# Polynomial Algorithms for Partitioning a Tree into Single-Center Subtrees to Minimize Flat Service Costs* 

N. Apollonio, I. Lari, F. Ricca, and B. Simeone<br>Dip. Statistica, Probabilità e Statistiche Applicate Università di Roma "La Sapienza", piazzale Aldo Moro 5, 00185 Roma, Italy

J. Puerto

Universidad de Sevilla Dep. to de Estadistíca e Investigación Operativa, Sevilla, Spain


#### Abstract

This paper deals with the following graph partitioning problem. Consider a connected graph with $\boldsymbol{n}$ nodes, $p$ of which are centers, while the remaining ones are units. For each unit-center pair there is a fixed service cost and the goal is to find a partition into connected components such that each component contains only one center and the total service cost is minimum. This problem is known to be NP-hard on general graphs, and here we show that it remains such even if the service cost is monotone and the graph is bipartite. However, in this paper we derive some polynomial time algorithms for trees. For this class of graphs we provide several reformulations of the problem as integer linear programs proving the integrality of the corresponding polyhedra. As a consequence, the tree partitioning problem can be solved in polynomial time either by linear programming or by suitable convex nondifferentiable optimization algorithms. Moreover, we develop a dynamic programming algorithm, whose recursion is based on sequences of minimum weight closure problems, which solves the problem on trees in $O(n p)$ time. © 2007 Wiley Periodicals, Inc. NETWORKS, Vol. 51(1), 000-000 2008


Keywords: tree partitioning; generalized packing problem; -1,0,1-perfect matrices; maximum closure; dynamic programming; NP-Complete problems

## 1. INTRODUCTION

Let $G=(V, E)$ be a connected graph with a set of $n$ vertices $V$ and a set of edges $E$. Suppose the subset $S \subset V$ is the set of $p=|S|$ fixed centers, which correspond to service points, while the subset $U=V \backslash S$ is the set of the $n-p$

[^0]units that must be served. Each center in $S$ provides a service, which is addressed to the units in $U$. Each unit is required to be connected - perhaps through other units - to exactly one center. There is a cost function $c: U \times S \longrightarrow \mathbb{R}$ which associates a cost $c_{i s}$ to each pair $(i, s), i \in U, s \in S$. The objective is to provide service to the set of units, by assigning each unit to exactly one center at minimum cost. We consider the case of flat service costs, that is, costs due only to the assignment of unit $i$ to center $s$.

This general framework fits a variety of real-life problems arising in all those service systems in which units must be assigned to centers and the service costs do not depend exclusively on the topology of the network or on the intensity of the rendered service (flat service costs). Actually, these problems can be formulated as graph partitioning ones where the graph $G$ may be:

> a physical graph: the graph corresponds to a TV or optical fibre cable network, to a pipeline, to a road network, to a river arm, etc.
> a contiguity graph: a territory divided into $n$ elementary units is given, each vertex of $G$ represents a territorial unit; an edge between two vertices exists if and only if the two corresponding units are neighboring.

In both cases, each customer-unit is connected to its center, say $s$, by a path and it is said to be served by $s$. A connected partition of $G$ is a partition of $V$ into non empty subsets, called components, such that each component induces a connected subgraph of $G$; the partition is centered if each component contains exactly one center. The cost of a centered partition of $G$ is given by the sum of the service costs $c_{i s}$, for all $i \in U$, $s \in S$, such that $i$ is served by $s$.

The problem can be stated as follows:
Minimum Cost Centered Partition Problem (MCP): Given a connected graph $G$, a subset $S \subset V$ and a cost function $c$, find a connected centered partition of $G$ with minimum cost.

In the following, we will denote an instance of problem MCP by ( $G, S, c$ ).

Notice that, given a cost function $c$, a minimum cost centered partition with respect to $c$ remains optimal when a constant $M$ is added to all the costs, since in each feasible solution the resulting change of the total cost is $M(n-p)$, a constant. In particular, by adding to the costs a large positive $M$, one can make them strictly positive; by adding a large negative $M$, one can make them strictly negative and then convert the minimization problem into an equivalent maximization one with positive benefits.

MCP can also be formulated in terms of spanning forests of $G$ as follows. Let $F$ be a spanning forest of $G$, and $\mathcal{F}(G)$ be the set of all the spanning forests of $G$. Let $T_{s}(F)$ denote the subtree of $F$ which contains the center $s$. For the sake of simplicity, in the following we shall identify the tree $T_{s}(F)$ with either its set of vertices or its set of edges, according to the context. This set will be denoted simply by $T$ when additional specification is not necessary. Notice that each spanning forest of $G$ defines a connected partition of $G$, and each connected partition of $G$ can be represented by a spanning forest of $G$. Then, problem MCP can be formulated as the problem of finding a spanning forest of $G$ such that each tree in $F$ contains exactly one center and the total service cost is minimized. We can formulate this problem as follows:

$$
\begin{equation*}
\min _{F \in \mathcal{F}(G)}\left\{\sum_{s \in S} \sum_{i \in T_{s}(F)} c_{i s}:|T \cap S|=1, \forall T \in F\right\} \tag{1}
\end{equation*}
$$

As an example of application suppose that a company providing boat repair and maintenance services in dockyards along a river basin wishes to design service zones for its dockyards. Each service zone must contain only one dockyard, which must be reachable by boat from each village in the zone. A village can be served by only one dockyard. Given an estimate of the company revenue when each village $i$ is served by each dockyard $s$, the service zones should be planned so as to maximize total revenue. This case is of particular interest since the typical structure of a river basin is tree-like.

Other real-life applications can be found in the management of optical fibre networks, highway networks, airlines routes networks, telephone networks. In all these cases the assumption of a tree-network is quite reasonable. For example, highway systems often have a tree structure, while airlines route structures and sector or area telephone networks are often configured as a set of connected star-like trees [17].

Some interesting applications of MCP are related to telecommunication network problems. Consider a network infrastructure, such as a cable tv network, and suppose that centers correspond to broadcasting stations, while units are those nodes in the network that must be served. In many cases the network is property of the State, which gives the network out by contract to different private providers. In this case the
network must be partitioned into connected subnetworks pertaining to different providers, for ease of maintenance and in order to avoid control conflicts. The costs associated with each center-unit pair correspond to the bid of the provider of the center to obtain the unit and mainly depend on marketing policies, spurs, and local factors related both to the center and to the unit. To guarantee that the providers charge low rates to the customers/citizens, the State will partition the network among the providers in order to minimize the total cost. On the other hand, if the owner of the infrastructure network is a private enterprise, the bids of the local providers may be regarded as owner's returns rather than collectivity costs; in this case the owner wishes to maximize its total return. This problem can again be modelled as a MCP one in maximization form. Notice that when the network is located in an elongated region, such as a coastal one, or when it is an early-stage network structure in a developing country, the hypothesis of a tree network is realistic.

Finally, we should mention that the initial motivation of our work was an application to political districting (PD) which consists in drawing a district map for political elections. It can be formulated as an MCP when the territory involved is represented by a contiguity graph. The fact that one looks for a connected partition reflects the usual contiguity requirement for the territorial units of a district. Other typical PD criteria are population balance and compactness. The former criterion is often enforced by bounds on the district population, while the latter one can be met through the minimization of the moment of inertia w.r.t. the district center (see, for example, Ref. [8]). Notice that, when inertia is adopted as a measure of compactness, for each district some vertex must be identified as the center of that district. Actually, there are a variety of other compactness measures that are defined on the basis of district centers [10]. Thus, the PD problem can be formulated as an MCP with side population balance constraints. However, by including such constraints into a lagrangean objective function, one obtains an MCP as in (1).

Besides being motivated by a variety of real-life applications the study of MCP on trees is theoretically justified by the status of its complexity. As a matter of fact, in Ref. [4] it has been shown that problem MCP is NP-hard on general graphs even in the case of two centers. Then the natural question that arises is whether the same problem is polynomially solvable in simpler graph topologies. As for some well-known problems in location analysis, such as the $p$ center or $p$-median [13,14], this is the case when the model is restricted to tree networks. Finally, our interest in problems over trees arises also from the following theoretical result, proven in Refs. [2, 3, 15, 16]: for a broad class of objective functions arising in numerous applications and for any connected graph $G$, there is always a spanning tree $T$ of $G$ such that some optimal connected partition of $T$ is also an optimal connected partition of $G$. The proof is non constructive, but it leads to an effective heuristic for finding a "good" connected partition of $G$ through the (optimal or approximate) solution of a sequence of restrictions of the problem to


FIG. 1. Consider the tree in (a), where black vertices are fixed as centers and the numbers refer to the costs $c_{i s}$ for each unit-center pair $(i, s)$. In (b) the solution found by the greedy algorithm is shown, while (c) shows the optimal solution.
spanning trees of $G$ (see [15]). These results easily extend to our problem.

