SIAM J. ALG. DISC. METH.
Vol. 4, No. 3, September 1983

A CLASS OF BALANCED MATRICES ARISING FROM LOCATION PROBLEMS*

ARIE TAMIR†

Abstract. A (0, 1)-matrix is balanced if it contains no square submatrix of odd order whose row and column sums are all two. Given two collections, $S=\{T_1,\cdots,T_m\}$ and $Q=\{T'_1,\cdots,T'_n\}$, of neighborhood subtrees of a tree T, let $A(S,Q)=(a_{ij})$ be the incidence matrix with $a_{ij}=1$ if and only if T_i intersects T'_j. It is shown that $A(S,Q)$ is balanced. This balancedness is then used to exhibit the existence of a polynomial algorithm to certain location problems.

1. Introduction. In his paper [1], Berge defined a (0, 1)-matrix to be balanced if it contains no square submatrix of odd order whose row and column sums are all two. He then characterized a balanced matrix in terms of the existence of integral solutions to certain linear programs whose constraints are defined by a balanced matrix. Berge's results were then refined and extended by Lovasz [8] and Fulkerson, Hoffman and Oppenheim [5].

In this work a special class of balanced matrices is presented. This class arises from location problems on tree networks.

Assume that an undirected tree $T=(N,E)$, with N and E denoting the sets of nodes and edges respectively, is embedded in the Euclidean plane, so that the edges are line segments whose endpoints are the nodes and the edges intersect one another only at nodes. Moreover, each edge of T has a positive Euclidean length. This embedding enables us to talk about points, not necessarily nodes, on the edges. For any two points x, y on T let $d(x,y)$ denote the distance between x and y, measured along the edges of T. $P(x,y)$ will denote the set of points on the simple path connecting x and y. T will also be used to denote the (infinite) set of points on T.

A subtree of T is a connected subset of the set T. A subtree, T_i, is called a *neighborhood* subtree if there exist a point $x_i \in T$ and $r_i \geqq 0$ such that $T_i=\{x \mid x\in T,\ d(x_i,x)\leqq r_i\}$. x_i is called the center of T_i.

Let $S=\{T_1,\cdots,T_k\}$ be a finite collection of subtrees of T, and define the intersection graph, $G(S)$, as follows. $G(S)$ has k nodes, corresponding to the k subtrees in S. Two nodes of $G(S)$ are connected by an edge if and only if the respective subtrees intersect. Defining a clique to be a maximal complete subgraph, let $A(S)$ be the node clique incidence matrix of $G(S)$, where nodes correspond to rows and cliques appear as the columns. The graph $G(S)$ has been shown in [2] to be chordal, i.e., for any circuit of order at least four there exists an edge, not of the circuit, which connects two nodes of the circuit. It is also proved in [2] that any chordal graph is realizable as the intersection graph of subtrees of a tree. Furthermore $G(S)$ is known to be perfect and its respective matrix $A^T(S)$ is therefore perfect, [10]. Perfectness is weaker than balancedness. In fact, even matrices $A^T(S)$ arising from chordal graphs $G(S)$ are not necessarily balanced. This is illustrated by the following chordal graph.

Example 1. Let T be as in Fig. (1a) and define the collection of subtrees $S=\{T_1,T_2,T_3,T_4,T_5,T_6\}$ as follows. $T_1=P(v_1,v_2)$, $T_2=P(v_2,V_3)$, $T_3=P(v_3,v_1)$,

* Received by the editors July 11, 1980, and in final revised form August 2, 1982.
† Department of Statistics, Tel Aviv University, Ramat Aviv, Israel.

$T_4=\{v_2\}$, $T_5=\{v_3\}$ and $T_6=\{v_1\}$. The respective intersection graph, $G(S)$, is given in Fig. 1(b), and

$$A^T(S)=\begin{bmatrix}1&1&1&0&0&0\\1&1&0&1&0&0\\0&1&1&0&1&0\\1&0&1&0&0&1\end{bmatrix}.$$

Considering the submatrix of $A^T(S)$ defined by the first three columns and the last three rows we observe that $A^T(S)$ is not balanced. (We note in passing that $G(S)$ is also realizable as the intersection of neighborhoods in R^2.)

