformation procedure provided by Tamir (1996). Consider a rooted tree T_r and, for a given vertex $v \in V(T)$, let $v_1, \dots, v_m$ be the set of its children.

1. First, consider each non-leaf vertex v of the original tree which has exactly one child, say v_1. Introduce a new vertex, $u_v(2)$, and connect it to v with a new edge. Assign a length equal to zero to this edge. The vertex $u_v(2)$ will be the second child of v in the new tree. It will be also a leaf of the new binary tree.
2. Consider each vertex v of the original tree which has at least three children, say $v_1, \dots, v_m$, with $m \geq 3$. Add $m-2$ new vertices, $u_v(2), \dots, u_v(m-1)$ and the following set of new zero length edges: $(v, u_v(2))(u_v(2), u_v(3)), \dots, (u_v(m-2), u_v(m-1))$. For each $s = 2, \dots, m-1$, replace the edge (v, v_s) by the edge $(u_v(s), v_s)$ with the same length. Similarly, replace the edge (v, v_m) by the edge $(u_v(m-1), v_m)$.

After processing each vertex $v \in V(T)$ of the original tree, the new tree is binary and has at most $2n-3$ vertices. Solving the problem on the original tree is equivalent to solving it on the new binary tree. From a computational point of view, for many facility location problems this transformation also allows a more efficient implementation of a divide-and-conquer strategy.

3.3.3 The spine decomposition of a tree

Another important tool for preprocessing the input tree is the so-called spine decomposition (SD), introduced by Benkoczi et al. (2009). This decomposition can be used to guide the computation of several relevant functions associated with subtrees of the given tree. This is a decomposition that partitions an input tree T into recursive connected components such that the depth of the recursion is $O(\log n)$ where n is the number of vertices in the tree.

We recall that CD can be represented by a rooted binary tree τ where each vertex of τ corresponds to a subtree of the original tree T. The root of τ, denoted by r_{CD}, represents the entire input tree T. In this tree representation, the depth of the recursion is equal to the height of τ. Thus, from the definition of the centroid, it immediately follows that the height of the decomposition is $O(\log n)$. Unfortunately, in this decomposition, a drawback is that the centroids of the subtrees at different levels of τ are unrelated. This problem is overcome by SD.

SD is applied on a rooted binary input tree T. Let r_{SD} be the root of the binary search tree τ representing the SD of T. SD consists of a collection of spines subtrees defined as follows (see also Bhattacharyya and Dehne (2008)). First, the spine $\pi(r_{\mathrm{SD}}, q)$ is chosen where $\pi(r_{\mathrm{SD}}, q)$ is a path $\{v_0 = r_{\mathrm{SD}}, v_1, \ldots, v_k = q\}$ in the original tree T such that q is a leaf. For all $i = 0, \ldots, k-1$ the vertex v_{i+1} is the child of vertex $v_i \in \pi(r_{\mathrm{SD}}, q)$ with the largest number of leaves in T. Next, recursively compute the spine decomposition for every subtree T_{u_i} rooted at a vertex u_i adjacent to $v_i \in \pi(r_{\mathrm{SD}}, q)$ that is not in the spine path $\pi(r_{\mathrm{SD}}, q)$. To make the recursion efficient, at each level of the decomposition two children, say s_1 and s_2 of a vertex of τ, must be balanced, that is, the difference between the subsets of leaves of the subtrees of T represented by s_1 and s_2 is minimal.

The main properties of SD are summarized below (see Benkoczi et al. 2009 for further details):

1. the height of the SD is $O(\log n)$, where the constant from notation O is at most 4.
2. the storage space complexity of the SD is $O(n)$.

3.3.4 Nestedness and other structural properties

In previous subsections, we assumed that the root of a tree T can be any vertex $v \in V(T)$. However, depending on the objective function considered, it may be more convenient to choose a specific vertex v as the root of T. In particular, for path center location problems with no constraints on the total length, it is well known that an optimal path center P^* of T is unique and contains the absolute center of T (Hedetniemi et al. 1981; Slater 1982) (nestedness property). The absolute center is defined as the point c in $A(T)$ that minimizes the eccentricity. Notice that c may be a vertex or a point in the interior of an edge of T. In this second case, one may always augment the set of vertices of the tree from $V(T)$ to $\{c\} \cup V(T)$. By exploiting this nestedness property, choosing c as the root of T allows to compute P^* in linear time (Hedetniemi et al. 1981). This property may be exploited also when the aim is to locate a subtree of T of limited total cost that minimizes the eccentricity.

Unfortunately, a similar property does not hold for the median objective function, so that rooting at a median vertex of T does not guarantee that the optimal median path contains it. In general, nestedness properties are rather difficult to prove. In some cases, examples contradicting the property can be provided showing that it does not hold [see, e.g., Puerto et al. (2009)]. Positive results for the nestedness property are known for location of trees with specific objective functions. It is known that center–median subtrees contain an optimal center–median point [see Tamir et al. (2002)] and that subtrees minimizing the sum of the k largest distances (i.e., k-centrum subtrees) also contain the corresponding point solutions [see, e.g., Puerto and Tamir (2005) and Tamir et al. (2005)]. This clearly implies that the same result is valid for the median subtree problem. These kinds of results have been extended also to subtree location problems in a tree network with the ordered median objective (see Sect. 5.4), where the λ-weights take at most two different values (Tang et al. 2012). Actually, for some other objective functions that we will also consider in this survey, this is still an open problem. The reader is referred to Rozanov (2015) and Rozanov and Tamir (2018) for the most recent results on this topic. These references extend the notion of nestedness to the property that a solution of a problem with a shorter length constraint is part of a solution of the same problem with a longer length constraint. In this more general framework, it is proven that the subtree location problem on trees satisfies nestedness. The approaches in Rozanov (2015) and Rozanov and Tamir (2018) resort to an embedding of the tree location problem in $\mathbb{R}^{|V|-1}$ by identifying subtrees with the vector of their distances to the leaves of the original tree rooted at some vertex.

There are other useful structural properties that can be exploited for designing efficient solution algorithms in path-shaped facility problems. Consider a weighted tree T and the problem of finding a median path, then the following two properties hold: (i) an optimal path is always discrete; (ii) an optimal path always connects two leaves of T. Of course, this property does not hold when searching for a median path of limited length L, since, depending on the value of L, it makes sense to distinguish between continuous and discrete paths.

4 Path location problems

In this section, we review those papers in the literature concerning the problem of locating paths on networks (i.e., $F = P$). In the following sections, the presentation is organized by discussing the location on trees first, and then providing results related to more general networks.

Regardless of the objective function adopted, the problem of locating discrete paths on trees is always polynomially solvable, since in a tree there are only $\binom{n}{2} = O(n^2)$ possible choices for the end vertices of a path, and between each possible such pair there exists in T only one path. Therefore, a complete enumeration of all paths requires $O(n^3)$ time. It is also well known that on trees these problems are polynomially solvable also if the facility is a continuous path (paths whose ends may be vertices or points along an edge), provided that the corresponding absolute point facility location problem is polynomially solvable for that objective function (Mesa and Boffey 1996). For the above reasons, what has been done in the literature for these problems is to provide efficient solution algorithms which improve on this time complexity. The survey of papers related to the location of paths on trees will focus on the results that at the current moment provide the best time complexity algorithm for every analyzed problem.

As we will see, when the network is not a tree, the majority of the results are related to NP-completeness of the problems. However, in some cases, polynomial algorithms can be provided also for connected graphs slightly more general than trees, as, for example, outerplanar graphs and cactus graphs that are (poorly) connected graphs that admit cycles.

4.1 Median path location

Although the problem of finding a median path P in a tree T has been solved in linear time by Slater (1982), the same problem with the additional constraint on the length of

P remained open until 1996. Actually, Mesa and Boffey (1996) only report on some complexity results provided in Minieka (1985), but no procedures are mentioned in their survey. It was just in 1996 that Peng and Lo (1996) presented the first algorithm for solving the problem of finding a discrete median path (or a core) of T with length exactly equal to a given value L. They consider a tree T with a positive length associated with each edge $e \in E(T)$ and weights $w(v) = 1$, for all $v \in V(T)$. The algorithm is based on a divide-and-conquer strategy following the same approach of the centroid decomposition method described in Sect. 3.2. The steps of the procedure are the following: (i) root the tree at a median vertex r and direct its edges from the root toward its leaves; (ii) find a path of length exactly equal to L which minimizes the distsum $D(P)$ among all paths of length L containing the root r; (iii) recursively repeat steps (i) and (ii) for each subtree rooted at a child of root r until the sizes of these subtrees are all smaller than L. For an efficient implementation of the above procedure, the authors provide a preprocessing phase in which formula (10) is applied to compute the distsum of a path passing through the current r with length exactly L. We note that the correctness of this solution strategy is a proof that the nestedness property does not hold for this problem. However, exploiting the median (centroid) property of T they solve the problem in $O(n \log n)$ time.

In 2002 Becker et al. (2002), the authors studied a problem slightly different w.r.t. the one in Peng and Lo (1996), namely finding a median path of length at most L. They show that the optimal value of this problem can be less than or equal to the optimal value of the problem with the exact length constraint. By applying a preprocessing phase followed by a centroid decomposition, they provide an algorithm that solves the problem on a tree with $w(v) = 1$, $v \in V(T)$, [unweighted case as in Peng and Lo (1996)] in $O(nL)$ time. For weighted trees [non-negative $w(v)$, $v \in V(T)$], their procedure has time complexity $O(n \log^2 n)$. The authors also note that the time complexity of their second algorithm becomes $O(n \log n)$ when applied to the unweighted case. Hence, by combining their two algorithms for the unweighted case, they achieve an overall time complexity of $O(n \min\{L; \log n\})$ which is (almost) linear when $L = O(1)$.

The most efficient algorithm for finding a core of bounded length on a tree has been provided by Alstrup et al. (1997, 2001). The authors introduce the notion of top tree as a special data structure used for recursively representing a given tree T (see Sect. 3.3.1). Exploiting a top tree representation of the original weighted tree T, the authors are able to provide a recursive algorithm that finds a discrete median path of length at most L in $O(n \log n)$ time. In Alstrup et al. (2001), the authors study also the problem of finding a continuous median path of bounded length. In this case, they present an $O(n \log n\, \alpha(n))$ time algorithm, where $\alpha(n)$ is the inverse of the Ackermann's function.