Thus, the main goal of this paper is to answer whether MCP is polynomially solvable on trees. Looking for an answer to this question, the reader may notice that the feasible solutions to MCP correspond to the bases of a matroid on the set of edges of the input graph. Nevertheless, the greedy algorithm does not apply here. The trouble is that the costs are defined on unit-center pairs, rather than on the elements of the ground set, namely, the edges (see Fig. 1). It is worth observing that if 10 is replaced by an arbitrarily large number $M$ in Figure 1, the Greedy Algorithm may perform as badly as possible. This feature makes this problem even more challenging.

In this paper we provide some polynomial time algorithms for MCP on trees. First of all, in spite of the discrete nature of the problem, we derive an LP formulation, whose coefficient matrix has $-1,0,1$ entries and hence is solvable in strongly polynomial time. Then, we obtain an alternative formulation with a smaller number of variables where a piecewise-linear concave function must be maximized subject to linear constraints. The latter problem can be solved in polynomial time, too, and the resulting complexity is typically lower. Finally, we describe a combinatorial algorithm with $O(n p)$ complexity in which dynamic programming techniques are combined with maximum weight closure ones.

The paper is organized as follows. In Section 2 we provide two Integer Linear Programming models for the MCP problem on trees and prove the integrality of the feasible polyhedra of their continuous relaxations: in Section 2.1 we provide a generalized set packing formulation of the problem, while in Section 2.2 we provide a branching formulation of the same problem. Section 3 describes a fast polynomial time algorithm for trees, while Section 4 is devoted to some further complexity results.

## 2. LINEAR AND NONLINEAR PROGRAMMING MODELS FOR TREES

In this section we present two Integer Linear Programming formulations for problem MCP on trees. Both formulations are interesting on their own because the corresponding polyhedra are integral and highlight different aspects of the
problem: while the first one relies on the intersection properties of the family of centered partitions, the second one relies on the directed branching nature of such partitions. Moreover, they are concise in the sense that they use only polynomially many constraints and variables; furthermore, their coefficient matrices have $-1,0,1$ entries. Therefore, in both cases strongly polynomial time linear programming algorithms can be found. In addition, the latter linear program can be reformulated as the maximization of a concave nondifferentiable function subject to linear constraints. Although this problem is nonlinear, its size is much smaller. In this way we trade size for nonlinearity with an overall complexity saving. For a proof of the complexity bounds in this section we refer the reader to Ref. [1].

Given a tree $T=(V, E),|V|=n$, as before we denote by $U$ the set of all units and by $S$ the set of all centers. Given a unit $i \in U$ and a center $s \in S$, let us define a path from $i$ to $s$ to be free if it does not contain any center but $s$. For each unit $i$, we define the set $S_{i}=\{s:$ the path from $i$ to $s$ is free $\}$. Notice that $i$ can be served only by the centers in $S_{i}$. Moreover, we denote by $c(i)=\left(c_{i s}\right)_{s \in S_{i}}$ the cost vector restricted to the pairs $(i, s), s \in S_{i}$.

The following lemma shows that, without loss of generality, in a tree one may assume that centers and leaves coincide.

Lemma 1 (Leaf Property). Any instance ( $T, S, c$ ) of $M C P$, where $T$ is a tree and $S$, c are arbitrary, can be reduced, preserving optimality, to a set of independent instances $\left(T_{i}, S_{i}, c(i)\right), i=1,2, \ldots, k, k \leq n-p$, where the $T_{i}$ 's are subtrees of $T$ such that: (1) the union of all the $T_{i}$ 's is equal to the whole tree $T$; (2) any pair of subtrees $T_{i}$ and $T_{j}, i \neq j$ intersects in at most one node, this node being a center; (3) $S_{i}$ is the set of leaves of $T_{i}$.

Proof. Instead of giving a formal proof, we refer the reader to Figure 2, which captures the gist of the proof.

On the basis of the result of Lemma 1, from now on, we assume that in our trees the leaves and the centers coincide.

Notice that in a tree, removing (cutting) $p-1$ edges results into a connected partition of $T$ into $p$ components, and every connected partition can be obtained in this way.

### 2.1. A Generalized Set Packing Formulation

In the first formulation we use the binary variables:

$$
y_{i s}= \begin{cases}1 & \text { if unit } i \text { is assigned to center } s  \tag{2}\\ 0 & \text { otherwise }\end{cases}
$$

In the following model we use order constraints, so that the partition is connected, and multiple choice constraints so that each unit is assigned to exactly one center. These constraints guarantee that the partition is centered. The Integer


FIG. 2. Any instance of MCP on a tree $T$ can be reduced to a set of independent instances of MCP on subtrees of $T$ and the set of the centers of $T$ coincides with the set of leaves of the subtrees of $T$ : (a) A tree $T$; (b) Any center in $T$ is a leaf in at least one subtree of $T$; (c) Any leaf in the subtrees of $T$ is a center of $T$. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

Linear Programming model based on order constraints is the following:

$$
\begin{array}{ll}
\min & \sum_{i \in U} \sum_{s \in S} c_{i s} y_{i s} \\
& \\
y_{i s} \leq y_{j s} & \forall i \in U, j \in U, s \in S \\
\sum_{s \in S} y_{i s}=1 & \forall i \in U \\
& \text { such that } j \text { lies in } P_{i s}  \tag{3}\\
y_{i s} \in\{0,1\} & \forall i \in U, s \in S
\end{array}
$$

where $P_{i s}$ is the unique path from $i$ to $s$ in the tree $T$. This model has $O(n p)$ variables and $O\left(n^{2} p\right)$ constraints. Actually, the ILP model (3) admits a more concise formulation. For any given pair $(i, s), i \in U, s \in S$, such that $i$ is not adjacent to $s$, let $j(i, s)$ be the vertex adjacent to $i$ in the unique path from $i$ to $s$. Then, by transitivity, the $O\left(n^{2} p\right)$ order constraints in (3) are equivalent to the $O(n p)$ order constraints:

$$
\begin{equation*}
y_{i s} \leq y_{j(i, s) s} i \in U, s \in S, i s \notin E . \tag{4}
\end{equation*}
$$

Hence (3) can be reformulated as

$$
\begin{array}{ll}
\min & \sum_{i \in U} \sum_{s \in S} c_{i s} y_{i s} \\
& y_{i s} \leq y_{j(i, s), s} \\
& \forall i \in U, s \in S, i s \notin E \\
& y_{s \in S} y_{i s}=1 \quad \forall i \in U  \tag{5}\\
& y_{i s} \in\{0,1\} \quad \forall i \in U, s \in S .
\end{array}
$$

An integral feasible solution to (5) (and hence to (3)) will be called a consistent assignment. Let $M$ be the 0,1 , -1 coefficient matrix of (5) - or of (3) - and let $n(M)$ be the column vector whose $i$-th component is the number of -1 's in the $i$-th row of $M$. The fractional generalized set packing polytope (see for instance Ref. [5]) is the set $\{y: M y \leq \mathbf{1}-n(M), \mathbf{0} \leq y \leq \mathbf{1}\}$, where $\mathbf{0}$ and $\mathbf{1}$ are the vectors whose components are all equal to 0 and to 1 ,
respectively. After replacing in (5)- or in (3)- the equations by inequalities of the $\leq$ type and the integrality constraints by the non negativity constraints $y_{i s} \geq 0$, one sees that the corresponding feasible polytope is precisely of the fractional generalized set packing type (inequalities $y \leq \mathbf{1}$ are already implied by the multiple choice constraints). Remark that there is exactly one component equal to -1 in each row corresponding to an order constraint and none in each row corresponding to a multiple choice constraint.