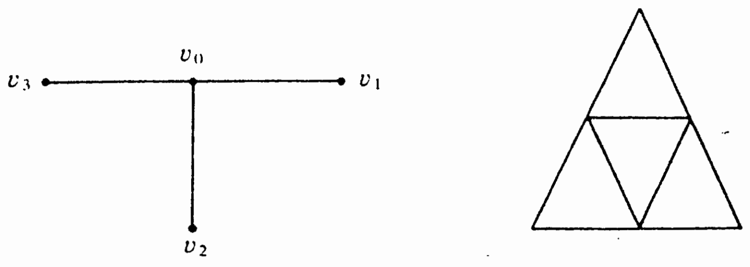

FIG. 1

In this paper we focus on collections of neighborhood subtrees and show that they, unlike collections of arbitrary subtrees do give rise to balanced matrices.

Let $S=\{T_1,\cdots,T_m\}$ and $Q=\{T'_1,\cdots,T'_n\}$ be two sets of neighborhood subtrees in T. Define the $m\times n$ incidence matrix $A(S,Q)=(a_{ij})$ by $a_{ij}=1$ if and only if the intersection $T_i\cap T'_j$ is nonempty and $a_{ij}=0$, otherwise. We will prove that $A(S,Q)$ is balanced. (In particular, $A(S)$ is balanced when S consists of neighborhood subtrees.) This result is then applied to exhibit the existence of polynomial algorithms for certain location models.

2. Balancedness and intersection graphs.

LEMMA 1. *Let $\{x_1,\cdots,x_k\}$, $k\geqq 3$, be a set of distinct points on T. Define $x_{k+1}=x_1$. There exist indices $1\leqq i_1<i_2<i_3\leqq k$ such that the paths $P(x_{i_1},x_{i_1+1})$, $P(x_{i_2},x_{i_2+1})$ and $P(x_{i_3},x_{i_3+1})$ of the tree T intersect at some point $y\in T$.*

Proof. We define the indices i_1, i_2, i_3 and the point $y\in T$ as follows. First let $x_{i_1}=x_1$ and $x_{i_2}=x_2$. Now y is chosen to be the closest point to x_3 on the path $P(x_1,x_2)$. It remains to define i_3. If $k=3$ set $i_3=3$ and the result clearly holds. Thus let $k>3$. Define

$$j=\begin{cases}k-1 & \text{if } y\notin P(x_i,x_3) \text{ for all } 3<i\leqq k-1,\\ \min\{i\mid 3\leqq i<k-1,\ y\in P(x_{i+1},x_3)\} & \text{otherwise.}\end{cases}$$

Suppose first that $j=k-1$. If $y\in P(x_k,x_3)$ then $y\in P(x_{k-1},x_k)$, since $y\notin P(x_{k-1},x_3)$. Set $i_3=k-1$ and then the paths $P(x_1,x_2)$, $P(x_2,x_3)$ and $P(x_{k-1},x_k)$ intersect at y. If $y\notin P(x_k,x_3)$ then $y\in P(x_k,x_1)$ since $y\in P(x_1,x_3)$. Set $i_3=k$ and the paths $P(x_1,x_2)$, $P(x_2,x_3)$ and $P(x_k,x_1)$ intersect at y.

Now suppose $j<k-1$. Set $i_3=j$. From the definition of j it follows that the paths $P(x_1,x_2)$, $P(x_2,x_3)$ and $P(x_j,x_{j+1})$ intersect at y. This completes the proof.

We are now ready to present the main result.

THEOREM 1. *Let $S=\{T_1,\cdots,T_m\}$ and $Q=\{T'_1,\cdots,T'_n\}$ be two sets of neighborhood subtrees in T. Let $A(S,Q)=(a_{ij})$ be the incidence matrix satisfying $a_{ij}=1$ if and* *only if $T_i \cap T'_j$ is nonempty. Then $A(S, Q)$ does not contain a square submatrix of size $k \geqq 3$ which has no identical columns, and its row and column sums equal to two.*

