MATHEMATICS OF OPERATIONS RESEARCH
Vol. 3, No. 3, August 1978
Printed in U.S.A.

COMPUTATIONAL COMPLEXITY OF THE GAME THEORY APPROACH TO COST ALLOCATION FOR A TREE*

NIMROD MEGIDDO

Tel Aviv University

In the game theory approach to the problem of allocating cost, the users of a facility are viewed as players in a cooperative n-person game. It is the nature of cooperative games that the power of each one of the 2^{n-1} possible coalitions is taken into account. Thus, practical problems of allocating cost may be intractable in the game theory approach because of the computational complexity involved. However, it is shown that good algorithms do exist for the nucleolus and the Shapley value allocations for a tree. The nucleolus can be computed within $O(n^3)$ operations and the Shapley allocation in $O(n)$ operations.

Introduction. One application of the theory of cooperative games in characteristic function form (see [12]) is to cost allocation problems. An abstract cost allocation problem could be described as follows. Let $N = \{1, \ldots, n\}$ $(n \geq 2)$ be the set of *users*. The users wish to establish a facility that will provide them various services; in an equivalent version of the problem they might wish to cooperatively utilize an existing facility. Usually, not every user needs all the different types of service. Hence, for every $S \subset N$ we denote by $v(S)$ the minimum cost required for establishing a facility which provides all the types of service that the members of S might need. Alternatively, the value $v(S)$ could represent the cost of maintenance of that portion of an existing facility which is utilized by some member of S. If all the users act cooperatively then the total cost is represented by the number $v(N)$, and the question is how should this total cost be allocated to the users.

Naturally, the solution concepts suggested in the theory of cooperative games can be employed here, even though these solutions are usually formulated in terms of payoffs rather than costs. Littlechild and Thompson [9], Suzuki and Nakayama [15], Bird [1], and Claus and Granot [2] discussed cost allocation problems of our type and treated than in a game-theoretic approach. The main interest is in the nucleolus [13] and the Shapley value [14], since each of these solution concepts specifies a unique cost allocation. Formal definitions will be provided later.

There may, however, be one difficulty when the game-theoretic approach is put into practice. In general, for the computation of game-theoretic solutions we need to know the values of the characteristic function v. Thus, the time required for the preparation of the input for a standard computer program which calculates the nucleolus (see [6]), say, grows exponentially with the number of users. Furthermore, during the computation of the nucleolus, at least one linear programming problem of size not less than $2^n \times n$ has to be solved. There is no available algorithm for the general linear programming problem which runs in polynomial-time (see [4], [5]), and it is not even known whether such an algorithm exists at all. Thus, the state-of-art is that the computation time of the nucleolus cannot be upper-bounded by less than $O(2^{2^n})$. Suppose that in some class of cost allocation problems the number of parameters that define the problem grows polynomially with the number of users. For such a class of problems the game-theoretic approach would be definitely dissatisfactory—unless a

* Received November 9, 1976; revised October 13, 1977.
AMS 1970 subject classification. Primary 90D12, 90C35, Secondary 68A20, 05C05.
IAOR 1973 subject classification. Main: Games. Cross references: Computational analysis, graphs.
Key words. Cost allocation, nucleolus, Shapley value, polynomial-time algorithms, tree, depth-first-search.

procedure is found which brings to the solutions without first calculating all the values of the characteristic function. It seems reasonable to demand that, for such a class of problems, a solution should be found within time which grows not faster than polynomially in the number of users. Such polynomial-time procedures are often called "good" algorithm (following J. Edmonds). For the particular class of problems considered in this paper good algorithms are shown to exist.

We consider the following class of games. A directed tree graph T is given, whose set of nodes is $N \cup \{0\}$, where N is our set of players (previously called users) and node 0 is the root. The length of an arc (i, j) of T is denoted by $d(i, j)$ and is assumed to be positive. For every $S \subset N$ let $v(S)$ denote the total length of arcs that belong to some path from the root to a node $i \in S$. The resulting cooperative game $(N; v)$ corresponds to a class of cost allocation problems described as follows. Suppose that T is a tree of shortest paths from all intersections to a central station in a transportation network. Suppose further that the cost of maintenance of each road is known and the users which are located at the intersections wish to share the total cost of maintenance for the entire tree of roads.