In 2008, Wang et al. (2008), stated that the complexity $O(n \log n)$ is optimal for the discrete median path. Given a tree network T with non-negative weights on its vertices and non-negative lengths assigned to the edges, the authors consider the problem of finding the conditional location of a median path of limited length on trees. Here, the term conditional means that the location of the path must also take into account that there are already some existing facilities in T. In fact, using the top tree data structure, they first present an algorithm for the unconditional location of a median path that can be extended to the conditional case without worsening the time complexity. The authors provide an $O(n \log n)$ and $O(n \log n\alpha(n))$ time algorithms for the discrete and the continuous case, respectively. The valuable result is that they prove that both the continuous and discrete path location problems have a complexity lower bound of $\Omega(n \log n)$ time. This means that the complexity of $O(n \log n)$ for the discrete case is optimal, so that it cannot be further improved.

After these results, little potential remained for complexity improvements on trees; therefore, the interest in extensive facility location moved to the problem of locating median paths in more complex graph structures. From a computational complexity point of view, in Hakimi et al. (1993) it is proved that locating one discrete or continuous path minimizing the sum of the distances with or without a length constraint is NP-hard on planar graphs with maximum vertex degree 5. The reduction is from the Hamiltonian path problem. In 1990, Richey (1990) proved that the discrete version of the problem is NP-complete on outerplanar graphs, but in Lari et al. (2008) the authors deepened this result since they note that the graph used in Richey (1990) for the reduction, is, in fact, a cactus graph. Hence, a stronger result can be stated, that is, the problem of locating one discrete path minimizing the sum of the distances with length at most L is NP-hard on the very simple class of cactus graphs (also known as almost trees). In Lari et al. (2011), the authors further refine this result and prove that, if the constraint on the length L is removed, finding a path P with distsum exactly equal to a given constant $K > 0$ remains NP-complete even on cactus graphs. In Becker et al. (2007), the median path problem has been considered in a rectangular $M \times N$ undirected grid graph G with non-negative weights on the vertices and positive weights on the edges. Here, the authors study the problem of finding a median path such that either the length of the path or its cardinality (number of edges) is bounded from above. The authors provide both computational complexity results and polynomial time algorithms by using a dynamic programming strategy after applying a preprocessing phase. They study different problems also defining the vertical distance between the two vertices in the same column of the grid graph G. Let $D(P)$ be the sum of the distances computed via the shortest paths from each vertex of G to P, and $d(P)$ be the sum of the vertical distances. In their paper, the authors prove that the following three problems are NP-complete: (i) finding a path P in G of length at most L such that $D(P) \leq K$; (ii) finding a path P in G between two fixed end vertices with length at most L and $D(P) \leq K$; (iii) finding a path P between two fixed end vertices with length at most L and $d(P) \leq K$. On the other hand, the following two problems are polynomially solvable: (iv) given two vertices v_1 and v_2 located at the opposite corners of the grid, finding a path P joining these two vertices, which minimizes $d(P)$, can be solved in $O(n)$ time; (v) given the two vertices v_1 and v_2, and a number $L \geq (N-1) + (M-1)$, the problem of finding a path P with cardinality at most L, which joins v_1 and v_2, with the minimum number of horizontal edges and which minimizes $d(P)$ can be solved in $O(nL)$ time.

A noteworthy contribution on the discussed problems on special classes of graphs is provided in Balasubramanian et al. (2009), where the authors consider the median path problem in bipartite permutation graphs, threshold graphs and proper interval graphs. All these graphs have weights on the vertices and lengths $\ell(e) = 1$ for all edges $e \in E(G)$. More precisely, in this paper the authors study the problem of finding a median path of bounded length and a conditional median path of length exactly equal to L. They refer to the definition of ordered paths, and, since for every path there is an ordered path with the same length and the same set of vertices, they restrict the search space by considering only ordered paths. By a DP approach, they are able to compute efficiently the sum of the distances of any ordered path through the computation of the sum of the distances of any ordered path with a fixed end edge [for technical details, the interested reader is referred to Balasubramanian et al. (2009) and the references therein]. Finally, they provide an $O(L|E|)$ time algorithm for finding a core of G with length exactly equal to L. In the conditional case, they also present an $O(|V||E|)$ time algorithm for bipartite permutation graphs and a $O(L|E|)$ time algorithm for threshold and proper interval graphs.

In Lari et al. (2011), the authors consider a connected outerplanar graph G with non-negative weights for the vertices and lengths equal to 1 for all edges and study the median path location problem on G. The outerplanar graph G is represented by blocks and bridges that correspond to the vertices of a representation tree τ in which an edge between two vertices exists if the two corresponding blocks/bridges share a cut vertex of G. No constraint on the length of P is introduced. The representation tree of G is rooted at each block and at each bridge, and the following procedure is repeated for each rooted τ. The root-block and the leaf-blocks are numbered in order to make it possible to compute some necessary quantities in a suitable visit of the graph (preprocessing phase). The algorithm performs a breadth first search (BFS) on the vertices of the rooted representation tree from the root to the leaves; at each step of the visit, each vertex of τ, which is itself an outerplanar graph, is visited twice face by face in clock and counterclock order. This allows to compute the distsum of each possible path starting in a vertex in the root block and ending in a vertex in the current block. In this way, the authors are able to solve the median path problem on G in $O(nk)$ time. When the end vertices of the path are fixed in advance, the same algorithm solves the problem in $O(n)$ time.

In conjunction with the problem of locating one median path, some authors investigated the problem of locating $p \geq 2$ paths on networks (p-core problem) by minimizing the sum of the distances. This latter problem with $p = 2$ was first introduced in Becker and Perl (1985) where the authors consider both cases of disjoint discrete paths and intersecting discrete paths on trees. By exploiting interesting relations between the 1-core and the 2-core of T, they propose two efficient algorithms for the disjoint case with time complexity of $O(n^2)$, and of $O(n \times \text{diam}(T))$, respectively, where $\text{diam}(T)$ is the number of edges in the diameter of the tree. They also observe that, since the average diameter of a tree is bounded by twice the average height of a rooted tree, the complexity of the second algorithm becomes $O(n\sqrt{n})$ time on average.

The problem of finding a set of $p \geq 2$ mutually disjoint paths in T that minimizes the sum of the distances of all vertices in T from any of the p paths was then considered in Wang and Lin (2000) and Wang (2002). They use a preprocessing phase and DP for finding a 2-core of a tree by searching for two disjoint median paths in T with fixed end vertices. In this paper, which is rather technical, the authors exploit a nice result given in Becker and Perl (1985), according to which, given a 1-core C of T, there exists a 2-core $\{P, Q\}$ of T such that: (i) either one among P and Q is equal to C; (ii) or one terminal vertex of C is a terminal vertex of P and the other is a terminal vertex of Q.

Based upon this property, the idea is to find a 1-core C first, and then, by removing one edge of C at a time, to compute a 1-core with a fixed end vertex (corresponding to one of the two end vertices of C) in each of the two resulting subtrees, respectively. The authors improve the complexity of the procedure proposed in Becker and Perl (1985) by providing a linear time algorithm for the 2-core of T. We observe that the general p-core problem, with $p > 2$, was first studied by Hakimi et al. (1993). They show that when p is a constant, the problem is polynomially solvable on trees and provide a simple procedure with time complexity of $O(n^{2p+1})$. Similarly, in Wang and Lin (2000) and Wang (2002), by applying their 2-core algorithm as a subroutine, the authors also show that the general p-core problem can be solved in $O(n^{p-1})$ time for any constant $p \geq 2$. Novik (1996) studies the general p-core problem on trees and gives an $O(pn^2)$ time algorithm.

In the paper by Hakimi et al. (1993), there is also a deep analysis about the computational complexity of the problem of finding a set of $p \geq 2$ mutually disjoint paths on general graphs. In fact, they show that the problem of locating $p \geq 2$ continuous or discrete disjoint paths that minimize the sum of the distances (with or without a length constraint) is NP-hard on planar graphs with maximum vertex degree 5 and on bipartite graphs. They exploit a reduction from the Hamiltonian path problem, and, since the Hamiltonian path problem is NP-hard in grid graphs [see Itai (1982)], the above problem remains NP-hard also in this class of graphs. In Puerto et al. (2014), these results have been further refined. Actually, the authors prove that given a cactus graph with non-negative weights for the vertices and positive lengths for the edges, the problem of finding $p \geq 2$ disjoint discrete median paths such that the sum of the edge lengths are less than or equal to L is NP-complete. They also observe that, in the case of $p = 2$, the problem remains NP-complete if any constraint on the total length of the two paths is introduced, but it is required that the sum of the distances must be exactly equal to a given value.

The above results assume that p is a constant. When p is an input variable, the problem becomes much harder to solve: in this case, locating $p \geq 2$ disjoint discrete paths that minimize the sum of the distances with cost at most L is NP-hard even on trees [see Hakimi et al. (1993)]. Furthermore, to the best of our knowledge, the problem of finding $p \geq 2$ disjoint continuous median paths with cost at most L is still an open problem [see also Tami and Lowe (1992)].

We are not aware of further exact algorithms applied for solving the median path problem on classes of graphs more general than trees. It is clear that, due to the actual importance of this kind of problems in real life applications, the problem of locating median paths on general graphs, and, in particular, on planar graphs, was widely studied in the literature by developing heuristic procedures that perform well when an exact algorithm is not available. It is easy to imagine how extended is the literature on this line of research; in our opinion, including it in the present survey would have diverted too much our presentation from its main focus.

To conclude this section, we point out that some authors have criticized the pertinence of the median criterion in facility location, since this criterion alone is not able to capture all the essential elements of a location problem. Actually, optimizing the median objective function may produce solutions that are unacceptable for those clients who are the furthest located from the facilities. In fact, it must be taken into account that the median criterion alone tends to favor clients who are clustered in population centers, penalizing those who are spatially dispersed.

4.2 Center path location

In this section, we review papers devoted to the problem of locating central paths in a given graph G. In general, it is supposed that the center objective is easier to optimize than the median one. Actually, some problems that are difficult to solve with the distsum objective function become polynomially solvable when considering the center criterion on the same class of graphs.