Proof. Let $T_i = \{x \mid d(x_i, x) \leqq r_i\}$, $i = 1, \cdots, m$, and $T'_j = \{x \mid d(y_j, x) \leqq s_j\}$, $j = 1, \cdots, n$. Then $a_{ij} = 1$ if and only if $d(x_i, y_j) \leqq r_i + s_j$. Suppose that $A(S, Q)$ contains a square submatrix $B = (b_{ij})$ of size $k \geqq 3$ which has no identical columns and its row and column sums are all equal to two. Without loss of generality suppose that this is the submatrix defined by the first k columns and k rows of $A(S, Q)$. Also, suppose that $b_{ij} = 1$ if and only if $i = j$, $j-1$ or $(i, j) = (1, k)$. First we note that $x_i \neq x_j$ for $1 \leqq i \neq j \leqq k$. Since, otherwise, we would have $T_i \subseteq T_j$ or $T_i \supseteq T_j$ which contradicts the fact that no row vector of B is greater than or equal to another row vector of B. Considering $\{x_1, \cdots, x_k\}$, let (x_{i_j}, x_{i_j+1}), $j = 1, 2, 3$, be the three pairs obtained from the previous lemma, and let y be the point on the path connecting x_{i_j} to x_{i_j+1}, $j = 1, 2, 3$.

The matrix B expresses the intersection relations between $\{T_i\}_{i=1}^k$ and $\{T'_i\}_{i=1}^k$. Each column of B contains exactly two 1's. Furthermore, these two 1's appear consecutively (mod k), and one of them is a diagonal element. Therefore, for $j = 1, 2, 3$ there exist T'_{i_j} intersecting T_{i_j} and T_{i_j+1} (exclusively). Without loss of generality suppose that

$$s_{i_1} - d(y, y_{i_1}) \leqq s_{i_2} - d(y, y_{i_2}) \leqq s_{i_3} - d(y, y_{i_3}).$$

Since $y \in P(x_{i_1}, x_{i_1+1})$ it follows that $y \in P(z, y_{i_1})$ where z is either x_{i_1} or x_{i_1+1}. Let $z = x_{i_1}$ then

$$\begin{aligned} 0 \leqq r_{i_1} + s_{i_1} - d(x_{i_1}, y_{i_1}) &= r_{i_1} + s_{i_1} - d(x_{i_1}, y) - d(y, y_{i_1}) \\ &\leqq r_{i_1} + s_{i_j} - d(x_{i_1}, y) - d(y, y_{i_j}) \leqq r_{i_1} + s_{i_j} - d(x_{i_1}, y_{i_j}), \qquad j = 1, 2, 3. \end{aligned}$$

Hence, we obtained the contradiction that the neighborhood subtree T_{i_1} intersects the three neighborhood subtrees T'_{i_j}, $j = 1, 2, 3$. This contradicts the fact that row i_1 of B contains exactly two 1's. A similar contradiction is obtained if we take $z = x_{i_1+1}$. Therefore the proof is now complete.

We note that since a point on T is a neighborhood subtree, Q for example may be a collection of points on T. Indeed, this is the special case arising from the location model considered in the next section.

COROLLARY 1. *$A(S, Q)$ defined as in Theorem 1 is balanced. In particular, the node clique incidence matrix of the intersection graph corresponding to the collection of neighborhood subtrees S is balanced.*

Proof. The first part is obvious from Theorem 1. Let $A(S)$ be the node clique incidence matrix of the intersection graph $G(S)$. It is shown in [3] that all the subtrees corresponding to a clique of $G(S)$ have a point in T contained in all of them. (The maximality of the clique ensures that this point is contained in no other subtree of the collection.) Thus there exists a set of points Y in T such that $A(S) = A(S, Y)$, and the result follows from Theorem 1.

Theorem 1 and Example 1 present one property which is satisfied by intersection graphs realizable by collections of neighborhood subtrees but not by chordal graphs which are known, [2], to be realized by collections of subtrees. Next, we demonstrate another property which is met by our class of balanced matrices but not by matrices arising from general chordal graphs.

Given that $A(S, Q)$ is balanced, it then follows from [5] that all the extreme points of the polyhedron $\{z \mid A(S, Q)z \geqq e, z \geqq 0\}$ are integral, (e is the vector of all 1's). As noted in the introduction, the node clique incidence matrix of a general chordal graph, which is not an intersection graph of neighborhood subtrees, is not necessarily balanced. Hence the results of [5] do not induce the above integrality property of the respective polyhedron defined by a general chordal graph. Indeed, the above integrality property, which is weaker than balancedness, is not shared by a general chordal graph.