In this paper we investigate the nucleolus and the Shapley value of games in the class defined above. We present properties of the nucleolus-allocation and the Shapley-allocation and develop good algorithms for computing them. We show that the nucleolus can be found in $O(n^3)$ time and the Shapley value in $O(n)$ time. Our results here generalize those of [7] and [8] since these papers deal, as a matter of fact, with the case of a tree which consists of a single path.

Preliminaries. For every $i \in N$ let $j(i)$ denote the "father" of i, i.e., that node which precedes i on the unique path from the root to i. A node j is said to be an *ancestor* of a node i if j belongs to that path from the root to i. A set $S \subset N$ is said to be *closed* if for every $i \in S, j \in S$ if j is an ancestor of i and $j \neq 0$. For an $S \subset N$ let S' denote the least closed set which contains S.

A set $B \subset S$ is said to be a *branch* of T if there exists a $j \in B$ such that $i \in B$ if and only if j is an ancestor of i. For every node $j \in N$ let $B(j)$ denote the branch of all the nodes of which j is an ancestor. Denote $d_i = d(j(i), i)$ and $d(B) = \sum_{i \in B} d_i$.

A node $i \in N$ is said to be a *neighbor* of a nonempty $S \subset N$ if $i \notin S$ and $j(i) \in S \cup \{0\}$. A nonempty $C \subset N$ is said to be *basic* if it is closed and has exactly one neighbor. Thus, a basic set is characterized by its unique neighbor, i.e., if j is the unique neighbor of a basic set C then $C = N \backslash B(j)$. For example, in the tree $\{(0, 1), (0, 2), (1, 3), (2, 4), (2, 5), (2, 6)\}$ $S = \{1, 4, 5\}$ is not closed, $S' = \{1, 2, 4, 5\}$ is the "closure" of S, S' has two neighbors, namely, node 6 and node 3. S' is not basic but $C_1 = \{1, 2, 3, 4, 6\}$ and $C_2 = \{1, 3\}$ are basic. $B = \{2, 4, 5, 6\}$ is a branch.

For every vector $y = (y_1, \ldots, y_n)$ and $S \subset N$ denote $y(S) = \sum_{i \in S} y_i$ $(y(\emptyset) = 0)$ and $e(S, y) = v(S) - y(S)$. The *core* (see [12]) is defined to be the set of all vectors y such that $y(N) = v(N)$ and $e(S, y) \geqslant 0$ for all $S \subset N$. Note that if y belongs to the core then $y_i = y(N) - y(N \backslash \{i\}) \geqslant v(N) - v(N \backslash \{i\}) \geqslant 0$ for every $i \in N$. The core in our class of games is never empty since, obviously, the allocation $(d_1, \ldots, d_n)$ belongs to the core. More general cases in which the core is not empty are discussed in [1] and [3]. A related case in which the core might be empty is presented in [11].

The nucleolus. For the definition of the nucleolus, if $x \in R^n$ then let $\theta(x)$ be the $(2^n - 2)$-tuple of the numbers $e(S, x)$ $(S \in 2^N \equiv \{S \subset N : \emptyset \neq S \neq N\})$, arranged in order of increasing magnitude. Then the nucleolus $z = (z_1, \ldots, z_n)$ is defined to be the unique vector (see [13]) which lexicographically maximizes the function $\theta(x)$ in the set[1] $\{x \in R^n : x_i \geqslant v(\{i\}), i \in N, x(N) = v(N)\}$. Since the core of our game is not

[1] Obviously, in our class of games $v(N) \leqslant \sum_{i \in N} v(\{i\})$.

empty, it follows that the nucleolus z also belongs to the core ([13, theorem 4]). Thus,

(1) *for every* $S \subset N$, $e(S, z) \geqslant 0$.

In other words, if cost is allocated according to the nucleolus then every set of users does not have to pay more than what it should have paid if the other users did not cooperate. Moreover, the minimum gain, $v(S) - z(S)$, of a set S is maximized by the nucleolus, and then the second smallest gain is maximized and so forth.