In 1981, Hedetniemi et al. (1981) first defined the path center P^* of a general graph G as the shortest path that minimizes the maximum (weighted) distance from any vertex of G (weighted eccentricity). A greedy procedure is presented in Hedetniemi et al. (1981) for solving such problem on trees in linear time. This algorithm is based on the fact that a path center of T always contains the absolute center of T (nestedness property). Evidence that path center optimization problems are more tractable than median path location problems is given by the fact that in 2006, Bhattacharya et al. (2006) and Bhattacharya et al. (2009) proposed a linear time algorithm for finding a discrete or continuous path center with or without a length constraint in tree networks with positive weights for vertices and positive lengths for edges. In fact, they consider the problem in a conditional setting supposing that there are some already existing facilities located in the tree. However, they first develop the algorithm for the unconditional case and then explain how to adapt it to the conditional problem without affecting the overall time complexity. They present a parametric pruning procedure that generalizes the Megiddo's prune-and-search algorithm [see Megiddo (1983)]. Basically, the main idea is to visit the tree and prune the vertices that do not determine the optimal service cost of the path, and to shrink the facility if some subpath is known to be a part of an optimal facility. Note that the $O(n)$ time complexity of the algorithm in Bhattacharya et al. (2006) and Bhattacharya et al. (2009) means that the computational complexity of the path center problem cannot be further improved.

As in the median case, path center location problems, with or without a length constraint, become NP-hard both in the continuous and in the discrete case when $p \geq 2$ disjoint paths must be located. The problem remains NP-hard on planar graphs with maximum vertex degree 5, bipartite graphs, grid graphs and cactus graphs [see Hakimi et al. (1993), Hakimi et al. (1993) and Puerto et al. (2014)]. However, when p is an input variable, (Tami and Lowe 1992) provide an $O(n^3p^2)$ time algorithm for locating $p \geq 2$ disjoint discrete or continuous paths minimizing the eccentricity objective on trees.

As for the distsum objective, also the eccentricity alone is not able to capture all the essential elements of a location problem. For instance, minimizing the eccentricity may lead to a significant increase in the total distance traveled by the clients. Therefore, even in the case of point location, some authors [see, e.g., Halpern (1976) and Halpern (1978)] already started modeling the problem as a bicriteria one by considering a combination of the sum of the distances and the maximum distance. This led to the formulation of path location problems by multicriteria optimization models. In the next section, we first illustrate the center–median approach. Then, focusing on other centrality measures, additional multicriteria optimization models are presented.

4.3 Center–median path location

Averbakh (1999) introduced the problem of finding a path-shaped facility on trees by considering both the center and the median criterion. Non-negative weights are associated with the vertices and edges have positive lengths. Assuming strictly positive weights for the leaves of the tree, they study the following problems: (i) find a path which minimizes the sum of the distances and such that the maximum distance from the furthest vertex of the tree to the path is bounded by a fixed constant; (ii) find a path which minimizes the maximum distance under the restriction that the sum of the distances is bounded by a fixed constant; (iii) find the set of all the Pareto-optimal paths w.r.t. the two criteria. They provide a two-phase dynamic programming procedure which solves all the above three problems. It first finds a set M of cardinality at most n, which includes the set of Pareto-optimal paths along with some extra paths in $O(n)$ time. A path that belongs to M is represented by the pair of real values corresponding to its eccentricity and distsum. They compute the absolute center c of the tree and show that all possible representations of the paths in M are computed by relying on a construction of paths not containing and containing c. In a second phase, this procedure finds the optimal path for the first two constrained problems by searching among the paths in M. Since they show that the cardinality of M is of at most n, the overall two-phase procedure has time complexity $O(n)$ for the first two problems. On the other hand, extracting from M the set of all Pareto-optimal paths requires $O(n \log n)$ time.

In a following paper, Becker et al. (2007) considered problems (i) and (ii), but with an additional constraint on the maximum length of the paths. In particular, they provide $O(n \log^2 n)$ divide-and-conquer algorithms based on a centroid decomposition of the tree. The idea for both procedures is that, once a centroid of the tree is computed, an optimal path passing through this vertex can be found. If it is not the optimal path for the problem under consideration, then this optimal path must lie entirely in one of the subtrees rooted at the vertices adjacent to the centroid. The algorithm is recursively applied to these subtrees. As already discussed in Sect. 3.2, a centroid decomposition strategy ensures that the depth of the recursion is $O(\log n)$. All the procedures are based on a preprocessing phase that allows to efficiently compute all the necessary quantities used in the algorithms.

The problem of finding paths with two objective functions (i.e., center and median objectives) has been also investigated in a paper by Puerto et al. (2006). They consider a tree network where each vertex is associated with a pair of non-negative weights representing the center weight and the median weight. A positive length is assigned to each edge. The authors solve the following problems: (i) minimize a convex combination of the eccentricity and the distsum; (ii) minimize the distsum subject to an upper bound on the eccentricity; (iii) minimize the eccentricity subject to an upper bound on the distsum. They also show how to find the set of all non-dominated paths with respect to both criteria in $O(n \log n)$ time. The optimal paths must be of length at most L. To solve the problems, they find the weighted 1-center of the tree and root the tree at this vertex. Then, they apply DP to compute the representation set of all the non-dominated points. A particular data structure is used to efficiently obtain a representation of the bicriteria paths in the plane $(E(P), D(P))$ where $E(P)$ and $D(P)$ are two real values corresponding to the eccentricity and the distsum of a path P, respectively. The authors prove that the set of non-dominated pairs has cardinality at most $2n$ and the three above problems can be solved in $O(n \log n)$ time.

4.4 Path location with equity measures

As shown by the presentation in the previous sections, and by other survey papers published in the past, median and center problems, together with objective functions that are convex combinations of these two, are the most studied criteria in the literature on facility location. This is the reason why also location of extensive facilities started from the study of this type of objectives. However, it is well known that, even if combined together, these two traditional criteria are not able to measure some relevant aspects in path location. Therefore, some authors studied new optimization criteria related to the idea of minimizing the dispersion of the clients' demand with respect to the located facility. Dispersion can be formalized in different ways by applying different measures of the variability of the distribution of the distances from the clients to a facility. Minimizing dispersion of clients w.r.t. the facility corresponds to pursuing some form of equity criterion in the location of the facility. This criterion is particularly meaningful when locating public services, which, in principle, should be available to everybody at the same accessibility level. In the literature, this criterion has been widely studied for point facility location (updated results on this subject can be found in Mesa et al. (2003)) introducing as objective functions the variance of the distance traveled by a customer to a facility, as well as the range objective function which is given by the difference between the maximum and the minimum distance of a customer to a facility. Another interesting equity measure is the Hurwicz objective, which is given by a convex combination of the maximum and the minimum distance from the clients to a facility. Also, the study of extensive facility location on trees was developed in the literature optimizing these objective functions.

Puerto et al. (2009) analyze situations where the tree has positive lengths associated with the edges and no weights for the vertices, and the objective functions are the range and the Hurwictz criteria. The paper by Puerto et al. (2012) is a follow-up of the previous one where the tree has also non-negative weights associated with its vertices and the focus is on the range objective function. In Caceres et al. (2004), the authors consider the variance objective function and a tree T with non-negative vertex weights and positive edge lengths. They search for the continuous path in T that minimizes the variance of the weighted distance from each client to a facility. The previous problem was generalized and further investigated in Puerto et al. (2009) where the analysis was extended to the location of continuous and discrete paths, also considering the additional constraint on the maximum length of the path.

Puerto et al. (2009) studied six different problems related to the location of a path-shaped facility of unrestricted length on a tree T (which is not a path), in which all the vertices had the same weight, while positive real lengths were associated with the edges. They consider the range objective function $R(P)$ and the Hurcwitz objective function $H(P)$ which are defined as follows:

$$R(P) = \max_{u\in V(T)\setminus P} d(u, P) - \min_{u\in V(T)\setminus P} d(u, P) = E(P) - \mu(P), \qquad (11)$$

$$H(P) = \alpha \max_{u\in V(T)\setminus P} d(u, P) + (1-\alpha) \min_{u\in V(T)\setminus P} d(u, P) = \alpha E(P) + (1-\alpha)\mu(P), \qquad (12)$$

where $E(P)$ is the eccentricity of a path P, $\mu(P)$ is the minimum positive distance from P to the vertices of G, and $0 \leq \alpha \leq 1$. Both $R(P)$ and $H(P)$ are variability measures such that $R(P) \geq 0$ and $H(P) \geq 0$.

The two main problems are unconstrained: (P1) locating a path which minimizes the range; (P4) locating a path which minimizes the Hurwicz objective function. Two additional range-type constrained optimization problems are studied, namely, (P2) locating a path which minimizes the maximum distance subject to the minimum distance bounded below by a constant, and (P3) locating a path which maximizes the minimum distance subject to the maximum distance bounded above by a constant. Also, two constrained Hurwicz-type problems are studied: (P5) locating a path which minimizes the maximum distance subject to the minimum distance bounded above by a constant, and (P6) locating a path which minimizes the minimum distance subject to the maximum distance bounded above by a constant. All problems are formulated both in the continuous and discrete version. It is shown that P1–P6 are NP-hard on general graphs both for discrete and continuous path problems. Linear time algorithms are provided for all problems on tree networks, with the only exception of the problem of finding a continuous path in T that minimizes the range. For this single case, the proposed algorithm has a time complexity $O(n^2)$. Since all problems' objectives or constraints are defined over the two measures $E(\cdot)$ and $\mu(\cdot)$, the solution approach is to embed all six problems into suitable bicriteria path problems with respect to the functions $E(\cdot)$ and $\mu(\cdot)$. Two partial orders are defined on pairs $(E(P), \mu(P))$ to solve problems P1–P3 and P4–P6, respectively. Each path P is represented by the corresponding pair $(E(P), \mu(P))$ and the procedure generates a superset of paths containing all Pareto-optimal paths. Since the cardinality of this superset is $O(n)$, all Pareto-optimal paths can be obtained by enumeration. The solution algorithms are based on DP: first, the (point) absolute center c of the tree is computed and the tree is rooted at c. The superset of paths is computed by distinguishing paths not containing c, starting from c, or passing though c.

Focusing on the range objective function, in Puerto et al. (2012), the authors extend their analysis to the case of a tree network in which the vertices have non-negative weights. As before, both the continuous and the discrete path problems are studied and results are extended to the case in which there is an additional constraint on the maximum length of the path. For all problems already studied in Puerto et al. (2009), polynomial algorithms are provided with a time complexity that is increased by only one order of magnitude w.r.t. the corresponding one in the vertex unweighted case (namely, $O(n^2)$ replaces $O(n)$ for all problems except for continuous version of P1 for which the complexity rises from $O(n^2)$ to $O(n^3)$). When a constraint on the path length is added, the complexity for the discrete path problems increases in the same way. On the contrary, all continuous problems with a length constraint require an algorithm with time complexity $O(n^3)$. The authors apply different techniques for designing the algorithms for the various problems. For problem P1, the algorithms are based on DP. For P2 and P3, in the discrete case, both problems can be solved by using the same algorithm applied for P1, with the additional task of performing a check for feasibility on each evaluated path with respect to $\mu(P)$ in P2 and to $E(P)$ in P3. For the continuous versions of the two problems, it is shown that they can be reformulated as equivalent discrete problems defined on a new augmented tree with $O(n)$ additional vertices w.r.t. T. Some results guarantee that in any edge (k, h) of T, there are only two relevant points called semivertices, implying that one can always augment the set of vertices of T by adding to V new vertices located at such points. The optimal solution of the continuous version of problem P2 in T corresponds to the optimal solution of the discrete version of the same problem P2 in the augmented tree.