Example 2. *Let G be* the chordal graph in Fig. 2. Let A be the node clique incidence matrix of G (with nodes corresponding to rows).

$$A=\begin{bmatrix}1&0&0&0&1&0&0\\0&1&0&0&0&1&0\\0&0&1&0&0&0&1\\1&1&0&1&0&0&0\\1&0&1&1&0&0&0\\0&1&1&1&0&0&0\\0&0&0&0&1&0&0\\0&0&0&0&0&1&0\\0&0&0&0&0&0&1\end{bmatrix}.$$

The polyhedron $\{x|Ax \geqq e, x \geqq 0\}$ possesses the nonintegral extreme point, $x = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 0, 1, 1, 1)$.

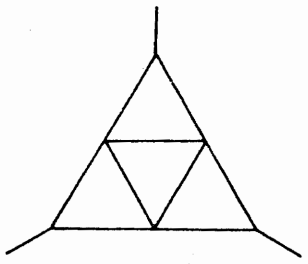

FIG. 2

Chordal graphs satisfy the following weaker integrality property.

THEOREM 2. *Let G be a chordal graph and let $A = (a_{ij})$ be its node clique incidence matrix with nodes corresponding to rows. If the equality constrained set covering polyhedron, $\{x|Ax = e, x \geqq 0\}$, is nonempty, it is a singleton consisting of a* 0–1 *vector.*

Proof. In fact we prove that for any integer vector f the system $Ax = f$ has at most one solution. Furthermore, if it exists, this solution is integer.

The proof is by induction on the number of nodes in G. The result is trivial for a graph consisting of one or two nodes.

Now, let G be a chordal graph. Using the induction hypothesis we may assume that G is connected. Then G contains a simplicial node, [2], i.e., a node, say i, that belongs to exactly one clique. Therefore, i is associated with a unit row of A, and if $a_{ij} = 1$ we must set $x_j = f_i$ if $Ax = f$. We can then eliminate the ith equation of the system $Ax = f$. Let $N(i)$ be the set of neighbors of i in G. $N(i)$ is nonempty since G is connected.

Let G' be the induced subgraph obtained by omitting node i and all edges connecting it to members in $N(i)$. The node clique incidence matrix of G', A', is a submatrix of A, defined as follows. If the complete subgraph induced by the nodes in $N(i)$ is maximal in G', then A' is the submatrix obtained by deleting the ith row of A. Otherwise, A' is obtained by deleting the ith row and the jth column of A.

Suppose first that A' is obtained by deleting only the ith row of A. Since G' is an induced subgraph it is chordal. By the induction hypothesis, the system $(Ax)_k = f_k$,

$\forall k \neq i$, is either inconsistent or else it has a unique solution, x', which is also integer. Thus the original system $Ax = f$ is consistent if and only if x' exists and $x'_j = f_i$. The uniqueness of x' as a solution to the subsystem implies its uniqueness with respect to the system $Ax = f$.

Next, suppose A' has one less column than A. The system $Ax = f$ may be written as $(Ax)_k = f_k$, $\forall k \neq i$, $x_j = f_i$. Substituting f_i for x_j in each equation $(Ax)_k = f_k$, $\forall k \neq i$, we obtain exactly the subsystem corresponding to A'. Now we use the chordality of G', and apply the induction hypothesis on the subsystem to conclude the validity of the result for the system $Ax = f$.

3. The location model. Given the tree T defined in the introduction, suppose that two finite subsets of T, Σ and Δ are specified. $\Sigma = \{y_1, \cdots, y_n\}$ is called the supply set and $\Delta = \{x_1, \cdots, x_m\}$ is the demand set. The demand points are to be served by centers which can be located only at points of Σ. Each demand point, x_i, must have at least a_i centers established at a distance not greater than $r_i \geqq 0$ from it. Due to capacity constraints at most b_j centers can be located at y_j. The cost of establishing any center at y_j is $v_j \geqq 0$. The problem is to find the minimum budget required for setting centers meeting the demand constraints.

We note that if T is replaced by a general (planar) network even a special case of the above model is known to be NP-hard, [6]. Turning back to a tree network, the demand constraints imply that for each x_i, $i = 1, \cdots, m$, at least a_i centers should be set at the neighborhood subtree $T_i = \{x \mid d(x, x_i) \leqq r_i\}$. Defining $S = \{T_1, \cdots, T_m\}$, the location problem is formulated as

$$\text{Minimize } \sum_{j=1}^{n} v_j z_j$$

$$\text{(1)} \qquad \text{s.t.} \quad Az \geqq a,$$

$$b \geqq z \geqq 0 \text{ and integer,}$$

where $A = A(S, \Sigma)$, $a = (a_1, \cdots, a_m)$, $b = (b_1, \cdots, b_n)$ and $e = (1, \cdots, 1)$.