In this section we present several properties of the nucleolus in our class of games and the goal is developing an efficient algorithm for the nucleolus. Let us assume, without loss of generality, that the sons of the root are precisely the nodes $1, \ldots, k$ $(1 \leqslant k \leqslant n)$. We shall show that it could further be assumed that $k = 1$, i.e., the root has a unique son. Since $\sum_{i=1}^{k} v(B(i)) = v(N)$ and $N = \bigcup_{i=1}^{k} B(i)$ is a partition, it follows from (1) that $z(B(i)) = v(B(i))$. Consider the subgame induced on the branch $B(i)$, i.e., the restriction v^i of v to sets $S \subset B(i)$, and let z^i denote the nucleolus of v^i. It can be easily verified that for every $j \in B(i)$, $z_j = z_j^i$. In other words, if the root 0 has more than one son then the tree decomposes to subtrees rooted in 0, and the degree of the root in each one of these subtrees is one. The nucleolus of the original tree could be calculated by taking the Cartesian product of the nucleoli of the subtrees. Also, we could henceforth assume that each such subtree contains at least two nodes besides the root, since the remaining case is trivial. In view of this, we assume without loss of generality that

(2) $j(i) = 0$ *if and only if* $i = 1$; *also* $n \geqslant 2$.

The algorithm which is developed below is based upon the following results which will later be stated and proved in detail. Suppose that $S^* \subset N$ is a set which minimizes the ratio of $v(S)$ to the total number of members and neighbors of S $(S \subset N)$. We shall prove that cost which the nucleolus allocates to each member of S^* is equal to the above ratio evaluated at S^*. The costs allocated to users which are not members of S^* are then calculated iteratively by considering games over the branches which correspond to the neighbors of S^*. The propositions stated below are also hoped to provide some insight into the "philosophy" of the nucleolus.

(3) *For every* $S \in 2^N$, $e(S, z) \geqslant e(S', z)$.

This follows from the fact that $v(S') = v(S)$ and $z_i \geqslant 0$, $i \in N$.

(4) *If* j *is an ancestor of* i *then* $z_j \leqslant z_i$.

This is a consequence of [13, theorem 3] and [10, theorem 9.3].

(5) $e(S, z) > 0$ *for every* $S \in 2^N$; *in particular* $z_i < v(\{i\})$, $i \in N$.

This follows from (2). It suffices to show that for some $x = (x_1, \ldots, x_n)$, $e(S, x) > 0$ for every $S \in 2^N$. Such an x could be defined by $x_1 = 0$ and $x_i = d_i + d_1/(n-1)$ for $i \neq 1$.

(6) *For every branch* B, $z(B) > d(B)$.

Since $B \neq N$, it follows by (5) that

$$d(B) = v(N) - v(N \backslash B) < z(N) - z(N \backslash B) = z(B).$$

Let D denote the class of all basic sets of T.

(7) *For every* $S \in 2^N$ *there is an* $R \in D \cup \{N \backslash \{1\}\}$ *such that* $e(R, z) \leqslant e(S, z)$.

A proof of (7) follows. Firstly, if $S' = N$, let $i \in N$ be such that $i \notin S$. Then, $e(S, z) \geqslant e(N \backslash \{i\}, z) = z_i \geqslant z_1 = e(N \backslash \{1\}, z)$. Secondly, if $S' \neq N$ and $i_1, \ldots, i_k$ $(k \geqslant 1)$ are the neighbors of S', let $R = S' \cup B(i_1) \cup \cdots \cup B(i_{k-1})$. If $k > 1$ then (3) and (6) imply $e(R, z) < e(S, z)$ and also clearly $R \in D$. The remaining case, $k = 1$, is trivial.

(8) $z_1 = e(N \backslash \{1\}, z) = \min\{e(C, z) : C \in D\}$.

We prove (8) as follows. If $z_1 < \min\{e(C, z) : C \in D\}$ then let $y = (y_1, \ldots, y_n)$ be defined by $y_1 = z_1 + \epsilon$, $y_i = z_i - \epsilon/(n-1)$, $i = 2, \ldots, n$, where $\epsilon > 0$ is sufficiently small. By (7), $\min\{e(S, z) : S \in 2^N\} < \min\{e(S, y) : S \in 2^N\}$, in contradiction to the assumption that z is the nucleolus. If $z_1 > \min\{e(C, z) : C \in D\}$ let $y_1 = z_1 - \epsilon$, $y_i = z_i + \epsilon/(n-1)$, $i = 2, \dots, n$, and for a sufficiently small $\epsilon > 0$ a contradiction follows analogously.