In Caceres et al. (2004), the path location on trees is investigated by adopting the variance objective function. The authors consider a tree T with positive weights associated with the vertices and positive lengths associated with the edges. A global equity measure for the location of a path-shaped facility on T is the variance of the weighted distances computed from each vertex to the facility. In fact, by minimizing this quantity the facility is located to make its distance to each client as close as possible to the average one. This means that all clients are treated in the same way. The authors study the location of continuous paths, so that they can follow a solution approach based on the explicit formulation of the problem as an optimization problem. Different cases are studied separately by distinguishing between paths with both tips in a given edge and paths with the two tips in different edges of the tree. For this latter case, the analysis is performed by fixing a pair of edges and finding the best path with the first tip in the first fixed edge and the other in the second fixed edge of the given pair. Different cases are generated in an organized way so that quantities necessary for the computations in each case can be updated efficiently. For each single problem, the best path is found explicitly by imposing first-order optimality conditions. In fact, for a fixed pair of edges of length l_j and l_k, respectively, the variance function is strictly convex (because the corresponding Hessian matrix is positive definite) on the compact set $[0; l_j] \times [0; l_k]$ and, therefore, the optimal solution corresponds to a stationary point.

The algorithm provided in Caceres et al. (2004) has $O(n^2 \log n)$ time complexity. This complexity has been improved by a factor of $O(\log n)$ in Puerto et al. (2009) where the same location problem is studied in a more general framework. Actually, in this more recent paper, the authors consider the same problem as in Caceres et al. (2004), but they study the optimal location either of a continuous or of a discrete path. The possibility of an additional constraint on the maximum length of the path is also investigated (bounded length problem). A DP technique is proposed for the bounded length continuous problem working in two stages which are repeated for every edge e of T: (1) first the algorithm finds the minimum variance feasible path with both tips in the given edge e; (2) for every other edge f of T, it finds the minimum variance feasible path with one tip in e and the other in f. At each step, the tips of the path are identified by solving a suitable mathematical program in which the length bound is imposed by an explicit constraint. For the bounded length discrete problem, the approach is similar: for each vertex r in T, the tree is rooted at r and the first evaluated path is the one corresponding to r itself; then the minimum variance feasible path with one tip in r and the other in T_r is found by using quantities computed via dynamic programming. The bound L on the maximum length of the path is taken into account by avoiding evaluating all paths in T_u when a path from r to u in T_u was already found with length already greater than L. The problems (both continuous and discrete) without a bound on the path length are solved by the same technique with a slight adaptation from the corresponding bounded problem: for a continuous path, the length constraint is not included in the mathematical program, while for the discrete path the algorithm simply skips to make the check on the length. The resulting computational complexities are therefore the same for the bounded or unbounded problems, and, in fact, they are equal to $O(n^2)$ for all the analyzed problems.

4.5 Path location with minimum loss objectives

The study of path location on networks had an extraordinary development in the second decade of the 2000s. Besides the classical measures, their possible combinations, and equity measures discussed in the previous section, attention was paid to problems of locating extensive facilities by minimizing different loss measures. Loss can be understood in different ways. In the papers that we review in this section, we distinguish two different types of loss. The first one is related to a loss in terms of client satisfaction that derives from a non-correct operation of the closest facility, and forces the client who needs to make use of the service to access a farther facility. The problem in this case typically involves the location of two facilities, each characterized by a probability of failure. The problem is generally referred to as reliable location and consists of locating facilities to minimize the global loss of all clients corresponding to a total expected service cost. The second type of loss is intrinsically present in the location problem when there is no complete parameter information. In this case, it may happen for example that vertex weights are not known, but only intervals of possible weight values are available for each client vertex. Therefore, the problem consists of finding the best location of the facility, according to an ordinary optimization criterion (such as for example the distsum), but relying only on partial information on the vertex weights. These mean that, in principle, all possible weight values must be considered for each vertex and a worst-case analysis on the objective function must be performed. These problems fall into the class of robust optimization problems in which a regret function is formulated w.r.t. the problem objective function and the problem configures as a minimax regret optimization problem. The regret is a measure of the loss that occurs when a facility is located, but, once the actual vertex weights become known, it does not correspond to the optimal one.

In Puerto and Ricca (2011), minimax regret path location problems are investigated. The problem is to find optimal paths, continuous or discrete, with respect to the center, the median and the center–median objective functions, when uncertain vertex weights are given in the form of closed intervals of real positive values. Non-negative real lengths are associated with each edge of the network. First of all, the authors note that these three problems are NP-hard on general graphs, since each of them contains as a special case the corresponding problem of finding a center, a median and a center–median path, respectively. For this reason, the problems are studied on tree networks providing polynomial time algorithms for each of them.

Let $\mathcal{P}$ be the set of all paths in a tree T. A scenario $W \in \Omega$ is a possible choice of a weight for each vertex in the corresponding given interval of values. Then, for a given $P \in \mathcal{P}$, and a given scenario $W \in \Omega$, the maximum regret of P w.r.t. a generic objective function, denoted by $F(W, P)$, is defined as follows:

$$R(P) = \max_{W\in\Omega} \max_{Q\in\mathcal{P}} \left[F(W, P) - F(W, Q)\right]. \tag{13}$$

The internal maximum $R(W, P) = \max_{Q\in\mathcal{P}} [F(W, P) - F(W, Q)]$ is the regret of path P under scenario W and measures the maximum deviation in terms of objective function between the evaluated path P and any other possible path Q in T which may be the optimal one once the actual vertex weights become known (i.e., under the unknown scenario that will come true). The maximum regret (13) is a measure of the worst-case deviation over all the possible scenarios $W \in \Omega$, and in the optimization problem it must be minimized over all possible paths $P \in \mathcal{P}$.

In Puerto and Ricca (2011), the authors consider three different objective functions $F(W, P)$ leading to three different minmax regret problems.

The Minimax regret path center problem searches for a path $P \in \mathcal{P}$ that minimizes:

$$R(P) = \max_{W\in\Omega} \max_{Q\in\mathcal{P}} \left[\max_{v\in V(T)} w(v)d(v, P) - \max_{u\in V(T)} w(u)d(u, Q)\right]. \tag{14}$$

The Minimax regret path median problem can be formulated as follows: find $P \in \mathcal{P}$ that minimizes

$$R(P) = \max_{W\in\Omega} \max_{Q\in\mathcal{P}} \left[\sum_{v\in V(T)} w(v)d(v, P) - \sum_{u\in V(T)} w(u)d(u, Q)\right]. \tag{15}$$

Finally, a convex combination of the two above objectives can be considered, provided that two different intervals of weights are associated with each vertex of the tree, one for the center and another one for the median part of the combination in the objective function. The resulting problem is called the double weighted Minimax regret path centdian problem.

For all the above problems, polynomial time algorithms are provided by Puerto and Ricca (2011), all based on a recursive scheme working only on a set of relevant scenarios for the vertex weights. The solution approach is based on some properties which allow to reduce the number of the evaluated paths and the number of scenarios to the point that the problem can be efficiently solved by enumeration on the restricted sets of relevant paths and relevant scenarios. For the three above problems, the time required by the suggested algorithms is $O(n^2)$, $O(n^4)$, and $O(n^5 \log n)$ for the minimax regret center, median and center–median problems, respectively.

In Ye and Wang (2015), the authors revisit the minimax regret problem for locating a median path on a tree in which vertex weights are uncertain and the only available information is that each vertex weight takes values in a given interval. They present an algorithm that improves on the time complexity of the procedure in Puerto and

Ricca (2011) from $O(n^4)$ to $O(n^2)$. The basic idea is that not all the relevant paths identified by Puerto and Ricca (2011) have to be evaluated, but only $O(n)$ of such paths are relevant. For each path P, the maximum regret $R(W, P)$ under scenario W is computed by separating the analysis for those paths Q that are disjoint from P and those that share a part of P (overlapping P). After a $O(n^2)$ time preprocessing, the authors are able to compute in $O(n)$ time the the worst-case regret of both types of paths. This leads to the final solution algorithm that improves by a factor $O(n^2)$ on the previous result in Puerto and Ricca (2011). Very recently, Ye (2017) has found a more efficient implementation for the minmax regret path center and path centdian problems in $O(n \log n)$ and $O(n^4)$ time, respectively.

The paper by Puerto et al. (2014) addresses reliable path-shaped facility location problems on networks. These problems are particularly difficult to solve, since the fact that the located facility may fail requires that more than one facility must be located and each client must be assigned to more than one facility.

Typically, the problem is to locate two facilities minimizing the expected service cost in the long run, assuming that probabilities of failure of the facilities are known in advance (2-unreliable path location problem). Let p_1 and p_2 be the two given probabilities of disruption of path-shaped facilities $\mathcal{P}^1$ and $\mathcal{P}^2$, respectively. For a given location of the two facilities that is denoted by $L = \{L(\mathcal{P}^1), L(\mathcal{P}^2)\}$, a client v is first associated with its closest facility, but, in case of disruption, he/she is re-directed to the other one, and, if this also fails, a fixed positive penalty β_v is applied to take into account the cost for the dissatisfaction of client v, since, in case no operating facilities are available to serve client v, he/she would be forced to access a facility outside the network. To formalize the objective function of the optimization problem, notation L_k^v, with $k = 1, 2$ is introduced to indicate the k-th closest facility to v, so that, if, for example, for a vertex v, $\mathcal{P}^2$ is the closest facility and $\mathcal{P}^1$ the second closest, one has $L_1^v = \mathcal{P}^2$ and $L_2^v = \mathcal{P}^1$. For each vertex v in V, the total expected weighted cost to serve v with L is given by:

$$Z_v[L] = w(v) \left[d(v, L_1^v)\ (1 - p_{L_1^v}) + d(v, L_2^v)\ p_{L_1^v}(1 - p_{L_2^v}) \right] + w(v) p_{L_1^v} p_{L_2^v}\ \beta_v \tag{16}$$

and the problem objective function is the sum of the above expected costs $Z[L] = \sum_{v \in V(G)} Z_v[L]$.