Certain instances of (1) have been considered in the literature. The special case of equal setting costs, v_j, for the centers and $a_i = 1$, $i = 1, \cdots, m$, $b_j = \infty$, $j = 1, \cdots, n$, can be solved in linear time by a modified version of the algorithm in [6]. A generalization of the latter special case, allowing arbitrary integer values for a_i is solved in [3]. There, the problem is reduced to finding a minimum cover of the nodes of $G(S)$, $S = \{T_1, \cdots, T_m\}$, by cliques, and observing the chordality of $G(S)$. The cliques are induced by the supply points. Applying the perfectness of $G(S)$ a dispersion location problem which is dual to this special case is also defined in [3]. Using only the perfectness property of A^T, the case considered in [3] was maximal in the sense that perfectness is equivalent to the existence of an integer solution to the linear program $\min \{\sum_{j=1}^{n} z_j \mid Az \geqq a, z \geqq 0\}$ for all nonnegative integers a, [4], [10].

The results of the previous sections, where the balancedness of $A = A(S, \Sigma)$ is proved, enable us to extend the class of "solvable" cases of (1).

We start with the special case of (1), where all the setting costs, v_j, are equal. This case is called the multiple coverage problem. Using [1] we note that the balancedness of A is equivalent to the existence of an integer solution to the linear program $\min \{\sum_{j=1}^{n} z_j \mid Az \geqq a, b \geqq z \geqq 0\}$, for all nonnegative integer vectors a, b. Thus, the multiple coverage problem can be solved polynomially using Khachian's algorithm, [7], for linear programs. Also, we have constructed a direct algorithm for the multiple coverage model. Since this algorithm is based on simple extensions of the main ideas embedded in the algorithms of [3], [6], we skip the description of our procedure. (The interested reader can obtain the detailed scheme from the author.) We mention that if, for example, the supply and demand sets consist only of nodes of the tree T, then the complexity of our direct algorithm is $O(n^2)$, where n is the number of nodes of T.

Secondly we consider the special case of (1) where $a_i = 1$, $i = 1, \cdots, m$. (The constraints $z \leqq b$ can be assumed to be redundant in this case.) The results in [5] ensure that all the extreme points of $\{z | Az \geqslant e, z \geqq 0\}$ are integral. Thus, again the problem can be solved polynomially using Khachian's algorithm, provided the v_j are rational. (Khachian's algorithm may find an optimal solution which is not extreme and therefore may not be integer. However, an optimal extreme point to a linear program can always be generated in polynomial time if some optimal solution is available.) In the next section we will present a direct algorithm for solving this case.

We now summarize the results on the location model (1). To our knowledge no efficient algorithms to solve (1) are available. Verifying whether this problem is polynomially solvable will require a different approach than the one presented above for the special cases. This is due to the fact that the integer solution to (1) may not be optimal to the relaxed linear program. This is illustrated by the following.

Example 3. Let T be given by Fig. 3. Suppose that $\Sigma = \Delta = \{x_1, x_2, x_3, x_4\}$ with $d(x_i, x_4) = 1$, $i = 1, 2, 3$. Also let $r_i = 1$, $i = 1, 2, 3, 4$. Finally set $b = e$, $a_i = v_i = 1$, $i = 1, 2, 3$, and $a_4 = v_4 = 2$. We then have that the solution to (1) is 3 while the optimal objective of the relaxed linear program is 2.5.

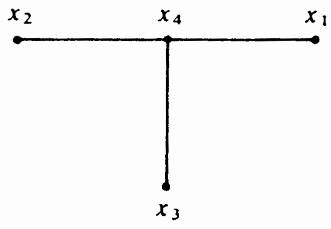

FIG. 3

Combining the results of the previous section with those of [1], [5], [7] our work shows the existence of efficient algorithms when either $a = e$, or the setting costs, v_j, are equal.

Finally we note another solvable case of (1), which is not implied by the above. If the matrix A is totally unimodular the model can now be solved efficiently by [7], if all data are rational. Total unimodularity is achieved, for example, by a tree which is a simple path. In this case the resulting graph is an interval graph.