Recall that $D = \{N \setminus B(j) : j = 2, 3, \dots, n\}$. Let J be defined by $j \in J \Leftrightarrow z_1 = e(N \setminus B(j), z)$. Let $J^* \subset J$ be the set of all $j \in J$ such that there is no ancestor k of j in J. Thus, $\bigcup_{j \in J} B(j) = \bigcup_{j \in J^*} B(j)$ and $B(j) \cap B(k) = \emptyset$ for every pair $j, k \in J^*$ $(j \neq k)$. Denote $W = N \setminus \bigcup_{j \in J} B(j)$.

(9) *For every $i \in W$ $z_i = z_1$.*

By (4) $z_i \geqslant z_1$. Suppose, per absurdum, that $z_i > z_1$ for some $i \in W$. Define $y = (y_1, \dots, y_n)$ by $y_1 = z_1 + \epsilon$, $y_i = z_i - \epsilon$ and $y_j = z_j$, $j \in N \setminus \{1, i\}$, where $\epsilon > 0$ is sufficiently small. If $S \in 2^N$ is such that $e(S, y) < e(S, z)$ then $1 \in S$ and $i \notin S$. If S is such that $e(S, z) = z_1$ then $S' \neq N$ (otherwise, if $S' = N$, then $e(S, z) \geqslant z_i > z_1$). In that case $S' = N \setminus B(j)$ for some j (since by (6) S' cannot have more than one neighbor). Since $S' \in 2^N$ it follows by (3) and (8) that $e(S, z) = e(S', z)$. Hence $S = S'$ and we reach the contradiction that $i \in B(j)$. Thus necessarily, $e(S, z) > z_1$ for every $S \in 2^N$ such that $e(S, y) < e(S, z)$. This implies that $\theta(y)$ is lexicographically greater than $\theta(z)$ and hence, a contradiction.

(10) *For every $i \in W$ $z_i = v(W)/(|W| + |J^*|)$.*

This follows from (9). Suppose that $J^* = \{j_1, \dots, j_k\}$. Then

$$\begin{array}{llllll}
z_1 & -z(B(j_1)) & & & = -d(B(j_1)), \\
z_1 & & -z(B(j_2)) & & = -d(B(j_2)), \\
\vdots & & & & \\
z_1 & & & -z(B(j_k)) & = -d(B(j_k)), \\
z(N) & & & & = v(N).
\end{array}$$

These imply $kz_1 + z(W) = v(W)$. Since $z(W) = z_1 \cdot |W|$ it follows that $z_i = v(W)/(|W| + k)$ for every $i \in W$. Let $K(S)$ denote the set of neighbors of a set S and let $k(S) = |K(S)|$.

(11) *For every $S \in 2^N$, $z_1 \leqslant v(S)/(|S| + k(S))$.*

Let $S \in 2^N$ be any set and, without loss of generality, assume that S is closed. Notice that $z(N) = v(N)$ and $z_i \leqslant v(\{i\})$, $i \in N$. For every i, $e(N \setminus \{i\}, z) \leqslant z_i$ and for every $j \in K(S)$, $t_j \equiv e(N \setminus B(j), z) = z(B(j)) - d(B(j))$. Thus, by (7) and (8),

$$\begin{aligned}
z_1 &\leqslant \min\left[\min\{z_i : i \in S\}, \min\{t_j : j \in K(S)\}\right] \\
&\leqslant \frac{\sum_{i \in S} z_i + \sum_{j \in K(S)} t_j}{|S| + k(S)} = \frac{z(N) - \sum_{j \in K(S)} d(B(j))}{|S| + k(S)} \\
&= \frac{v(S)}{|S| + k(S)}.
\end{aligned}$$