The above 2-unreliable path location problem is in fact a multi-facility path location problem that is difficult to solve even for two facilities and on very special classes of graphs. This result follows from the observation that when fixing one probability of disruption to 1 and the other to 0, the problem becomes the (classical) median path location problem which is NP-hard even on very simple classes of graphs. In view of this, Puerto et al. (2014) focus on the 2-unreliable path location problem on trees providing a polynomial algorithm that finds the optimal (discrete) path in $O(n^2)$ time. The analysis is performed by separating the evaluation of solutions given by pairs of vertex-disjoint paths and those given by pairs of paths which intersect. For the latter case, a dynamic programming approach is proposed, applying different techniques for the case in which the two paths intersect in exactly one vertex or in at least one edge.

The evaluation of solutions given by pairs of disjoint paths is based on the idea that if paths are vertex-disjoint, it is always possible to find in T an edge (i, j) such that, removing (i, j) from T produces two subtrees T_{ij} and T_{ji}, each containing only one of the two paths. A direct consequence of this is that for the given pair of paths, it is clear which is the closest facility to each client. This leads to the formulation of a restricted version of the problem w.r.t. edge (i, j) in which the two facilities can be located independently as median paths in the trees T_{ij} and T_{ji}. Provided that the vertex weights of each subtree are suitably adjusted at each step, the 2-unreliable path location problem with vertex-disjoint paths can be solved by repeatedly removing an edge of the tree T and locating the two median paths in the two subtrees.

Generalizing to a greater number of facilities, the authors point out that, under the additional constraint that the paths to be located must be vertex-disjoint, an analysis similar to the one followed in the paper can be adopted to solve the three unreliable vertex-disjoint path location problem in $O(n^3)$ time.

5 Tree location problems

A natural extension of the problem of locating paths minimizing a given centrality measure is locating trees or subtrees on a network. In this section, we illustrate the main results and the main techniques for locating tree-shaped facilities. As in Sect. 4, we first start by considering as underlying graph a tree T. It is obvious that, if no constraint is introduced on the tree to be located, an optimal subtree of a tree T is T itself. From a complexity viewpoint, the problem, with the additional constraint on the maximum length of the subtree, has been considered in Hakimi et al. (1993). In particular, in the discrete case, the problem of locating in T a subtree that minimizes the center objective function can be solved in polynomial time by a simple greedy algorithm; on the other hand, the problem becomes NP-hard when the distsum criterion is considered. Minieka (1985) showed that both problems are solvable in polynomial time in the continuous case.

5.1 Median tree location

Some authors have addressed the question of finding approximation algorithms for this problem. Here, we mention the paper by Tamir (1998) who provides fully polynomial time approximation schemes (FPTAS) which generate a $(1+\epsilon)$-approximation for the minimization problem, and a $(1-\epsilon)$-approximation for the maximization version of the problem, both in $O(\frac{n^2}{\epsilon})$ time. The author considers a tree network with non-negative weights on the vertices and positive lengths for edges. He studies the following problem: find a discrete subtree which minimizes/maximizes the sum of the distances with total cost bounded above/below by L. By means of a preprocessing phase and DP, the author presents FPTAS for both versions since he shows that they generalize the knapsack problem. Note that the reformulation as a knapsack problem of the problem of finding a subtree that minimizes the sum of the distances from the vertices leads also to the result that in the continuous case the median subtree problem with bounded length can be solved in linear time (Tamir 1998).

It is worth mentioning that, in the final remarks of the paper, the author suggests how the presented approach can be easily modified to provide FPTAS even for some nonlinear objective functions commonly used in location theory. More generally, he claims that his approach can be used to obtain FPTAS for any objective function $F(Y) = \sum_{j=1}^{n} f_j(Y)$, where $f_j(Y)$, $j = 1, \dots, n$, is a nondecreasing integer-valued function (not necessarily linear or stepwise linear) of the distance from a vertex v to a subtree Y of T. However, no details are given about this and, to the best of our knowledge, the problem is still open.

The location of a tree-shaped facility is also studied by Kim et al. (1996) for tree networks in both the continuous and the discrete version. The authors rely on a single objective function that controls both the length of the located subtree (setup cost) and the transport/service cost, i.e., the sum of the costs of the clients to access the facility. The authors investigate a problem in which these costs are expressed by a general function that, in the specific case, becomes the service cost of the median subtree location problem when it is computed as the weighted distance from the client to the facility. The authors provide a polynomial time algorithm to solve the general problem which requires $O(n^2)$ time to solve both the discrete and the continuous case.

The approach in Kim et al. (1996) applies DP to the tree T_r rooted at some vertex r. In a preprocessing phase, the algorithm computes quantities related to partial service costs associated with the vertices of T_r. This is the step that requires a major effort in the algorithm and leads to the overall complexity of $O(n^2)$. The algorithm then proceeds with a bottom-up computation of recursive formulas related to the best objective function value of a subtree T_i in T_r that includes also the edge connecting vertex i to its parent in T_r. This requires $O(n)$ time as usual. In spite of the above complexity, it is shown that in the special case when in the objective function the total service cost is computed as the sum of the weighted client–facility distances, the preprocessing can be done in $O(n)$ time, so that the overall time required to solve the discrete problem becomes linear. Other special cases that can be solved by this algorithm in linear time are illustrated, but these problems are more related to another problem, called the covering subtree problem, than to location (see, e.g., Boffey 1998).

For the continuous version of the subtree location problem, the complexity of the algorithm remains $O(n^2)$. This problem is solved relying on the results for the discrete case, since the authors prove that for the studied problem there always exists an optimal discrete subtree. The techniques adopted are augmenting T by $O(n^2)$ new vertices and applying centroid decomposition on a Bitree T^* (see Kim et al. 1996 for details).

The same problem studied in Kim et al. (1996) was investigated by George and Revelle (2003) who propose a mathematical programming approach. In this paper, the authors consider the location of a subtree on a given tree T in which two weights are associated with each edge (i, j), one corresponding to the edge length and the other measuring a construction cost that must be taken into account if edge (i, j) is included in the located subtree. Vertex weights are also defined and two different versions of the problems are studied: (i) the restricted case is the ordinary one, where distances between pairs of vertices in T are computed as the shortest paths on T; (ii) in the unrestricted case, such distances are computed on paths in an auxiliary complete graph G with the same set of vertices of T. As the authors themselves say, this variant is not explicitly recognized in the literature and, for this reason, in this survey we report only on the restricted version of the problem. Therefore, the problem on T is to locate a discrete subtree in a biobjective optimization framework that, on the one hand, requires to minimize the sum of the weighted distances from the vertices of T to the located subtree, and, on the other hand, requires the smallest possible construction cost for the subtree. The problem is formulated as an integer linear program with $O(n)$ constraints and two binary variables associated with each edge of T. The general solution idea is to systematically remove from T those edges that have a high construction cost and a small length. In this way, removed edges (those that will not be included in the subtree) will contribute at the same time to reduce the construction cost of the facility and the total transportation burden for demand vertices. To solve the program, a single objective function is obtained as the weighted sum of the construction and the service cost objectives. For the problem studied in Kim et al. (1996), which was proved to be polynomially solvable, the model performance is obviously excellent, providing a practical solution tool for this problem. Besides the results obtained for the unrestricted problem, based on a similar IP, the contribution of this paper to the literature of subtree location can be found also in the idea of applying the proposed IP models to generate an approximated trade-off curve of Pareto-optimal solutions w.r.t. the two objectives via the systematic generation of different pairs of weights for the two objectives combined in the model objective function.

The median subtree location problem has been further investigated by Tamir et al. (2005) in the context of conditional location. Conditional location problems have attracted several researchers, since they fit in many practical situations where there are some existing located facilities in the underlying network which already provide the service to the customers. Sometimes, to improve the service, an increased budget becomes available for locating additional facilities providing the same service. The authors consider discrete and continuous versions of the median problem and assume that the tree network has positive weights on vertices and a positive length associated with each edge. No constraint is placed on the shape of the already existing facilities that may be either paths or subtrees. Tamir et al. (2005) prove that the conditional continuous median subtree problem is NP-hard even for star graphs with all vertex weights being equal to 1. In the case of a conditional median subtree location problem (discrete or continuous), the authors provide an FPTAS.

As already pointed out, Minieka (1985) showed that in the continuous case, polynomial time algorithms for finding a median subtree of T do exist. However, Minieka was only interested in whether this problem could be solved in polynomial time or not, so that the procedure that he presented was not efficient. Wang (2000), propose efficient parallel algorithms on the EREW PRAM model[1] for finding in a tree network with arbitrary positive lengths associated with the edges, a subtree of specified length which minimizes the sum of the distances. The optimal tree may have partial edges. The author uses the Euler-tour technique and the tree contraction that are two well-known parallel techniques for computing tree functions. We do not enter into the details of such parallel methods since it is out of the scope of this survey. In any case, in the sequential case, the algorithm proposed in Wang (2000) can be regarded as an efficient implementation of Minieka $O(n^2)$ algorithm that allows to find a median

[1] This abbreviation stands for memory access models "Exclusive Read Exclusive Write Parallel Random Access Machine".

location of a tree-shaped facility of specified length in $O(\log n \log\log n)$ time using $O(n)$ work.

Some other authors focused their attention on other subtree location problems which are polynomially solvable on trees. Minieka (1985) was the first author who studied the minimum distsum subtree problem with the additional constraint that the number of vertices of the subtree is specified in advance. Later, Peng et al. (1993) introduced the notion of k-tree core of a tree, that is, a subtree minimizing the sum of the distances that has exactly k leaves. They provide two algorithms with complexity of $O(kn)$ and $O(n \log n)$ time, respectively. In 1997, Shioura and Uno (1997) presented another algorithm which is a modified version of the $O(kn)$ time algorithm by Peng et al. (1993) for unweighted tree networks. It first finds a core C of the given tree T in linear time, then finds $k-2$ paths needed to construct a k-tree core, and then adds them to C. Exploiting the result in Morgan and Slater (1980) that for any k-tree core S of T there exists a $(k+1)$-tree core S' such that $S \subset S'$, the same authors introduce the notion of local-rooted core. Let S be a subtree in T and v be a vertex not in S. A local-rooted core is a path P which maximizes the distance saving among all the paths starting from v, extending in T and not containing vertices in S. The key property is that they prove that a local-rooted core P is always a path starting from a vertex v adjacent to a vertex in S. Exploiting this property, they show that for any $(k-1)$-tree core S, if P is a local-rooted core for S which maximizes the distance saving among all the vertices adjacent to S, then, $S \cup P$ is a k-tree core of T. After a preprocessing phase, most of the effort of the algorithm is dedicated to finding the set of local-rooted cores of a median path C of T and then sort all the elements to find the $k-2$ largest paths in this set. For unweighted trees and by using Radix sort, they show that a k-tree core of T can be found in linear time.