4. Solving the location problem.

In this section we present a direct algorithm for solving the location problem (1) described in the previous section, with $a_i = 1$, $i = 1, \cdots, m$. To simplify the presentation we consider here the following special case. Given the tree $T = (N, E)$ with N and E the sets of nodes and edges respectively, suppose that $\Sigma = \Delta = N$, i.e., demand and supply occur at the nodes only. Given $r_i \geqq 0$, $i \in N$ we wish to minimize the budget for setting centers such that each demand point is covered by a center, i.e., each $i \in N$ is at a distance of at most r_i from some center.

To present the algorithm we first assume that the tree is rooted at some distinguished node, say v. For each node $j \in N$ define $B(j)$ as the set of descendents of j, i.e., the entire set of nodes having j on the path connecting them with v. In particular $j \in B(j)$. Also define $S(j)$ to be the set of "sons" of j, i.e., the nodes having j as the immediate predecessor on the path connecting them with v. $T(j)$ will denote the minimal subtree containing $B(j)$.

Let $j \in N$. Suppose that a center already exists at some node in $N-B(j)$ whose distance from j is t. (If more than one center exists in $N-B(j)$ consider only the closest to j.) This center may clearly cover some nodes of $B(j)$. Suppose, further, that no centers exist in $B(j)$. Now define $h(j,t,s)$ to be the minimum budget required to cover the nodes of $T(j)$, given that new centers are set at $B(j)$ only, with the closest being at a distance s from j, and the closest existing center in $N-B(j)$ is at a distance t from j.

Let $D(j)(F(j))$ be the set of distances from j to the members in $B(j)(N-B(j))$. Also the value $s=\infty(t=\infty)$ indicates that no center is set at $B(j)(N-B(j))$. Define $\bar{D}(j)=D(j)\cup\{\infty\}$ and $\bar{F}(j)=F(j)\cup\{\infty\}$. Then $h(j,t,s)$ is defined only for $s\in\bar{D}(j)$ and $t\in\bar{F}(j)$. Furthermore, we compute $h(j,t,s)$ only for $t\leqq s$ since $h(j,t,s)=h(j,s_j^*,s)$ for all $t\geqq s$ in $\bar{F}(j)$. (s_j^* is the smallest element in $\bar{F}(j)$ which is not smaller than s.)

Defining $H(j,t,s)=\min_{p\geqq s} h(j,t,p)$, we obtain $H(j,t,s)=H(j,t,\lceil s\rceil_j)$, where $\lceil s\rceil_j$ is the smallest element in $\bar{D}(j)$ which is not smaller than s. The answer to the location problem is given by $H(v,\infty,0)$. Our algorithm is based on a recursive computation of $h(j,t,s)$ leading to $H(v,\infty,0)$.

Starting with the tips of the rooted tree we obtain the following recursion for $t\in\bar{F}(j)$, $s\in\bar{D}(j)$ and $t\leqq s$.

If j is a tip, then $h(j,t,0)=v_j$ and

$$h(j,t,\infty)=\begin{cases}0 & \text{if } t\leqq r_j,\\ \infty & \text{if } t>r_j.\end{cases}$$

Suppose j is not a tip; then

$$h(j,t,\infty)=\begin{cases}0 & \text{if } d(i,j)+t\leqq r_i \text{ for all } i\in B(j),\\ \infty & \text{otherwise,}\end{cases}$$

$$h(j,t,0)=v_j+\sum_{i\in S(j)} H(i,d(i,j),0),$$

and for $0\neq s\in D(j)$

$$h(j,t,s)=\begin{cases}\infty & \text{if } t>r_j,\\ \displaystyle\min_{\substack{i\in S(j)\text{ and}\\ s-d(i,j)\in D(i)}}\Big\{h(i,t+d(i,j),s-d(i,j))+\sum_{\substack{k\in S(j)\\ k\neq i}} H(k,t+d(k,j),s-d(k,j))\Big\} & \text{when } t\leqq r_j.\end{cases}$$

Simplifying the expression for $t\leqq r_j$ we obtain

$$h(j,t,s)=\sum_{k\in S(j)} H(k,t+d(k,j),\lceil s-d(k,j)\rceil_k)$$
$$+\min_{\substack{i\in S(j)\text{ and}\\ s-d(i,j)\in D(i)}}\{h(i,t+d(i,j),s-d(i,j))-H(i,t+d(i,j),s-d(i,j))\}.$$

Having established the recursive relations leading to the optimal solution we next demonstrate that the complexity of the suggested algorithm is $O(n^3)$ when n is the number of nodes of T.