(12) *If S^* minimizes $v(S)/(|S| + k(S))$ over 2^N then for every $i \in S^*$ and $j \in K(S^*)$, $z_i = t_j = v(S^*)/(|S| + k(S^*))$.*

By (10), $z_1 = v(S^*)/(|S| + k(S^*))$. On the other hand, $z_1 \leqslant z_i$ for every $i \in S^*$ and $z_1 \leqslant t_j$ for every $j \in K(S^*)$. Thus,

$$z_1 \cdot (|S^*| + k(S^*)) \leqslant z(S^*) + \sum_{j \in K(S^*)} \left[z(B(j)) - d(B(j))\right] = v(S^*).$$

However, since the latter must be an equality (see (10)) it follows that $z_1 = z_i = t_j$ for every $i \in S^*$ and $j \in K(S^*)$.

Let S^* be any set which minimizes $v(S)/(|S| + k(S))$ over 2^N. For every branch

$B(i)$, where i is a neighbor of S^*, let T^i denote the tree obtained by deleting from T all nodes $j \notin B(i)$ and adjoining a unique arc (O_i, i) with $d(O_i, i) = d_i + z_1$ (O_i is the root of T^i). Let z^i denote the nucleolus with respect to T^i ($z^i = (z_j^i)_{j \in B(i)}$).

(13) *For every $i \in K(S^*)$ and $j \in B(i)$, $z_j = z_j^i$.*

Let v^i denote the characteristic function with respect to tree T^i, namely, $v^i(S) = v(S \cup N \backslash B(i)) - v(N \backslash B(i)) + z_1$. Suppose, per absurdum, that the claim is not true for some $i \in K(S^*)$. Let $\hat{z}$ denote the restriction of z to the coordinates $j \in B(i)$. Thus, $\hat{z} \neq z^i$. Obviously $\hat{z}_j \leqslant v^i(\{j\})$ (otherwise, if $\hat{z}_j > v^i(\{j\})$, we reach the contradication $e(\{j\} \cup N \backslash B(i), z) < 0$) and $\hat{z}(B(i)) = v^i(B(i))$. It follows that[2] $\theta^i(\hat{z})$ is lexicographically smaller than $\theta^i(z^i)$. Suppose that

$$\theta^i(\hat{z}) = (e^i(S_1, \hat{z}), \ldots, e^i(S_m, \hat{z})),$$
$$\theta^i(z^i) = (e^i(T_1, z^i), \ldots, e^i(T_m, z^i))$$

where $e^i(S_1, \hat{z}) \leqslant \cdots \leqslant e^i(S_m, \hat{z})$, $e^i(T_1, z^i) \leqslant \cdots \leqslant e^i(T_m, z^i)$ and j_0 is such that $e^i(S_{j_0}, \hat{z}) < e^i(T_{j_0}, z^i)$ and for $j < j_0$, $e^i(S_j, \hat{z}) = e^i(T_j, z^i)$. Define $y = (y_1, \ldots, y_n)$ by $y_j = z_j$ for $j \notin B(i)$ and $y_j = z_j^i$ for $j \in B(i)$. It follows from (3) and (6) that for every $R \subset N \backslash B(i)$ and $S \subset B(i)$, $e(S \cup N \backslash B(i), z) \leqslant e(S \cup R, z)$ and $e(S \cup N \backslash B(i), y) \leqslant e(S \cup R, y)$. Also, for $j < j_0$ and every $R \subset N \backslash B(i)$, $e(S_j \cup R, z) = e(T_j \cup R, y)$ and $e(S_{j_0} \cup N \backslash B(i), z) < e(T_{j_0} \cup N \backslash B(i), y)$. It thus follows that $\theta(y)$ is lexicographically greater than $\theta(z)$. Since $y(N) = v(N)$ and $y_j \leqslant v(\{j\})$ for $j \in N$, this implies a contradiction.

Propositions (12) and (13) imply that the following algorithm computes the nucleolus z. The algorithm operates on a list L of subtrees (of the type T^i mentioned above) of T. For a tree τ in L let $N(\tau)$ denote the "set of users" of τ.