When non-negative weights are assigned to each edge of T, their algorithm can find a k-tree core in $O(n)$ time, and k-tree cores for all k in $O(n \log n)$ time. The very interesting result is that they also show that $\Omega(n \log n)$ is the time for solving the latter problem and that therefore their procedure is optimal.

Becker et al. (2002) studied the (k, ℓ)-core problem, that is, the problem of locating a subtree S with a diameter at most equal to ℓ and with at most k leaves. Given a tree T, the diameter is the maximum distance between two vertices of T. They search for a subtree of T with at most k leaves since, given ℓ, they show that the optimal value of a subtree with exactly k leaves can be greater than the optimal value of a subtree with a number of leaves less than k. As in Shioura and Uno (1997), they consider two cases: (i) the underlying graph is an unweighted tree; (ii) a non-negative length and a non-negative weight are associated with each edge and vertex of T, respectively (weighted case).

In the first case, the authors present an algorithm which is a modified version of the procedure described in Shioura and Uno (1997). Actually, with constraint on the diameter of S, the property stating that "for a k-tree core S with diameter bounded by ℓ there exists a $(k+1)$-tree core S' with diameter at most ℓ such that $S \subset S'$" does not hold. This implies that it is no longer possible to find a core of length at most ℓ first, and then add $(k-2)$ local-rooted cores. However, exploiting the definition of a midpoint of a path, it is easy to classify all the subtrees of T having a diameter of at most ℓ by the midpoints v of their diameters. Then, given v, they root the tree at v and then prune the paths starting from v to obtain a new tree where all the vertices have a distance from v at most equal to $\frac{\ell}{2}$. In the new tree, they can apply the algorithm of Shioura and Uno (1997) for unweighted trees with an overall time complexity of $O(n^2)$.

In the weighted case, Becker et al. (2002) rely on CD according to which, given a centroid c, either a (k, ℓ)-core contains c or a (k, ℓ)-core is fully contained in one of the subtrees obtained by removing c from T. Hence, given a subtree rooted at the current centroid c, they first find all the paths P of length at most ℓ starting from c, then prune the tree as in the unweighted case, and finally, for each path P, they search a k-tree core of the resulting pruned tree that contains P. After a preprocessing phase applied to compute all the necessary quantities to calculate the distsum of a path in T and the set of local-rooted cores, the algorithm for finding a (k, ℓ)-core of a given weighted tree runs in $O(n^2 \log n)$ time.

In Wang et al. (2006), an $O(n^2)$ time procedure is presented for finding in a weighted tree T a (k, ℓ)-tree core having exactly k leaves. The algorithm is based on the following definition: a subtree of T is ℓ-maximal if it has diameter less than or equal to ℓ and any larger subtree containing it has diameter strictly greater that ℓ. Let M be the set of all ℓ-maximal subtrees of T. Any subtree of an ℓ-maximal subtree has diameter $\leq \ell$. Clearly, there is a subtree in M that contains a (k, ℓ)-tree core of T. Therefore, the (k, ℓ)-tree core problem can be solved as follows. First, construct M; second, within each subtree $S \in M$ find a k-leaf subtree S that minimizes the distsum. Finally, compute a (k, ℓ)-tree core of T by selecting, among the k-leaf subtrees obtained in the second step, the one with the minimum distsum. The authors prove that the number of ℓ-maximal subtrees of T is at most n. Additionally, they show that the set M can be constructed in $O(n^2)$ time. Hence, a (k, ℓ)-tree core of an edge weighted tree T can be found in $O(n^2)$ time. In the same paper, Wang et al. (2006) also present a new algorithm for the case with lengths of all edges equal to 1, which is comparable to that of Shioura and Uno (1997). This algorithm uses DP along with the computation of local-rooted cores of given rooted subtrees of T. They show that finding a (k, ℓ)-tree core of an edge unweighted tree T can be done in $O(\ell kn)$ time. Therefore, when the values of k and ℓ are small compared to n, this algorithm could be preferred to that in Becker et al. (2002).

5.2 Tree center location

Although the tree center location problem on tree networks is easier to solve than the corresponding median problem, for a long time this problem has been left aside in the literature. Actually, only Minieka (1985) and Hakimi et al. (1993) presented simple greedy algorithms for the continuous and the discrete tree center problems with bounded length, respectively, both of which run in $O(n^2)$ time by naive implementation. In 1997, Shioura and Shigeno (1997) showed that both the continuous and discrete versions of this problem in a tree network can be solved in $O(n)$ time. Exploiting the nestedness property for which the absolute center is always contained in an optimal center subtree (see Hakimi et al. 1993; Minieka 1985), the idea is that the continuous and discrete tree center problems of limited length can be modeled in linear time as the continuous bottleneck knapsack problem, respectively, and thus they can be solved in $O(n)$ time, as well. We observe that these results mean that it is not possible to further improve on the computational complexity of these problems.

Since the center objective function is more tractable than the median one, some authors have continued to analyze these problems in the conditional case. In Tamir et al. (2005), a study on the conditional location problem on trees is presented where the facility has to be a subtree of limited length. According to the center objective function, the authors first prove a nestedness property for the conditional case. Then, they present the main strategy based on a feasibility test. Given a real number r, they determine whether there exists a subtree rooted at a distinguished vertex of length not exceeding L, such that the weighted distance of each vertex from the subtree is at most r. For the problem of finding the conditional location of a center subtree, the authors provide an $O(n \log n)$ time algorithm.

The above time complexity has been further improved in Bhattacharya et al. (2006, 2009), where the authors present a linear time algorithm for finding a tree of bounded length, minimizing the center objective function on tree networks in the presence of existing facilities. We have already cited this paper in Sect. 4, where we reported a brief description of the method proposed in Bhattacharya et al. (2006, 2009) for finding a discrete or continuous path center with or without length constraint on trees with positive weights for vertices and positive lengths for edges. By exploiting the same strategy (based on pruning the vertices that do not belong to an optimal tree, and shrinking the facility if some subtree is known to be part of an optimal service center), the authors prove that the problem with length constraint can be solved in linear time on trees. Their procedure can be used also in the unconditional case without worsening the overall time complexity, so that the algorithm in Bhattacharya et al. (2006) and Bhattacharya et al. (2009) represents a valid alternative strategy compared to that in Shioura and Shigeno (1997) for the subtree center location problem. It is worth noticing that also in the conditional case the linear time complexity of the algorithm in Bhattacharya et al. (2006) and Bhattacharya et al. (2009) is optimal and cannot be further improved.

To conclude this section, we mention that in Wang (1998) the author considers the k-tree center problem, which is the problem of finding a subtree of a tree T that has exactly k leaves and minimizes the maximum distance. When $w(v) = 1$ for all $v \in V$, he provides an $O(n)$ time algorithm.

5.3 Center–median tree location

From a computational complexity point of view, center–median location problems inherit all the (theoretical) complexity results of the median tree location variants. For instance, the problem of locating a discrete center–median tree of limited length is NP-hard even on tree networks, since this problem contains the discrete median tree location problem as a special case. On the other hand, the center–median tree location problem is polynomially solvable in the continuous case.

In Tamir et al. (2002), the center–median problem of locating a subtree of bounded length on a tree T is studied introducing two versions of the problem: (i) in the first one, the subtree is restricted to contain a point fixed in advance; (ii) in the second, this restriction is removed, but the authors show that an optimal unrestricted center–median subtree always contains a center–median point, so that the latter problem can be solved by choosing the center–median point as the fixed point of the former problem and using the same algorithm. In the tree T, lengths are associated with the edges, while a pair of weights are associated with each vertex, the first referring to the center part of the objective function and the second used to compute the median part. They study the continuous version of problem (i) since the discrete problem contains as a special case the discrete median subtree location problem which is NP-hard. The problem is formulated as a convex piecewise linear programming problem that is solved in $O(n \log n)$ time through an algorithm that relies on a greedy procedure to evaluate the model objective function.

Furthermore, the center–median objective has been considered by Dvir and Segal (2008) for the problem of locating a (k, ℓ)-subtree of a tree network T. The authors call this structure a (k, ℓ)-coredian tree of T. They extend some results found for the k-tree core of a tree and prove that, for every $k \geq 3$, a k-coredian tree, that is, a subtree with exactly k leaves (without a length constraint), contains a coredian path (i.e., a center–median path without length constraint). Thus, a k-coredian tree can be found by first determining a coredian path P whose end points are leaves of T, and then adding to it $k - 2$ paths. These additional paths are those that maximize a distance saving function among all the paths that can be attached to P to obtain a k-coredian tree. The authors extend the classical definition of distance saving of a path P_{rv}, where r is an arbitrary vertex and v is a leaf of T. The distance saving of P_{rv} is defined as the difference between the center–median function of v and the center–median function of P_{rv}. Even if the details are not reported in the paper, these savings can be recursively computed in $O(n)$ time if the tree T is rooted at a specified vertex. In particular, Dvir and Segal root the tree at the (absolute) center c of T. Hence, following the strategy of Shioura and Uno (1997), finding a k-coredian tree takes $O(n)$ time. The introduction of the constraint ℓ on the diameter of the subtree to be located makes the problem more difficult to solve. In spite of this, the authors show that adopting the same strategy described in Wang et al. (2006), which is based on the definition of the set of all ℓ-maximal subtrees of T, a (k, ℓ)-coredian tree of T can be found in $O(n^2)$.