Remark 1. Notice that if either the multiple choice constraints or the order constraints are deleted, the constraint matrix of the resulting system is totally unimodular. In particular, when the cost function is metric, i.e., when there exists a metric $d$ on $V \times V$ such that $c_{i s}=d(i, s)$ for each $i \in U$ and $s \in S$, the order constraints in (5), being redundant, can be deleted and the corresponding polytope is integral. However the overall matrix is not totally unimodular. In fact, $M$ is not even balanced. Indeed $M$ can contain an unbalanced cycle submatrix. Recall (see for instance Ref. [5]) that an unbalanced cycle submatrix of a $0,1,-1$ matrix $A$ is a square submatrix having exactly two nonzero entries per row and per column and such that the sum of its entries is not a multiple of four. The reader can verify that a cycle of this type can be found in a 2-spider (i.e., a tree obtained from a star by the insertion of a node within each edge). It is worth noticing that, for any instance of MCP on a tree T, every unbalanced cycle submatrix of the coefficient matrix of (5) intersects an odd number of rows corresponding to multiple choice constraints. Moreover, the sum of the coefficients of every order constraint is zero.

After Remark 1, the constraint matrix $M$ is not balanced (hence not totally unimodular). Nonetheless it is still a nice matrix. Recall that a $-1,0,1$-matrix is perfect if and only if the corresponding generalized packing polytope is integral (see Ref. [5]).

Theorem 1. The $0,1,-1$ coefficient matrix $M$ of (5) is perfect.

Proof. Let us replace the equations in (5) by inequalities of the $\leq$ type, and let $Q$ be the resulting feasible polytope. We need to show that $Q$ is integral. Let $y$ be an arbitrary point of $Q$. For each $s \in S$, let $H_{s} \equiv H_{s}(y)=\left\{i: i \in U, y_{i s}>0\right\}$ and let $H \equiv H(y)=H_{1} \cup \ldots \cup H_{p}$.

Then, in view of the order constraints in (5) - or equivalently of (3) - the following property holds:

$$
\begin{equation*}
i \in H_{s} \text { and } j \in P_{i s} \Rightarrow j \in H_{s} . \tag{6}
\end{equation*}
$$

Claim: $H$ can be partitioned as $H=U_{1} \cup \ldots \cup U_{p}$, where, for each $s \in S, U_{s} \subseteq H_{s}$ and $\{s\} \cup U_{s}$ induces a subtree of $T$ ( $U_{s}$ is allowed to be empty).

The required partition can be obtained as follows:
$U_{1}=H_{1} ; U_{2}=H_{2} \backslash H_{1} ; \cdots ; U_{p}=H_{p} \backslash\left(H_{1} \cup \cdots \cup H_{p-1}\right)$.

The fact that $\{s\} \cup U_{s}$ induces a subtree of $T$ for each $s \in S$ follows from (6). Notice that any two such subtrees are disjoint. Therefore, the collection of all such subtrees is a centered (but generally non spanning) forest of $T$.

Hence, if one defines $v_{i s}=1$ if $i \in U_{s}$ and $v_{i s}=0$ otherwise, one gets an integral feasible point (and thus an extreme point) $v$ of the polytope $Q$.

Now let $\delta$ be the smallest positive component of $y$, and let $y^{\prime}=y-\delta v$. Then $y^{\prime}$ still belongs to $Q$ and has at least one more null component than $y$. In this way, starting from $y^{0}=y$, one obtains a sequence $y^{0}, y^{1}, \ldots$, of points of $Q$ such that, for each $k, y^{k+1}=y^{k}-\delta^{k} v^{k}$, where $\delta^{k}>0$ and $v^{k}$ is a binary point of $Q$. Since each $y^{k+1}$ has at least one more null component than $y^{k}$, there must be a smallest $N \leq(n-p) p$ such that $y^{N+1}=\mathbf{0}$. Hence one gets

$$
\begin{equation*}
y=\delta^{0} v^{0}+\cdots+\delta^{N} v^{N} \tag{7}
\end{equation*}
$$

Let $y_{i s}^{N}$ be a positive component of $y^{N}$. Since the sequence $\left\{y_{i S}^{k}\right\}$ is nonincreasing, for each $k=0,1 \ldots, N$, one has $y_{i s}^{k}>$ 0 . Hence $i \in H\left(y^{k}\right)$. Therefore,

$$
\begin{equation*}
v_{i 1}^{k}+\ldots+v_{i p}^{k}=1 \tag{8}
\end{equation*}
$$

From (7) and (8) it follows that

$$
\delta^{0}+\ldots+\delta^{N}=y_{i 1}+\cdots+y_{i p} \leq 1
$$

Since the null vector $\mathbf{0}$ is an (extreme) integral point of $Q$, the point $y$ can be written as a convex combination of $N+1$ integral points of $Q$, namely,

$$
y=\left(1-\delta^{0} \ldots-\delta^{N}\right) \mathbf{0}+\delta^{0} v^{0}+\ldots+\delta^{N} v^{N}
$$

In conclusion, $Q$ is the convex hull of its integral points; therefore, $Q$ is integral.

Remark 2. Since $N \leq(n-p) p$, we have actually given a direct proof of Carathéodory's Theorem for the polytope $Q$.

Corollary 1. The feasible polytope $P$ of (5) is integral.
Proof. $P$ is a face of $Q$.
Remark 3. Whereas the extreme points of $Q$ correspond to (possibly non spanning) centered forests, those of $P$ correspond to spanning centered forests.

After Corollary 1 one can solve (5) by linear programming. Since all entries of the matrix $M$ are $-1,0$, or 1 the corresponding linear program can be solved in strongly polynomial time after a well known result of Tardos (1986) (see Ref. [18]). Tardos' bound in our case is $O\left((n p)^{6}\right)$ and thus it is much lower than in the general case (details and further results can be found in Ref. [1]).

Theorem 1 exploits the intersection properties of the family of the consistent assignments of $T$; such properties are inherited by the "chordal graph" structure of the family of
centered partitions of $T$. The next Theorem 2 provides a deeper insight about the nature of the polytope $Q$. To elicit this nature consider the family $\mathcal{T}$ of all subtrees of $T$ containing exactly one center and at least one unit. Each member $T^{\prime}$ of $\mathcal{T}$ is thus a subtree of $T$ induced by a set of the form $\{s\} \cup U^{\prime}$, for some $s \in S$ and some non empty subset $U^{\prime}$ of units. Let $G(\mathcal{T}) \equiv G$ be the intersection graph of $\mathcal{T}$ : the vertex set of $G$ is $\mathcal{T}$ and two vertices are joined by an edge in $G$ if the corresponding subtrees intersect. The following result gives us a view of the structure of $Q$ in terms of the clique polytope $Q(G)$ of the chordal graph $G$, namely, the polytope
$Q(G)=\left\{x \in \mathbb{R}_{+}^{\mathcal{T}}: \sum_{v \in K} x_{v} \leq 1, \forall K \in \mathcal{K}, x_{v} \geq 0, \forall v \in V(G)\right\}$,
where $\mathcal{K}$ is the family of maximal cliques of $G$.

Theorem 2. The polytope $Q$ is the image of the clique polytope $Q(G)$ of $G$ under an integral linear mapping.

Proof. Let $F$ be any centered (not necessarily spanning) forest of $T$ and let $F_{1}, \ldots F_{t}, t \leq p$, be the components of $F$ containing at least one unit. Let $s_{h} \in S$ be the only center in $F_{h}$, for $h=1, \ldots, t$. The correspondence between centered forests of $T$ and integral (extreme) points $y$ of $Q$, stated in Remark 3 is the bijection defined by:

$$
y_{i s_{h}}=1 \Leftrightarrow i \in F_{h}, \quad h=1, \ldots t .
$$

Moreover,

$$
\begin{equation*}
y=u^{F_{1}}+\cdots+u^{F_{t}} \tag{9}
\end{equation*}
$$

where, for $h=1, \ldots t, u_{i s}^{F_{h}}=1$ if $i \in F_{s}$ and $s=s_{h}, u_{i s}^{F_{h}}=0$ otherwise. Thus,

$$
\begin{equation*}
\mathcal{S}=\left\{F_{1}, \ldots, F_{t}\right\} \leftrightarrow y=u^{F_{1}}+\cdots+u^{F_{t}} \tag{10}
\end{equation*}
$$

sets a one-to-one correspondence between the family of stable sets of $G$ and the set of integral points of $Q$. The graph $G$ is chordal, hence perfect. It follows that $Q(G)$ is the convex hull of the incidence vectors of its stable sets. Therefore, (10) sets a one-to-one correspondence between the extreme points of $Q(G)$ and those of $Q$. It remains to prove that every extreme point of $Q$ is the image of an integral extreme point of $Q(G)$ under a linear integral mapping. Let $T^{\prime}$ be any member of $\mathcal{T}$ and let $s$ be the unique center in $T^{\prime}$. Let $u^{T^{\prime}}$ be the vector whose entries are indexed by $U \times S$ and are defined by $u_{i t}^{T^{\prime}}=1$, if $t=s$ and $i \in V\left(T^{\prime}\right) \backslash\{s\}$ and $u_{i t}^{T^{\prime}}=0$ otherwise. Let $C$ be the matrix whose columns are the vectors $u^{T^{\prime}}$ for $T^{\prime} \in \mathcal{T}$. By (10), $y=C \chi_{\mathcal{S}}$ is an integral point of $Q$, where $\chi_{\mathcal{S}}$ is the incidence vector in $\mathcal{T}$ of the stable set $\mathcal{S}$ of $G$, and $C$ defines the required integral linear mapping.