Algorithm

0. Initiate with $L = (T)$ and $d_i = d(j(i), i)$ ($i \in N$).
1. Let τ be the next tree in L. Find a closed set $S^* \in 2^{N(\tau)}$ which minimizes $\zeta(S) \equiv d(S)/(|S| + k(S))$. Notice that $\zeta(S)$ is the total length (with respect to τ) of the subtree corresponding to S, divided by the number of nodes and neighbors of this subtree.
2. Delete τ from L.
3. Assign $z_i = \zeta(S^*)$ for every $i \in S^*$.
4. For every neighbor i of S^*, if i is a leaf of T then assign $z_i = \zeta(S^*) + d_i$; otherwise, set $d_i = d_i + \zeta(S^*)$ and adjoin the subtree T^i to the list L.
5. If $L = \emptyset$ terminate; otherwise, go to 1.

The validity of this algorithm follows from (12) and (13). Notice that the search for S^* in step 1 is carried out in a domain whose size may grow exponentially with n. However, this search can always be completed within polynomial-time bound, as we show in the Appendix. To apply the algorithm of the Appendix, delete the leaves of τ (obviously no leaf belongs to S^*) and define $a_i = d_i$, $b_i = k(\{i\})$ for the remaining nodes. The algorithm of the Appendix terminates within $O(n^2)$ operations, and there are n subtrees of the type T^i. Hence, the nucleolus is computed within $O(n^3)$ operations.

The Shapley value. The Shapley allocation could be defined as $s = (s_1, \ldots, s_n)$ where

$$s_i = \sum_{S \subset N} \frac{|S|!\,(n - |S| - 1)!}{n!} \left[v(S \cup \{i\}) - v(S) \right]$$

(see [14] for the properties characterizing s).

[2] We denote e^i, θ^i, etc., for the operators e, θ, etc., respectively, when these relate to the game v^i.

Unlike the case of nucleolus, the analysis of the Shapley allocation is much simpler. It is based on the additive nature of s as an operator over the class of games, i.e., $s_i[v+u] = s_i[v] + s_i[u]$ (see [14]). It turns out that our function v can be expressed as a sum of functions v^i for which the Shapley allocation has a simple form. These are defined as follows. For $i \in N$ and $S \in 2^N$ let $v^i(S) = d_i$ if $S \cap B(i) \neq \emptyset$ and $v^i(S) = 0$ otherwise. It can be easily verified that $v(S) = \sum_{i \in N} v^i(S)$. The value for the game v^i is simply $s_j[v^i] = d_i/|B(i)|$ if $j \in B(i)$, and $s_j[v^i] = 0$ otherwise. It follows that the Shapley cost allocation is characterized by the property that the cost of each arc is allocated equally to the users of the arc. The allocation s can be computed efficiently by the following algorithm. Let $b(i) = |B(i)|$, $i \in N$. The numbers $b(i)$ can be calculated by a "depth-first-search" over T, beginning at node 1 (see [16]). This is done in $O(n)$ operations. Then s is defined recursively by $s_i = s_{j(i)} + d_i/b_i$, starting from $s_0 = 0$. Thus, the whole computation requires only $O(n)$ operations.

Appendix. Minimizing $(\sum_{i \in S} a_i)/(\sum_{i \in S} b_i)$ over concentric subtrees S of a directed tree.

A subtree S of a directed tree T is called *concentric* if both have the same root. Let T be a directed tree whose set of nodes is $\{1, \ldots, m\}$, where 1 is the root. We shall use the symbol S, without confusion, both for a subtree and for its set of nodes. Thus, a set $S \subseteq \{1, \ldots, m\}$ corresponds to a concentric subtree if and only if $1 \in S$ and for every $1 \neq i \in S$ all the ancestors of i belong to S. For any vector $y = (y_1, \ldots, y_m)$ and $S \subset T$ denote $y(S) = \sum_{i \in S} y_i$ ($y(\emptyset) = 0$). We shall consider the following problem.

PROBLEM. Given a nonnegative vector $a = (a_1, \ldots, a_m)$ and a positive vector $b = (b_1, \ldots, b_m)$, find a concentric subtree S of T which minimizes the ratio $\gamma(S) \equiv a(S)/b(S)$.