5.4 Ordered median tree location

The ordered median function unifies and generalizes the most common facility location criteria mentioned above, such as the median, the center and the center–median. If there are n demand points, the ordered median function is characterized by a vector of modeling weights, $\Lambda = (\lambda_1, \ldots, \lambda_n)$, satisfying $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \geq 0$. For a given subtree S, let $X(S) = \{x_1, \ldots, x_n\}$ be the set of weighted distances of the n demand points to S. The value of the ordered median objective at S is obtained as follows: sort the n elements in $X(S)$ in non-increasing order, and then compute the scalar product of the sorted list with the sequence Λ. It is easy to see that when $\lambda_i = 1$, $i = 1, \ldots, n$, we get the median objective. When $\lambda_1 = 1$, and $\lambda_i = 0$, $i = 2, \ldots, n$, we obtain the center objective, and when $\lambda_i = \alpha$, $i = 1, \ldots, n-1$ and $\lambda_n = 1$, we obtain the problem known as the α-center–median. Another important

Table 1 Median path location problems

Paper	Underlying graph	Characteristics	Comments
Alstrup et al. (2001)	Tree	Continuous length constr.	$O(n \log n\alpha(n))$ algorithm
Alstrup et al. (2001), Wang et al. (2008)	Tree	Discrete length constr.	$O(n \log n)$ algorithm Optimal $\Omega(n \log n)$ in the continuous case Wang et al. (2008)
Richey (1990), Lari et al. (2008)	Cactus graphs	Discrete/continuous length constr.	NP-hard
Becker et al. (2007)	Grid graphs	Discrete length constr.	NP-hard
Becker et al. (2007)	Grid graphs	Fixed end vertices vertical distance cardinality constr.	$O(n)$ algorithm $O(nL)$ algorithm
Balasubramanian et al. (2009)	Bipartite permutation threshold graphs interval graphs	Discrete length constr. equal edge weights	$O(mL)$ algorithm
Balasubramanian et al. (2009)	Bipartite permutation	Discrete length constr. conditional case equal edge weights	$O(nm)$ algorithm
Balasubramanian et al. (2009)	Threshold graphs interval graphs	Discrete length constr. conditional case equal edge weights	$O(mL)$ algorithm
Lari et al. (2011)	Outerplanar graphs	Discrete	$O(kn)$ algorithm
Wang and Lin (2000), Wang (2002)	Tree	Discrete $p = 2$ disjoint paths	$O(n)$ algorithm
Novik (1996)	Tree	Discrete $p > 2$ Disjoint paths	$O(pn^2)$ algorithm
Puerto et al. (2014)	Cactus graphs	Discrete length constr. $p = 2$ disjoint paths	NP-hard
Hakimi et al. (1993)	Tree	Discrete length constr. $p \geq 2$ disjoint paths p input variable	NP-hard
Tami and Lowe (1992)	Tree	Continuous length constr. $p \geq 2$ disjoint paths p input variable	Open problem

special case is the k-centrum objective, characterized by $\lambda_i = 1, i = 1, \ldots, k$, and $\lambda_i = 0, i = k + 1, \ldots, n$ (see, e.g., Kalcsics et al. 2003; Nickel and Puerto 2005).

Puerto and Tamir Puerto and Tamir (2005) considered the tactical and strategic location of tree-shaped facilities, using the ordered median objective. The models where there is a constraint on the total length of the extensive facility are called tactical, whereas those where the length of the facility is not a constraint but a decision variable, which enters also into the objective, are called strategic. For the strategic model, in Puerto and Tamir (2005) the authors study both the discrete and continuous versions in the special case in which the subtree must contain a specified vertex and prove that the convex ordered median function is submodular with respect to the lattice defined over the join and meet operations. This property leads to strongly polynomial algorithms for the discrete model, based upon minimizing a submodular function over that lattice. For the continuous problem, the authors present a compact LP formulation. Specifically, when the tree network has n vertices, the LP formulation uses $O(n^2)$ variables and $O(n^2)$ constraints. For the tactical model, the discrete version is NP-hard even for the special case of the median objective. For this reason, the paper only considers the continuous problem, when the subtree contains a specified vertex, in terms of a compact linear problem with $O(n^2)$ variables and $O(n^2)$ constraints. The validity of this formulation implies that the objective value of the parametric version of the model, where the length of the subtree is the parameter, is a decreasing piecewise linear convex function of the length of the subtree. The LP formulation implies that the time to solve the unweighted model, i.e., when the distances of all demand points are equally weighted, is strongly polynomial. For the particular case of the k-centrum objective, the paper presents stronger results based on a dynamic programming algorithm that can solve both the discrete and continuous strategic models of this problem. The complexity of the algorithms for the k-centrum objective function is improved based on a nestedness property of the subtree solution with respect to the point solution. Some further polynomial algorithms are presented for other ordered median functions where the vector of λ is non-increasing monotone and there are at most two different values in the vector Λ (Tang et al. 2012). In addition, Puerto et al. Puerto et al. (2017) provide improved complexity results for the location of subtrees under the k-centrum objective function. For the continuous versions of the strategic and tactical subtree problems, algorithms with a complexity of $O(n \log n)$ are also reported in Puerto et al. (2017).

6 Conclusions and final remarks

Due to an increasing interest in the location of extensive facilities during the last two decades, we deemed it necessary to collect the most recent results on this topic in a new survey paper. In continuation with the work by Mesa and Boffey (1996), the aim of the present paper is to produce a comprehensive and organized review of the literature on extensive facility location, providing a guide for researchers interested in the field and encouraging cross fertilization of ideas for the development of new research directions.

Concluding our discussion, in the following we provide some tables which summarize the most relevant results on location of paths and trees on networks. For a given problem, we indicate only the papers giving the best results presented in the literature so far. We adopt the same format already used in Mesa and Boffey (1996), providing a concise description of the characteristics of the problems under review. In the column labeled Characteristics, we specify if the facility to be located is discrete or continuous. When necessary, we also add details defining the precise variant of the problem (if there is a constraint on the length of the facility, or if the location is in the conditional framework, or if $p \geq 2$ facilities must be located, etc.). In the column labeled Comments, we report on the best polynomial computational complexity, if any; otherwise, we state if the problem is NP-hard or still open. The first column of each table refers to the problem's underlying graph.

Path location problems have received the greatest attention in the literature on extensive facility location of the last 20 years. On the basis of the results shown in Tables 1, 2 and 3 for the classical objective functions (median, center and center–median), there seems to be little potential for complexity improvements in the location on tree networks. In our opinion, even if these problems are NP-hard on very special classes of graphs, it would be interesting to investigate them on other specific graphs classes. There are some results of this type in Balasubramanian et al. (2009), but in this case the median path problem is studied only in the very special case when $\ell(e) = 1$ for all $e \in E(G)$ (unweighted case).

Referring to multiple facility location, we observe that in several real applications the location of a set of paths is preferable to a single one. Therefore, the location of $p \geq 2$ continuous paths on trees optimizing the three classical objective functions seems to be a viable research line to be developed.

Table 2 Center path location problems

Paper	Underlying graph	Characteristics	Comments
Bhattacharya et al. (2006, 2009)	Tree	Discrete/continuous length constr. conditional/unconditional	$O(n)$ algorithm Optimal
Tami and Lowe (1992)	Tree	Discrete/continuous $p \geq 2$ disjoint paths p input variable	$O(n^3 p^2)$ algorithm
Hakimi et al. (1993), Puerto et al. (2014)	Cactus graph	Discrete/continuous length constr. $p \geq 2$ disjoint paths	NP-hard

Table 3 Center–median path location problems

Paper	Underlying graph	Characteristics	Comments
Averbakh (1999)	Tree	Discrete	$O(n)$ algorithm Optimal
Puerto et al. (2006)	Tree	Discrete length constr.	$O(n \log n)$ algorithm

Table 4 Path location problems with other objective functions

Paper	Underlying graph	Characteristics	Objective functions	Comments
Puerto et al. (2012)	Tree	Discrete/continuous length constr.	Range center subject to median median subject to center	$O(n^2)$ algorithm $O(n^3)$ algorithm $O(n^3)$ algorithm
Puerto et al. (2009)	Tree	Discrete/continuous length constr.	Variance	$O(n^2)$ algorithm
Puerto et al. (2014)	Tree	Discrete $p = 2$ intersecting paths	Expected loss	$O(n^2)$ algorithm NP-commplete on cactus graps disjoint paths may not exist
Ye (2017)	Tree	Discrete	Regret center Regret center–median	$O(n \log n)$ algorithm $O(n^4)$
Ye and Wang (2015)	Tree	Discrete	Regret median	$O(n^2)$ algorithm

Table 5 Median tree location problems

Paper	Underlying graph	Characteristics	Comments
Tamir (1998)	Tree	Discrete	Fully polynomial time approximation scheme
Kim et al. (1996)	Tree	Discrete/continuous Median and set up cost	$O(n^2)$ algorithm
Tamir et al. (2005)	Star graphs	Discrete conditional case	NP-hard
Wang (2000)	Tree	Continuous	$O(\log n \log\log n)$ EREW PRAM model using $O(n)$ work
Shioura and Uno (1997)	Tree	Discrete subtree with exactly k leaves fixed k	$O(n)$ algorithm Optimal
Shioura and Uno (1997)	Tree	Discrete subtree with exactly k leaves for all k	$O(n \log n)$ algorithm Optimal
Becker et al. (2002)	Tree	Discrete subtree with at most k leaves diameter at most ℓ	$O(n^2 \log n)$ algorithm
Wang et al. (2006)	Tree	Discrete subtree with exactly k leaves diameter at most ℓ	$O(n^2)$ algorithm

Table 6 Center tree location problems

Paper	Underlying graph	Characteristics	Comments
Shioura and Shigeno (1997)	Tree	Discrete/continuous	$O(n)$ algorithm Optimal
Bhattacharya et al. (2006, 2009)	Tree	Discrete/continuous conditional case	$O(n)$ algorithm Optimal
Wang (1998)	Tree	Subtree with exactly k leaves vertex-unweighted tree	$O(n)$ algorithm Optimal

Table 7 Center–median tree location problems

Paper	Underlying graph	Characteristics	Comments
Tamir et al. (2002)	Tree	Discrete continuous	NP-hard $O(n \log n)$ algorithm
Dvir and Segal (2008)	Tree	Discrete subtree with exactly k leaves diameter at most ℓ	$O(n^2)$ algorithm

Also, path location problems minimizing a given equity measure are worth further investigation (see Table 4). Actually, several authors have already pointed out that the traditional criteria are not able to catch the dispersion of the clients' demand in location problems. Some interesting contributions appeared in the recent literature. However, there are some problems that are still open. For instance, for the regret objective function, the problem of locating a path with uncertainty on the edge weights or on both edge and vertex weights has been left out, as well as the problem including an additional constraint on the length of the path.

As shown by this survey, at the moment, scientific production on subtree location on graphs is far less developed than the one on path location. For example, the majority of the many optimization criteria that were studied for paths are yet to be investigated for the more complex structure of a subtree. The relevance of this problem is motivated by many real life applications in which this type of models fits well. On the other hand, it is clear that the problem of locating a subtree appears to be more difficult than the others, but this should be taken as a challenge for those who are interested in this research area (Tables 5, 6, 7).