### 2.2. An Optimal Branching Formulation

An alternative Integral Linear Programming formulation for problem MCP on trees can be obtained on the basis of the notion of consistent orientation, which provides a different, but useful, perspective on the problem. Given an instance ( $T, S, c$ ) of MCP, a consistent orientation of $T$ is an orientation of some of the edges of $T$ such that:
(i) for each undirected edge $i j$, at most one of the two arcs ( $i, j$ ) and $(j, i)$ is present;
(ii) the outdegree of each unit is 1 ;
(iii) the outdegree of each center is 0 (centers must be roots).

Notice that some edges of $T$ remain undirected. Consistent orientations correspond to a special kind of antibranchings (in-forests, assembly forests): they must also be spanning and centered (in each component the root is the unique center).

Proposition 1. In any consistent orientation of $T$, the number of undirected edges is $|S|-1$.

Proof. The total number of vertices, centers, and units are equal to $n, p$, and $n-p$, respectively. After property (ii), the total number of arcs is $n-p$, so the number of undirected edges is $n-1-(n-p)=p-1$.

Proposition 2. There is a one-to-one correspondence between centered partitions and consistent orientations of $T$.

Proof. With any centered partition $\pi$ of $T$, we associate an orientation $\omega$ of some edges of $T$ as follows. For each unit $i$, let $s$ be the unique center serving $i$, and let $e$ be the first edge along the path from $i$ to $s$. Direct $e$ out of unit $i$. All the edges that are cut in $\pi$ remain undirected. Clearly, $\omega$ is a consistent orientation. Conversely, given any consistent orientation of $T$, the connected components of the corresponding antibranching define a centered partition $\pi$ of $T$.

For any consistent orientation $\omega$ and each unit $i$, there is a unique directed path from $i$ to a center $s(i)$, the one that serves $i$ in the centered partition associated with $\omega$. Then, the cost of any consistent orientation $\omega$ is given by

$$
\sum_{i \in U} c_{i s(i)}
$$

Notice that the cost of $\omega$ coincides with the cost of the centered partition $\pi$ associated with $\omega$. Then, consider the following problem:
Minimum Cost Consistent Orientation Problem (MCO): Given a tree $T$, a set of centers $S$ and a cost function $c$, find a consistent orientation of $T$ with minimum cost.

In view of Proposition 2, we obtain the following result.
Theorem 3. In the case of trees, problems MCP and MCO are mutually reducible in polynomial time.

Now let us formulate an Integer Linear Programming model for MCO. We shall adopt, without loss of generality, the maximization form of MCO (with respect to benefits $c_{i s}^{\prime}$ instead of costs $c_{i s}$ ). Given a consistent orientation $\omega$, for each edge $i j$ of $T$ introduce binary variables $x_{i j}$ and $x_{j i}$; for each unit $i$ and each center $s$, introduce binary variables $y_{i s}$, with the following meaning:

$$
\begin{gathered}
x_{i j}= \begin{cases}1 & \text { if }(i, j) \text { is an arc in } \omega \\
0 & \text { otherwise, }\end{cases} \\
y_{i s}= \begin{cases}1 & \text { if } s=s(i) \\
0 & \text { otherwise }\end{cases}
\end{gathered}
$$

Thus, the $x_{i j}$ 's are decision variables that define the orientation $\omega$, while the $y_{i s}$ 's are auxiliary variables which are needed in order to compute the total benefit of $\omega$ as a linear function.

Let $D$ be the symmetric digraph $(V, A)$, where $A=$ $\{(i, j),(j, i): i j \in E\}$, and let $A_{i s}$ be the set of arcs of the elementary directed path in $D$ from $i$ to $s$. Then MCO admits the following ILP formulation:

$$
\begin{array}{ll}
\max & \sum_{i \in U} \sum_{s \in S} c_{i s}^{\prime} y_{i s} \\
& \\
x_{i j}+x_{j i} \leq 1 & \forall i j \in E \\
\sum_{j:(i, j) \in A} x_{i j}=1 & \forall i \in U \\
x_{s f}=0 & \forall s \in S,(s, f) \in A \\
y_{i s} \leq x_{h k} & \forall i \in U, s \in S,(h, k) \in A_{i s} \\
x_{i j} \in\{0,1\} & \forall(i, j) \in A  \tag{f}\\
y_{i s} \in\{0,1\} & \forall i \in U, s \in S .
\end{array}
$$

Constraints (a), (b), and (c) enforce properties (i), (ii), and (iii) of consistent orientations, respectively. The order constraints (d) and the strict positivity of the benefits $c_{i s}^{\prime}$ imply that, in every optimal solution of (11),

$$
\begin{equation*}
y_{i s}=\min _{(h, k) \in A_{i s}} x_{h k}=\prod_{(h, k) \in A_{i s}} x_{h k}, \quad i \in U, s \in S \tag{12}
\end{equation*}
$$

where the second equality holds since the variables $x_{h k}$ are binary. Thus, $y_{i s}=1$ iff in $\omega$ there is a directed path from $i$ to $s$. It follows that the objective function represents the total benefit of the consistent orientation $\omega$. The above model involves $O(n p)$ variables and $O\left(n^{2} p\right)$ constraints.

Consider the continuous relaxation of (11) obtained by the replacement of (e) and (f) by the non negativity constraints $x_{i j} \geq 0$ and $y_{i s} \geq 0$, respectively; and let $K$ be the corresponding feasible polytope. In view of Theorems 1 and 3, the following result is not surprising.

Theorem 4. The polytope $K$ is integral.
Proof. It will be enough to exhibit an integral linear mapping

$$
y \mapsto\left[\begin{array}{l}
x \\
y
\end{array}\right]=\left[\begin{array}{l}
B \\
I
\end{array}\right] y
$$

that maps the integral polytope $P$ of (5) into $K$. Then the (integral) extreme points of $P$ are mapped into extreme points of $K$ (indeed any convex combination of points of $K$ defines a convex combination with the same coefficients of the $y$-components of these points). Moreover, since the transformation is integral, also the latter ones are integral.

Let $i j$ be any edge of $T$. Let us denote by $T_{i j}^{i}$ and $T_{i j}^{j}$ the two branches of $T$ resulting from the deletion of $i j$ and containing $i$ and $j$, respectively. For any subtree $T^{\prime}$ of $T$, let $\operatorname{Lv}\left(T^{\prime}\right)$ be the set of leaves of $T^{\prime}$.

Given a point $y \in P$, let $x=B y$ be the vector whose components are given by

$$
x_{i j}=\left\{\begin{array}{ccc}
\sum_{s \in \operatorname{Lv}\left(T_{i j}^{j}\right)} y_{i s} & \text { if } & i \notin S,(i, j) \in A  \tag{13}\\
0 & \text { if } & i \in S,(i, j) \in A
\end{array}\right.
$$

Then (c) holds by definition. Let us show that (a), (b), (d) also hold.