In the initial phase we generate and sort each one of the sets $D(j)$, $F(j)$, $j\in N$. This will consume $O(n^2\log n)$ time.

Now, given j we show that the total effort needed to compute $h(j,t,s)$ for all $t\in\bar{F}(j)$ and $s\in\bar{D}(j)$, $t\leqq s$, is $O(n^2|S(j)|)$.

First, for each $k \in S(j)$ compute $\lceil s-d(k,j)\rceil_k$, for all $s \in D(j)$. This will enable us to use previously computed values of the functions $h(k,\cdot,\cdot)$ and $H(k,\cdot,\cdot)$ for $k \in S(j)$. Since $D(j)$ and $D(k)$ are already sorted this step is done in $O(n)$ time for each $k \in S(j)$, or in $O(n|S(j)|)$ for all $k \in S(j)$.

Next, for each $s \in D(j)$ the set of indices with $s-d(i,j) \in D(i)$ is found. Like the preceding step this is performed in $O(n|S(j)|)$ time for all $s \in D(j)$.

Finally we turn to a given pair (t, s) with $t \in F(j)$, $s \in D(j)$. Using the recursive relations and the information acquired in the previous steps $h(j,t,s)$ is computed in $O(|S(j)|)$ time. Thus the effort for computing $h(j,t,s)$ and $H(j,t,s)$ for all pairs (t,s), $t \in F(j)$, $s \in D(j)$ is $O(n^2|S(j)|)$, and the bound for the entire algorithm becomes $O(n^3)$.

It is easily verified that this bound is not affected if one also wishes to find the optimal locations of the centers yielding the minimum budget. The space required for implementing the algorithm is also $O(n^3)$.

We have provided an efficient procedure to solve the location problem where it is required to minimize the budget for covering each demand point. This procedure can now be used to solve the following related problem.

Suppose that the total budget available for setting centers at the supply points is $B>0$. Given this constraint one wishes to establish centers such that the maximum distance from a demand point to its nearest center is minimized.

It is clear that the minimum of the maximum distance is an element in the set

$$R=\{d(x_i, y_j) \mid x_i \in \Delta,\ y_j \in \Sigma\}.$$

Hence the optimal value is the minimum element $r \in R$ such that the minimum budget, needed to ensure that each demand point is covered within a radius r does not exceed B. The procedure described above will be used to determine for any given r whether the respective budget exceeds B. To find the optimal value we can use the sophisticated search on the set R which is used in [9] to find an optimal element in the case where the setting costs, v_j are equal.

Note added in proof. We note that a special case of Theorem 1 is proved in R. Giles, *A balanced hypergraph defined by certain subtrees of a tree*, Ars Combinatoria, 6 (1978), pp. 179–183.

REFERENCES

[1] C. Berge, *Balanced matrices*, Math. Programming, 2 (1972), pp. 19–31.

[2] P. Buneman, *A characterization of rigid circuit graphs*, Discrete Math., 9 (1974), pp. 205–212.

[3] R. Chandrasekaran and A. Tamir, *Polynomially bounded algorithms for locating p centers on a tree*, Math. Programming, 22 (1982), pp. 304–315.

[4] D. R. Fulkerson, *Blocking and anti-blocking pairs of polyhedra*, Math. Programming, 1 (1971), pp. 168–194.

[5] D. R. Fulkerson, A. J. Hoffman and R. Oppenheim, *On balanced matrices*, Math. Programming Study, 1 (1974), pp. 120–132.

[6] O. Kariv and S. L. Hakimi, *An algorithmic approach to network location problems. Part 1: The p-centers*, SIAM J. Appl. Math., 37 (1979), pp. 513–538.

[7] L. G. Khachian, *A polynomial algorithm in linear programming*, Dokl. Akad. Nauk USSR, 244, 5, Feb. (1979).

[8] L. Lovasz, *Normal hypergraphs and the perfect graph conjecture*, Discrete Math., 2 (1972), pp. 253–267.

[9] N. Megiddo, A. Tamir, E. Zemel and R. Chandrasekaran, *An $O(n \log^2 n)$ algorithm for the kth longest path in a tree with applications to location problems*, SIAM J. Comput., 10 (1981), pp. 328–337.

[10] M. W. Padberg, *Perfect zero-one matrices*, Math. Programming, 6 (1974), pp. 180–196.