References

Alstrup S, Holm J, Lichtenberg KD, Thorup M (2005) Maintaining information in fully dynamic trees with top trees. ACM Trans Algorithms 1:243–264

Alstrup S, Lauridsen PW, Sommerlund P, Thorup M (1997) Finding cores of limited length. In: Proceedings of 5th international workshop on algorithms and data structures (WADS), lecture notes in computer science, vol 1272. Springer, Berlin, pp 45–54

Alstrup S, Lauridsen PW, Sommerlund P, Thorup M (2001) Finding cores of limited length, IT-C Technical Report Series 2000-4, University of Copenhagen

Averbakh I, Berman O (1999) Algorithms for path medi-centers of a tree. Comput Oper Res 26:1395–1409

Balasubramanian S, Harini S, Rangan CP (2009) Core and conditional core path of specified length in special classes of graphs. In: Das S, Uehara R (eds) WALCOM: algorithms and computation. WALCOM 2009. Lecture notes in computer science, vol 5431, Springer, Berlin, Heidelberg, pp 262–273

Becker RI, Perl Y (1985) Finding the two-core of a tree. Discrete Appl Math 11:103–113

Becker RI, Lari I, Scozzari A, Storchi G (2002) Efficient algorithms for finding the (k, ℓ)-core of tree networks. Networks 40:208–215

Becker RI, Chang YI, Lari I, Scozzari A, Storchi G (2002) Finding the ℓ-core of a tree. Discrete Appl Math 118:25–42

Becker RI, Lari I, Scozzari A (2007) Algorithms for central-median paths with bounded length on trees. Eur J Oper Res 179:1208–1220

Becker RI, Lari I, Scozzari A, Storchi G (2007) The location of median paths on grid graphs. Ann Oper Res 150:65–78

Benkoczi R, Bhattacharya B, Tamir A (2009) Collection depots facility location problems in trees. Networks 53:40–62

Bhattacharya B, Shi Q, Tamir A (2009) Optimal algorithms for the path/tree-shaped facility location problems in trees. Algorithmica 55:601–618

Bhattacharya B, Hu Y, Shi Q, Tamir A (2006) Optimal algorithms for the path, tree-shaped facility location problems in trees, ISAAC, lecture notes in computer science, vol 4288. Springer, Berlin, pp 379–388

Bhattacharyya B, Dehne F (2008) Using spine decomposition to efficiently solve the length-constrained heaviest path problem for trees. Inf Process Lett 108:293–297

Boffey B (1998) Efficient solution methods for covering tree problems. TOP 6:205–221

Caceres T, López-de-los-Mozos MC, Mesa JA (2004) The path-variance problem on tree networks. Discrete Appl Math 145:72–79

Díaz-Báñez JM, Mesa JA, Schöbel A (2004) Continuous location of dimensional structures. TOP 154:22–44

Dvir A, Segal M (2008) The (k, ℓ) coredian tree for ad hoc networks. In: IEEE the 28th international conference on distributed computing systems workshops. https://doi.org/10.1109/ICDCS

Frederickson GN, Johnson DB (1983) Finding the k-th paths and p-centers by generating and searching good data structures. J Algorithms 4:61–80

George JW, Revelle CS (2003) Bi-objective median subtree location problems. Ann Oper Res 122:219–232

Hakimi SL, Schmeichel EF, Labbé M (1993) On locating path-or tree-shaped facilities on networks. Networks 23:543–555

Halpern J (1976) The location of a center–median convex combination on an undirected tree. J Reg Sci 16:237–245

Halpern J (1978) Finding minimal center–median convex combination (cent-dian) of a graph. Manag Sci 24:534–544

Hassin R, Tamir A (1986) Efficient algorithms for optimization and selection on Series-parallel graphs. SIAM J Algebraic Discrete Methods 7:379–389

Hedetniemi SM, Cockaine EJ, Hedetniemi ST (1981) Linear algorithms for finding the Jordan centre and path centre of a tree. Transp Sci 15:98–114

Itai A, Papadimitriou CH, Szwarcfiter JL (1982) Hamilton paths in grid graphs. SIAM J Comput 4:676–686

Kalcsics J, Nickel S, Puerto J (2003) Multifacility ordered median problems on networks: a further analysis. Networks 41:1–12

Kim TU, Lowe TJ, Tamir A, Ward JE (1996) On the location of a tree-shaped facility. Networks 28:167–175

Labbé M, Laporte G, Martín IR, González JJS (2005) Locating median cycles in networks. Eur J Oper Res 160:457–470

Laporte G, Martín IR (2007) Locating a cycle in a transportation or a telecommunications network. Networks 50:92–108

Laporte G, Nickel S, Saldanha da Gama F (2015) Location science. Springer International Publishing, Switzerland

Lari I, Ricca F, Scozzari A (2008) Comparing different metaheuristic approaches for the median path problem with bounded length. Eur J Oper Res 190:587–597

Lari I, Ricca F, Scozzari A, Becker RI (2011) Locating median paths on connected outerplanar graphs. Networks 57:294–307

Megiddo N (1983) Linear-time algorithms for linear programming in R^3 and related problems. SIAM J Comput 12:759–776

Mesa JA, Boffey TB (1996) A review of extensive facility location in networks. Eur J Oper Res 95:592–603

Mesa JA, Puerto J, Tamir A (2003) Improved algorithms for several network location problems with equality measures. Discrete Appl Math 130:437–448

Minieka E (1985) The optimal location of a path or tree in a tree network. Networks 15:309–321

Morgan CA, Slater JP (1980) A linear algorithm for a core of a tree. J Algorithms 1:247–258

Nickel S, Puerto J (1999) A unified approach to network location. Networks 34:283–290

Nickel S, Puerto J (2005) Location theory: a unified approach. Springer, Berlin

Novik A (1996) Improved algorithms for locating tree or path shaped facilities on a tree network, M.S. Thesis, School of Mathematical Sciences, Tel Aviv University, Tel Aviv, Israel

Peng S, Lo W-T (1996) Efficient algorithms for finding a core of a tree with a specified length. J Algorithms 20:445–458

Peng S, Stephens AB, Yesha Y (1993) Algorithms for a core and k-tree core of a tree. J Algorithms 15:143–159

Puerto J, Tamir A (2005) Locating tree-shaped facilities using the ordered median objective. Math Programm 102:313–338

Puerto J, Rodríguez-Chía AM, Tamir A, Perez-Brito D (2006) The bi-criteria doubly weighted center–median path problem on a tree. Networks 47:237–247

Puerto J, Ricca F, Scozzari A (2009) Extensive facility location problems on networks with equity measures. Discrete Appl Math 157:1069–1085

Puerto J, Ricca F, Scozzari A (2009) The continuous and discrete path-variance problems on trees. Networks 53:221–228

Puerto J, Ricca F, Scozzari A (2011) Minimax regret path location on trees. Networks 58:147–158

Puerto J, Ricca F, Scozzari A (2012) Range minimization problems in path-facility location on trees. Discrete Appl Math 160:2294–2305

Puerto J, Ricca F, Scozzari A (2014) Reliability problems in multiple path-shaped facility location on networks. Discrete Optim 12:61–72

Puerto J, Rodríguez-Chía AM, Tamir A (2017) Revisiting k-sum optimization. Math Program 165:579–604

Richey MB (1990) Optimal location of a path or tree on a network with cycles. Networks 20:391–407

Rozanov M (2015) The nestedness property of the convex ordered median location problem on a tree. MSc thesis, Tel-Aviv University. http://primage.tau.ac.il/libraries/theses/exeng/free/3257656.pdf

Rozanov M, Tamir A (2018) The nestedness property of location problems on the line. TOP. https://doi.org/10.1007/s11750-018-0471-x

Schöbel A (2015) Location of dimensional facilities in a continuous space. In: Laporte G et al (eds) Location science. Springer International Publishing, Cham

Shioura A, Shigeno M (1997) The tree center problems and the relationship with the bottleneck knapsack Problem. Networks 29:107–110

Shioura A, Uno T (1997) A linear time algorithm for finding a k-tree core. J Algorithms 23:281–290

Slater PJ (1982) Locating central paths in a graph. Transp Sci 15:1–18

Takamizawa K, Nishizeki T, Saito N (1982) Linear-time computability of combinatorial problems on series-parallel graphs. J ACM 29:623–641

Tamir A (1996) An $O(pn^2)$ algorithm for the p-median and related problems on tree graphs. Oper Res Lett 19:59–64

Tamir A (1998) Fully polynomial approximation schemes for locating a tree-shaped facility: a generalization of the knapsack problem. Discrete Appl Math 87:229–243

Tamir A (2004) Sorting weighted distances with applications to objective function evaluations in single facility location problems. Oper Res Lett 32:249–257

Tamir A, Lowe TJ (1992) The generalized p-forest problem on a tree network. Networks 22:217–230

Tamir A, Puerto J, Pèrez-Brito D (2002) The centdian subtree on the tree networks. Discrete Appl Math 118:263–278

Tamir A, Puerto J, Mesa JA, Rodriguez-Chia AM (2005) Conditional location of path and tree shaped facilities on trees. J Algorithms 56:50–75

Tang H, Cheng TCE, Ng CT (2012) A note on the subtree ordered median problem in networks based on nestedness property. J Ind Manag Optim 8:41–49

Wang B-F, Lin J-J (2000) Finding a Two-Core of a tree in linear time, ISAAC, lecture notes in computer science, vol 1969. Springer, Berlin, pp 467–478

Wang B-F (1998) Finding a k-tree core and a k-tree center of a tree network in parallel. IEEE Trans Parallel Distrib Syst 9:186–191

Wang B-F (2000) Efficient parallel algorithms for optimally locating a path and a tree of a specified length in a weighted tree network. J Algorithms 34:90–108

Wang B-F (2002) A 2-Core of a tree in Linear time. SIAM J Discrete Math 15:193–210

Wang B-F, Ku S-C, Shi K-H (2001) Cost-optimal parallel algorithms for the tree bisector and related problems. IEEE Trans Parallel Distrib Syst 12:888–898

Wang B-F, Peng S, Yu H-Y, Ku S-C (2006) Efficient algorithms for a constrained k-tree core problem in a tree network. J Algorithms 59:107–124

Wang B-F, Lin T-C, Lin C-H, Ku S-C (2008) Finding the conditional location of a median path on a tree. Inf Comput 206:828–839

Ye J-H (2017) Improved algorithms for minmax regret path location problems on trees, Ph.D. Dissertation, Department of Computer Science, Nation Tsing Hua University

Ye J-H, Wang B-F (2015) On the minmax regret path median problem on trees. J Comput Syst Sci 81:1159–1170