Let $i j \in E$. Then after (5) one has $\sum_{s \in S} y_{i s}=\sum_{s \in S} y_{j s}=$ 1 and $y_{i s} \leq y_{j s}$ for each $s \in \operatorname{Lv}\left(T_{i j}^{j}\right)$. Thus

$$
x_{i j}=\sum_{s \in \operatorname{Lv}\left(T_{i j}^{j}\right)} y_{i s} \leq \sum_{s \in \operatorname{Lv}\left(T_{i j}^{j}\right)} y_{j s}=1-x_{j i}
$$

So (a) are satisfied. Furthermore, for each $i \in U$,

$$
\sum_{(i, j) \in A} x_{i j}=\sum_{(i, j) \in A} \sum_{s \in \mathrm{Lv}\left(T_{i j}^{j}\right)} y_{i s}=\sum_{s \in S} y_{i s}
$$

so (b) also hold. Finally, for each $(h, k) \in A_{i s}$, one has $s \in$ $\operatorname{Lv}\left(T_{h k}^{k}\right)$, and thus

$$
y_{i s} \leq y_{h s} \leq \sum_{t \in \operatorname{Lv}\left(T_{h k}^{k}\right)} y_{h t}=x_{h k}
$$

Hence (d) hold as well.
In conclusion, the linear transformation (13) maps $P$ into $K$. Moreover, it maps integral points of $P$ into integral points of $K$. Hence the statement follows.

As in the case of (5), one may obtain a concise reformulation of (11) involving $O(n p)$ variables and $O(n p)$ constraints only. It is enough to replace the order constraints (11.d) by the order constraints:

$$
\begin{align*}
& y_{i s} \leq y_{j(i, s) s} \quad i \in U, s \in S, i s \notin E \\
& y_{i s} \leq x_{i j(i, s)} \tag{14}
\end{align*}
$$

where the index $j(i, s)$ has the same meaning as in (5). Let us denote by (ILP2) the resulting ILP, and its continuous relaxation by (LP2). The ILP's (11) and (ILP2) are "equivalent", in the sense that every feasible solution to (ILP2) is feasible also in (11) (since, by transitivity, constraints (14) imply constraints (11.d)), and in the opposite direction, in view of (12), every optimal solution to (11) is optimal also in (ILP2). The same kind of equivalence holds for the continuous relaxation
of (11) and (LP2). Needless to say, also (LP2) can be solved in $O\left((n p)^{6}\right)$ time by Tardos' algorithm.

Summing up, in order to solve (11) it is enough to solve its continuous relaxation by linear programming. It should be noticed that in every optimal solution to the continuous relaxation the first equality of (12) must still hold. It follows that (11)-or (ILP2)- can be equivalently reformulated as:

$$
\begin{array}{ll}
\max & \sum_{i \in U} \sum_{s \in S} c_{i s}^{\prime} \min _{(h, k) \in A_{i s}} x_{h k} \\
& x_{i j}+x_{j i} \leq 1
\end{array} \quad \forall i j \in E=1 . \quad \forall i \in U,
$$

We have obtained a linearly constrained piecewise-linear concave maximization problem having only $2 n-p-2$ variables, which can be effectively solved by nondifferentiable convex minimization methods [6, 9, 12]. Notice that the coefficient matrix of (15) is totally unimodular and that the extreme points (which are integral) are precisely the incidence vectors of the centered spanning antibranchings of $D$. Moreover, the objective function of (15) (restricted to the vertices of the unit hypercube) is a polynomial in boolean variables with nonnegative coefficients. Therefore, (15) can also be seen as a (strongly polynomial time solvable) special case of the problem of maximizing a supermodular function over the family of the common independent sets of two matroids, which is in general hard to solve and contains ordinary NP-complete instances.

Problem (15) can be solved in polynomial time by binary search over a threshold $z$ on the value of its objective function $f(x)$. At each iteration of the binary search one finds a point (if any) of a certain convex set $C(z)$ by the algorithm in Ref. [19]. Let $C_{\max }$ be the maximum absolute value of the $c_{i s}^{\prime}$ 's. The overall complexity of such procedure (see Ref. [1]) is $O\left(\left(n^{6} p^{2}+n^{5} p \log C_{\max }\right)\left(\log C_{\max }+\log (n p)\right)\right)$ which is better than the complexity of Tardos' algorithm when $p$ is not too small and $C_{\text {max }}$ is not too large.

The above nonlinear approach becomes particularly attractive when (NP-complete) capacitated versions of MCO are dealt with and one wants to find good bounds on the optimum.

In conclusion, we have shown the polynomial time solvability (although with relatively high complexity) of the equivalent problems MCP and MCO. In the next section, we shall derive an $O(n p)$ algorithm based on a combination of dynamic programming and maximum weighted closure procedures.

## 3. A LINEAR TIME COMBINATORIAL ALGORITHM

In this section we describe a $O(n p)$ polynomial algorithm for partitioning a tree into single-center subtrees so as to minimize (maximize, resp.) flat service costs (benefits, resp.).

Notice that such complexity is linear in the input size. In the following we shall adopt the minimization form of MCP.

Before describing the algorithm, we introduce some definitions and notation. For any undirected or directed graph $G$, we denote by $V(G)$ its vertex-set. Let $T=(V, E)$ be an arbitrary rooted tree. As usual, we denote by $(i, j)$ an edge directed from node $i$ to node $j$. Node $i$ is said to be the predecessor (or the father) of $j$ and node $j$ is a successor (or a child) of $i$. The set of predecessors and successors of node $i$ are denoted by $\operatorname{Pred}(i)$ and $\operatorname{Succ}(i)$, respectively. If there is a directed path from node $i$ to node $j$, then $i$ is called an ancestor of $j$, and $j$ a descendant of $i$. We regard $i$ to be both an ancestor and a descendant of itself. The set of ancestors and descendants of node $i$ are denoted by $\operatorname{Anc}(i)$ and $\operatorname{Desc}(i)$, respectively. For a given node $i$ the closure of $i$ is any subset of nodes such that, whenever it contains node $i$, it contains all the predecessors (and hence all the ancestors) of $i$. If $Z$ is any subset of nodes, the cocycle $\partial Z$ of $Z$ is the set of all edges with exactly one vertex not in $Z$. We call the downtree of $T$ at $v$, and denote it by $T_{v}$, the subtree of $T$ rooted at $v$ induced by $\operatorname{Desc}(v)$. Given an edge $(u, v)$, the partial downtree $T_{u v}$ is the subtree of $T$ induced by $u \cup \operatorname{Desc}(v)$. The subtree $T_{u v}$ is rooted at $u$ and it is obtained from $T_{v}$ by the addition of the edge $(u, v)$.

In the following we assume that $T$ is rooted at one of its units, say $r$. We are going to describe a polynomial algorithm for solving MCP which is based on a bottom-up dynamic programming recursion. In order to implement such a recursion, a sequence of minimum weight closure problems on trees is solved.

Recall that for a given directed graph $(N, A)$ and any weight function $w$ on the vertices, a subset $X \subseteq N$ is a closure of $N$ if $i \in X$ implies $j \in X$ for any $j$ successor of $i$. As usual, the weight function $w$ is extended to subsets by additivity: $w(U) \equiv \sum_{h \in U} w_{h}, U \subseteq N$. The minimum weight closure problem (MWC) is to find a closure $\bar{X}$ of $N$ with minimum weight:

$$
w(\bar{X})=\min _{X \text { closure of } N} \sum_{i \in X} w_{i} .
$$

Let $i$ be a unit, $T_{i}$ the downtree of $T$ rooted at $i$, and $l$ an arbitrary center (leaf) of $T_{i}$. We define $z_{i l}$ to be the minimum service cost of any centered partition of $T_{i}$, subject to the condition that $i$ is served by $l$; we shall also define

$$
\begin{equation*}
z_{i}=\min \left\{z_{i l} \mid l \text { is a center of } T_{i}\right\} \tag{16}
\end{equation*}
$$

Thus, $z_{i}$ is just the minimum service cost of any centered partition of $T_{i}$. Denote by $\Pi_{i s}$ the set of all centered partitions of $T_{i}$ such that $i$ is served by $s$. The dynamic programming algorithm starts from those nodes $g$ that are predecessors of leaves $l$. Clearly,

$$
z_{g l}=c_{g l}
$$

Now, consider an arbitrary unit $i$ and assume that, for each child $j$ of $i$ and each center $t$ in $T_{j}$, all values $z_{j t}$ have been previously computed by the dynamic programming algorithm. Assume also that the $z_{h}$ values are stored for each proper


FIG. 3. The white vertices are assigned to center $s$, while the vertices in grey are assigned to some center different from $s$.
descendant $h$ of node $i$. Let $s$ be any center in $T_{i}$. Then, in any cheapest centered partition of $T_{i}$ where $i$ is served by $s$, all nodes along the directed path $P_{i s}$, from $i$ to $s$, are also being served by $s$. Moreover, if $T_{v}$ is an arbitrary downtree whose root $v$ has its father $f$ in $P_{i s}$, some units of $T_{v}$ may also be served by $s$ (see Fig. 3). The set $R$ of all such units must induce a subtree of $T_{v}$ rooted at $v$. Notice that this subtree depends only on $T_{v}$ and $P_{f s}$, but is independent of other similar downtrees $T_{v^{\prime}}$ (where the father of $v^{\prime}$ lies in $P_{i s}$ ), no matter whether $v^{\prime}$ has the same father as $v$ or not. Indeed, the fact that a certain unit $h$ belongs to $R$ is influenced only by the nodes along the path from $h$ to $s$ and by the descendants of $h$ in $T_{v}$. Therefore, we can restrict our attention to a single such downtree $T_{v}$.

In order to compute the value $z_{i s}$ (and hence $z_{i}$ ) using the dynamic programming recursion, we need to find the above set $R$ efficiently. One should notice that this is in itself a problem of type MCP, where $T$ is replaced by the partial downtree $T_{f v}$ and $f$ becomes a center with service costs $c_{h f}=$ $c_{h s}$ for all the units $h$ of $T_{f v}$. To find $R$, consider the subtree $U_{v}$ induced by all units of $T_{v}$. For each node $h$ of $U_{v}$ consider the following three weights:

$$
\begin{gather*}
u_{h}=c_{h s},  \tag{17}\\
l_{h}=z_{h}-\sum_{m \in \operatorname{Succ}(h)} z_{m}  \tag{18}\\
w_{h}=u_{h}-l_{h} . \tag{19}
\end{gather*}
$$

We will refer to $u_{h}, l_{h}$, and $w_{h}$ as the upper weight, the lower weight, and the weight of $h$, respectively.

Remark 4. In view of the above formulas (17), (18), (19), the weight $w_{h}$ can be rewritten as

$$
\begin{equation*}
w_{h}=c_{h s}-\left(z_{h}-\sum_{m \in \operatorname{Succ}(h)} z_{m}\right) . \tag{20}
\end{equation*}
$$

The above expression has an interesting economic interpretation: at node $h$ two options are possible: (i) $h$ is served by
$s$, or (ii) $h$ is served by some leaf of the downtree $T_{h}$. Then the weight $w_{h}$ is actually the marginal cost of option (i) w.r.t. option (ii).

Lemma 2. For each node h of $U_{v}$, one has

$$
\begin{equation*}
z_{h}=\sum_{k \in \operatorname{Desc}(h)} l_{k} \tag{21}
\end{equation*}
$$

Proof. By induction on the depth of $h$. If $h$ is a leaf of $U_{v}$, then $\operatorname{Desc}(h)=\{h\}, \operatorname{Succ}(h)=\emptyset$ and both (18) and (21) amount to $l_{h}=z_{h}$. Assume that (21) holds for each node of depth $d-1$, and let $h$ have depth $d$. One has from (18)

$$
z_{h}=l_{h}+\sum_{m \in \operatorname{Succ}(h)} z_{m}
$$

and from the inductive hypothesis,

$$
z_{h}=l_{h}+\sum_{m \in \operatorname{Succ}(h)} \sum_{k \in \operatorname{Desc}(m)} l_{k}
$$

Theorem 5. Let $i$ be any node of $T$, $s$ a leaf of $T_{i}$ and $P_{i s}$ the directed path from i to $s$ in $T$. Let $v$ be a unit outside $P_{i s}$ whose fatherf belongs to $P_{i s}$. Finding the set $R$ of those units of $T_{v}$ that are served by $s$ in some cheapest partition of $\Pi_{i s}$ is reducible to the minimum weight closure problem in the rooted tree $U_{v}$ with node weights $w_{h}$.

Proof. First of all, notice that a subset of nodes of a tree rooted at $v$ is a closure if and only if it is either empty or it induces a subtree with root $v$. Let $\pi$ be a cheapest partition in $\Pi_{i s}$, and let $R$ be the set of units of $T_{v}$ that are served by $s$ in $\pi$. Let $\partial R$ be the cocycle of $R$ in $U_{v}$. Since $R$ is either empty or it induces a subtree of $T_{v}$ with the same root $v, R$ is a closure of $U_{v}$. Furthermore, the set $R$ must be chosen so as to minimize the overall contribution of $T_{v}$ to the service cost of $\pi$. Such contribution, in view of the decomposition followed in the algorithm, is equal to

$$
\begin{aligned}
\gamma(R)= & \sum_{h \in R} c_{h s}+\sum_{(p, q) \in \partial R} z_{q} \\
& \text { by Bellman's Optimality Principle } \\
= & \sum_{h \in R} u_{h}+\sum_{(p, q) \in \partial R} \sum_{k \in \operatorname{Desc}(q)} l_{k} \quad \text { by (21) } \\
= & \sum_{h \in R} u_{h}+\sum_{k \in V\left(U_{v}\right) \backslash R} l_{k} \quad \text { since } R \text { is a closure in } U_{v} \\
= & \sum_{h \in R} w_{h}+\sum_{k \in V\left(U_{v}\right)} l_{k} \quad \text { by (19) } \\
= & w(R)+z_{v} \quad \text { by }(21) .
\end{aligned}
$$

Conversely, if $R$ is a closure, then the above identities hold in reverse order. Therefore, $\gamma(R)$ and the weight of $R$ differ by a constant, and the statement follows.

On the above grounds, it follows from Bellman's Optimality Principle that each $z_{i s}$ satisfies the recursion:

$$
\begin{equation*}
z_{i s}=c_{i s}+z_{v^{\prime} s}+\sum_{v \in \operatorname{Succ}(i)-\left\{v^{\prime}\right\}} \gamma_{v s} \tag{22}
\end{equation*}
$$

where $v^{\prime}$ is the successor of $i$ along $P_{i s}$, and $\gamma_{v s}$ is equal to $z_{v}$ plus the minimum weight of a closure in $U_{v}$, with node weights $w_{h}$ given by (19). Notice that these node weights depend on the values $z_{h}$ for all units $h$ of $U_{v}$, which have already been computed before node $i$ is processed. In the above recursive formula we consider the following provisions:
if $i$ is a leaf of $U_{r}$ :

$$
\begin{equation*}
z_{i s}=c_{i s} \tag{23}
\end{equation*}
$$

if $i$ is not a leaf of $U_{r}$ but it is adjacent to $s$ in $T$ :

$$
\begin{equation*}
z_{i s}=c_{i s}+\sum_{v \in \operatorname{Succ}(i)} \gamma_{v s} . \tag{24}
\end{equation*}
$$

A pseudocode of the dynamic programming procedure is shown in Figure 4. The minimum weight closure problem on a tree can be solved by the linear time algorithm in Ref. [11]. The algorithm is based on the following property:

Property 1. If l is a leaf of the tree andf is its father then:

- if $w_{l} \geq 0$ then there exists an optimal closure $R$ such that $l \notin R$;
- if $w_{l}<0$ then in any optimal closure $R$, if $f \in R$ then $l \in R$.


## algorithm FIND-MCP

input: An instance ( $T, S, c$ ) of MCP.
output: A Minimum Cost Centered Partition of $T$.
begin
visit the tree $U_{r}$ bottom-up in reverse bfs order
for each visited unit $i$ do
for each unit $v \in \operatorname{Succ}(i)$ do
for each center $s$ in $T_{i}$ but not in $T_{v}$ do
for each unit $h$ of $U_{v}$
compute the weights $w_{h}$ according to (19); end for
compute a minimum weight closure $R_{v s}$ in $U_{v}$; $\gamma_{v s}:=w\left(R_{v s}\right)+z_{v} ;$ end for
end for
for each center $s$ in $T_{i}$ do
compute $z_{i s}$ according to (22), (23) or (24);
end for compute $z_{i}$ and $l_{i}$ according to (16) and (18) resp.; end for
recover an optimal partition of $T$ by a top-down visit of $U_{r}$; end

FIG. 4. A pseudocode of the dynamic programming procedure for finding a MCP.

In view of the above property one can either delete or contract edge $f l$, obtaining a tree of smaller size. The algorithm visits the tree bottom-up in reverse breadth first search order performing the two above reductions until a single node is obtained.

At the end of the dynamic programming procedure, an optimal partition can be recovered by the following procedure: start from the root $r$ and let $z_{r s}=z_{r}$; obtain the component $C_{s}$ as the union of all closures $R_{v s}$ such that $v$ is not in $P_{r s}$ but it is adjacent to a vertex in $P_{r s}$; delete $C_{s}$ from $T$ and restart the procedure from any tree of the resulting rooted forest.

### 3.1. Complexity Analysis

In FIND-MCP, the reverse breadth first search visit of the tree ensures that each time a formula is applied the values in the right hand side have been already computed. Thus, $w_{h}$ can be obtained in constant time, $\gamma_{v s}$ in $O\left(\left|R_{v s}\right|\right)$ time, $z_{i s}$ in $O(|\operatorname{Succ}(i)|)$ time, $z_{i}$ in $O(p)$ time and $l_{i}$ in $O(|\operatorname{Succ}(i)|)$ time. So, computing all $z_{i s}, z_{i}$ and $l_{i}$ requires $O(n p)$ time.

For each center $s$ and for each tree $U_{v}$ such that unit $v$ does not lie in $P_{r s}$ but it is adjacent to a vertex in $P_{r s}$, algorithm FIND-MCP computes the weights $w_{h}$, solves a minimum weight closure problem and computes the $\gamma_{v s}$ in time linear in the number of vertices of $U_{v}$ (see Fig. 3). Since, for each center $s$, the trees $U_{v}$ defined above are mutually disjoint, the time complexity for computing the weights $w_{h}$, solving the minimum weight closure problems and computing $\gamma_{v s}$ for all centers in $S$ is $O(n p)$.

The optimal partition can be obtained in $O(n)$ time by the recursive procedure described at the end of Section 3.

The resulting overall time complexity of algorithm FINDMCP is $O(n p)$.

We do not see how the above $O(n p)$ complexity could be improved. In fact, if an arbitrary cost is changed, the optimal solution may change. Hence, one has to look to all the $O(n p)$ costs in order to compute the optimal solution.

## 4. FURTHER COMPLEXITY RESULTS

The aim of the present section is to give more insights on the hardness of MCP on a general graph $G=(V, E)$. Any instance of MCP takes the form ( $G, S, c$ ). In Ref. [4] it has already been shown that bounding $|S|$ does not lead to easier solvable instances. Unfortunately, as shown below by Theorem 6.(a), enlarging the class of input graphs leading to polynomial time solvable classes of MCP, looks hopeless, even if strong conditions are imposed on $c$. Actually, in view of Theorem 6.(b), the main source of difficulty in solving MCP lies in the costs structure: requiring $c$ to be metric makes the problem easy to solve through a simple $O(n p)$ greedy algorithm. So both MCP on trees and MCP with metric costs barely lie within the boundary separating, so as to speak, easy instances from hard ones. This is confirmed by Theorem 6.(c)
where, even when $G$ is a tree and $c$ is metric, requiring some further conditions makes the problem NP-complete.

In order to proceed with the section we need some definitions. Let $w: E \rightarrow \mathbb{Z}_{+}$be a weighting of the edges of a graph $G$ and, for $u, v \in V$ denote by $d_{w}(u, v)$ the length of a shortest path from $u$ to $v$ with respect to $w$ and by $d(u, v)$ the geometric distance between $u$ and $v$ (the minimum number of edges of a path from $u$ to $v$ ). The service cost function $c$ is monotone, if $d(u, s)<d(v, s) \Rightarrow c_{u s} \leq c_{v s}$. The service cost function is said to be metric, if $c$ is proportional to $d_{w}$ for some $w$. Lower and upper capacity functions $b_{1}, b_{2}: S \rightarrow \mathbb{Z}$, with $b_{1} \leq b_{2}$ are also given. A capacitated centered partition (with respect to $b_{1}$ and $b_{2}$ ) is a centered partition $\pi=\left\{C_{1} \ldots, C_{p}\right\}$ such that for all $t=1, \ldots, p$

$$
b_{1}(t) \leq \sum_{v \in C_{t}} c_{v t} \leq b_{2}(t), \text { where } t \text { is the center in } C_{t}
$$

The following theorem summarizes the above mentioned results on the complexity of the problem.

## Theorem 6. Let $G$ be a connected graph, $S$ a set of p centers

 and $c$ a cost function. Then,(a) Problem MCP is NP-complete even if c is monotone and the input graph is bipartite.
(b) Problem MCP can be solved in strongly polynomial time if the input graph is arbitrary and $c$ is metric (compare with our main result for trees).
(c) It is NP-complete to decide if a 2 -spider admits a feasible capacitated partition. Therefore, the capacitated version of MCP is NP-hard even for trees and even for metric assignment functions.

Proof. (a). Reduction from SAT. Let $\mathbf{C}_{1}, \ldots, \mathbf{C}_{m}$ be $m$ clauses over the set of variables $\left\{u_{1}, \ldots, u_{n}\right\}$. Construct a bipartite graph as follows. For each clause $\mathbf{C}_{i}, i=1, \ldots, m$ there is a vertex $v_{i}$. For each variable $u_{j}, j=1, \ldots, n$ there is a vertex $z_{j}$. There is an edge joining $v_{i}$ to $z_{j}$ if and only if clause $\mathbf{C}_{i}$ contains variable $u_{j}$. The graph built so far is just the bipartite graph representing clause-variable incidence. For each vertex $z_{j}$ take two more vertices $s_{j}$ and $t_{j}$ and connect them to $z_{j} ; s_{j}$ represents literal $u_{j}$ while $t_{j}$ represents literal $\bar{u}_{j}$. The resulting graph $B$ is bipartite with shores $\left\{v_{1}, \ldots, v_{m}\right\} \cup\left\{s_{1}, t_{1}\right\} \ldots \cup\left\{s_{n}, t_{n}\right\}$ and $\left\{z_{1}, \ldots, z_{m}\right\}$. Moreover, we can suppose that such graph is connected (since the set of SAT instances whose corresponding bipartite graph is connected forms an NP-complete subclass of SAT). Let $S=\left\{s_{1}, t_{1}\right\} \cup \ldots \cup\left\{s_{n}, t_{n}\right\}$ and $U=V(B) \backslash S$. Define the assignment costs as follows:

- if variable $u_{j}$ occurs in clause $\mathbf{C}_{i}$ as $u_{j}$ set $c_{v_{i} s_{j}}=0$;
- if variable $u_{j}$ occurs in clause $\mathbf{C}_{i}$ as $\bar{u}_{j}$ set $c_{v_{i} t_{j}}=0$;
- set $c_{z_{j} s_{j}}=c_{z_{j} t_{j}}=0$, for $j=1, \ldots, n$;
- set the assignment costs equal to 1 otherwise.

The function $c$ is 0,1 -valued. Moreover, by construction, for $s \in S$ and $i \in U, c_{i s}=0 \Rightarrow d(i, s) \leq 2$ and $c_{i s}=1 \Rightarrow$
$d(i, s) \geq 2$. Hence for $s \in S$ and distinct $i, j \in U, c_{j s}>c_{i s}$ implies $d(j, s) \geq d(i, s)$. It follows that $c$ is monotone.

Next, we claim that there is a centered partition of cost zero if and only if the formula is satisfiable. First of all, observe that
(25) the cost of assigning $v_{i}$ to $q \in\left\{s_{j}, t_{j}\right\}$ is zero if and only if variable $u_{j}$ occurs in clause $\mathbf{C}_{i}$ either (case 1) as unnegated if $q=s_{j}$ or (case 2) as negated if $q=t_{j}$. Moreover, in both cases, $z_{j}$ and $v_{i}$ are adjacent in $B$ and $z_{j}$ lies on a path from $v_{i}$ to $q$.

By (25) it follows that any truth assignment that sets $u_{j}=1$ in the former case and $u_{j}=0$ in the latter one, satisfies clause $\mathbf{C}_{i}$. Therefore, any centered partition of cost zero defines a truth assignment satisfying all clauses. Conversely, suppose that there is a truth assignment satisfying all clauses. Assign $z_{j}$ to $s_{j}$ if variable $u_{j}$ is set to 1 by the truth assignment. Assign $z_{j}$ to $t_{j}$ otherwise. Let $q_{j} \in\left\{s_{j}, t_{j}\right\}$ denote the center which $z_{j}$ is assigned to. For $i=1, \ldots, m$, clause $\mathbf{C}_{i}$ contains a literal $x_{i}$ set to 1 by the truth assignment. Let $x_{i}$ belong to $\left\{\bar{u}_{j(i)}, u_{j(i)}\right\}$ where the index $j(i)$ is as small as possible. Assign $v_{i}$ to $q_{j(i)}$. By (25) such an assignment has cost zero. Moreover, $v_{i}$ and $z_{j(i)}$, which are adjacent in $B$, are both assigned to the same center $q_{j(i)}$. Therefore - up to trivial components $\left\{\left\{q_{j}, z_{j}\right\} \cup V_{j}, j=1, \ldots, n\right\}$, where $V_{j} \equiv\left\{v_{i}: j(i)=j\right\}$, is a centered partition of $B$ of cost zero.
(b). As in Section 2, assume that unit $i$ is served by center $s(i)$. Thus

$$
\begin{equation*}
\sum_{i \in U} c_{i s(i)}-\sum_{i \in U} \min _{t \in S} c_{i t}=\sum_{i \in U}\left(c_{i s(i)}-\min _{t \in S} c_{i t}\right) \geq 0 \tag{26}
\end{equation*}
$$

Let $\sigma$ denote the mapping that assigns to $i \in U$ the point $s$ of $S$ that minimizes $c_{i t}$ over $S$. Ties are broken supposing that $S$ is arbitrarily linearly ordered and by giving priority to smaller centers. If $c$ is metric the set $C_{s}=\{i \in U: \sigma(i)=s\}$ is connected in $G$ for each $s \in S$. Indeed, if $h \in C_{s}, s$ is the point of $S$ closest to $h$; if $i$ lies on a shortest path from $h$ to $s$ (shortest w.r.t. to the metric $c$ ), then $s$ is also the point of $S$ closest to $i$. Therefore, $i \in C_{s}$, and $\{s\} \cup C_{s}$ induces a connected component of $G$. By (26), it follows that $\left\{\{s\} \cup C_{s}, s \in S\right\}$ is a centered partition of minimum cost. One obtains a centered partition such that if $v \in U$ is assigned to $s \in S$ then all units along a shortest path from $v$ to $s$ are assigned to $s$ as well by the following simple $O(n p)$ greedy algorithm. Start with no unit being assigned. Let $v \in U$ be a unit not assigned yet and let $s \in S$ minimize $c_{v t}$ over $S$ (ties are broken with the same rule used above). Assign to $s$ all units along a shortest path form $v$ to $s$. Repeat until all units have been assigned.
(c). Reduction from SUBSET SUM [7]. Let $a_{1} \ldots a_{p}$ be an instance of SUBSET SUM. Let $M=\sum_{i=1}^{p} a_{i}$. Let $G$ be a star with $p+1$ leaves $v_{0}, v_{1}, \ldots, v_{p}$ and let $q$ denote the unique non-leaf node in $G$. Set $S=\left\{v_{0}, v_{1} \ldots, v_{p}\right\}$ and define $b_{1}, b_{2}: S \rightarrow \mathbb{Z}$ as follows: $b_{1}\left(v_{0}\right)=b_{2}\left(v_{0}\right)=M / 2$ and $b_{1}\left(v_{i}\right)=0, b_{2}\left(v_{i}\right)=1$, for $i=1, \ldots, p$. Insert a new vertex $u_{i}$ on each edge $q v_{i}, i=1 \ldots p$. Define edge weights
as follows: edges $v_{0} q$ and $u_{i} v_{i}, i=1 \ldots p$, have weight zero; edges $q u_{i}, i=1 \ldots p$ have weight $a_{i}$. Denote by $w$ this weight function and let $c_{v t}=d_{w}(v, t)$ be the length of a shortest path between $v$ and $t, v \notin S, t \in S$. Then every feasible capacitated centered partition defines a partition of $\{1, \ldots, p\}$ into sets $A$ and $B$ such that $\sum_{i \in A} a_{i}=\sum_{i \in B} a_{i}$ and conversely.

## Acknowledgments

We are indebted to both referees for their valuable suggestions.

## REFERENCES

[1] N. Apollonio, I. Lari, J. Puerto, F. Ricca, and B. Simeone, "Partitioning a tree into single-center subtrees to minimize flat service costs", Technical Report, Universidad de Sevilla Dep. to de Estadistíca e Investigación Operativa, (in preparation).
[2] Y. Bartal, On approximating arbitrary metrics by tree metrics, Proc of the 30th Ann Symp Foundation Comput Sci (1998), 161-168.
[3] M. Charikar, C. Chekuri, A. Goel, and S. Guha, Rounding via trees: Deterministic approximation algorithms for group Steiner trees and $k$-median, Proc 30th Ann Symp Foundation Comput Sci (1998), 114-123.
[4] R. Cordone, A short note on graph tree partition problems with assignment or communication objective functions, Internal Report DEI 2001.7, Politecnico di Milano, 2001.
[5] G. Cornuéjols, Combinatorial Optimization. Packing and Covering, SIAM, CBMS-NSF, Philadelphia, 2001.
[6] A. Frangioni, Generalized bundle methods, SIAM J Opt 13 (2002), 117-156.
[7] M.R. Garey and D.S. Johnson, Computers and intractability, W. H. Freeman, New York, 1999.
[8] R.S. Garfinkel and G.L. Nemhauser, Optimal political districting by implicit enumeration techniques, Manag Sci 16 (1970), 495-508.
[9] J.-L. Goffin and J.-P. Vial, Convex nondifferentiable optimization: a survey focussed on the analytic center cutting plane method, Opt Methods Software 17 (2002), 805-867.
[10] Grilli di Cortona, C. Manzi, A. Pennisi, F. Ricca, and B. Simeone, Evaluation and optimization of electoral systems, SIAM Monographs on Discrete Mathematics and Applications, SIAM, Society for Industrial and Aplied Mathematics, Philadelphia, 1999.
[11] P.L. Hammer and B. Simeone, Order relations of variables in $0-1$ programming, Surveys in combinatorial optimization, S . Martello, G. Laporte, M. Minoux, and C.C. Ribeiro (Editors), Vol. 31, Annals of Discrete Mathematics, 1987, pp. 83-112.
[12] J.B. Hiriart-Urruty and C. Lemaréchal, Convex analysis and minimization algorithms, Springer, Berlin, 1996.
[13] O. Kariv and S.L. Hakimi, An algorithmic approach to network location problems I: The $p$-centers, SIAM J Appl Math 37 (1979), 513-538.
[14] O. Kariv and S.L. Hakimi, An algorithmic approach to network location problems II: The p-medians, SIAM J Appl Math 37 (1979), 539-560.
[15] M. Maravalle and B. Simeone, A spanning tree heuristic for regional clustering, Commun Stat Theory Methods 24 (1995), 623-639.
[16] B. Simeone, Optimal graph partitioning, Atti Giornate di Lavoro AIRO, Urbino 1978, AIRO (Italian Association for Operations Research), Urbino, 1978, pp. 57-73.
[17] B.C. Tansel, R.L. Francis, and T.J. Lowe, Duality: Covering and constraining $p$-center problems on trees, Discrete location theory, P. Mirchandani and R.L. Francis (Editors), Wiley, New York, 1990, pp. 349-386.
[18] E. Tardos, A strongly polynomial algorithm to solve combinatorial linear programs, Oper Res 34 (1986), 250-256.
[19] P.M. Vaidya, A new algorithm for minimizing convex functions over convex sets, Math Progr 73 (1996), 291-341.


[^0]:    Received June 2005; accepted September 2006
    Correspondence to: N. Apollonio; e-mail: nicola.apollonio@uniroma1.it. *In memory of Stefano Pallottino
    Contract grant sponsor: Spanish Research; Contract grant numbers: MTM 2004:0909, HA2003:0121
    DOI 10.1002/net. 20197
    Published online in Wiley InterScience (www.interscience.wiley. com).
    (C) 2007 Wiley Periodicals, Inc.