We shall develop a polynomial-time algorithm for this problem. For every $i \in T$ let T_i denote the maximal subtree of T which is rooted in i. Notice that $T_1 = T$. Let K_i denote the set of "sons" of i. Given a nonnegative number r, we say that node i is *r-active* if there is a subtree S of T, which is rooted in i, such that $\gamma(S) \leqslant r$. Notice that by our terminology S is a concentric subtree of T_i. The set of all r-active nodes of T will be denoted by T^r and we also denote $K_i^r = T^r \cap K_i$.

For any given $r \geqslant 0$ we now define subtrees S_i^r ($i = 1, \ldots, n$), such that S_i^r is rooted in i. The definition is recursive: (i) If i a leaf let $S_i^r = \{i\}$. (ii) Assuming S_j^r has been defined for all $j \in K_i$, let $S_i^r = \{i\} \cup \bigcup\{S_j^r : j \in K_i^r\}$. Thus, S_i^r consists of i itself and the subtrees of S_j^r which correspond to r-active sons of i.

LEMMA. *For every $i \in T$ and $r \geqslant 0$, i is r-active if and only if $\gamma(S_i^r) \leqslant r$.*

PROOF. The "if" part is definitional. We shall prove the "only if" part by induction. The claim is trivial if i is a leaf. Assume that the claim is true for every $j \in T_i \backslash \{i\}$, and suppose that i is r-active. Thus, there is a subtree S^* of T, rooted in i, such that $\gamma(S^*) \leqslant r$. If $j \in S^*$ is a son of i and is not r-active then, by definition, $\gamma(S^* \cap T_j) > r$ and it follows that $\gamma(S^* \backslash T_j) \leqslant r$. Similarly, if $j \notin S^*$ is an r-active son of i then, by the induction hypothesis, $\gamma(S_j^r) \leqslant r$ and it follows that $\gamma(S^* \cup S_j^r) \leqslant r$. These two observations imply (by deleting T_j for non-r-active sons and adjoining S_j^r for r-active sons which do not belong to S^*, and applying this successively to every node) that in fact $\gamma(S_i^r) \leqslant r$. This completes the proof.

Let $q = \min\{\gamma(S) : S \text{ is a concentric subtree of } T\}$. ∎

THEOREM. *$r = q$ if and only if $\gamma(S_1^r) = r$.*

PROOF. For the "only if" part, suppose that $r = q$. Thus, 1 is r-active and by the lemma $\gamma(S_1^r) \leqslant r$. However, $\gamma(S_1^r) \geqslant q$ and hence $\gamma(S_1^r) = r$.

For the "if" part, suppose first that $r < q$. In this case, by the definition of q, 1 is not r-active and hence, by the lemma, $\gamma(S_1^r) > r$. For the remaining case, suppose that $r > q$. We already know by the "only if" part that $\gamma(S_1^q) = q$. Thus, $\gamma(S_1^q) < r$. If every $i \in S_1^r \backslash \{1\}$ is q-active then $S_1^r = S_1^q$ and hence $\gamma(S_1^r) = \gamma(S_1^q) = q < r$; otherwise, there is an $i \in S_1^r \backslash \{1\}$ which is not q-active (but is r-active of course). Notice that S_1^q is a concentric proper subtree of S_1^r. Let V denote the set of all nodes $i \in S_1^r \backslash S_1^q$ such that i is a son of a node in S_1^q. Thus, $S_1^r = S_1^q \cup \bigcup\{S_i^r : i \in V\}$ and also $S_i^r \cap S_j^r = \emptyset$ if $i, j \in V$ and $i \neq j$. Since for every $i \in V$ $\gamma(S_1^q) = q < \gamma(S_i^r) \leqslant r$, it follows by the implication $A/B < C/D \Rightarrow (A + C)/(B + D) < C/D$ $(A, C \geqslant 0$, $B, D > 0)$ that

$$\gamma(S_1^r) = \frac{a(S_1^q) + \sum_{i \in V} a(S_i^r)}{b(S_1^q) + \sum_{i \in V} b(S_i^r)} < \operatorname{Max}\{\gamma(S_i^r) : i \in V\} \leqslant r$$

and this completes the proof. ∎

Our algorithm is based upon the theorem and could be roughly described as follows. If $r \geqslant q$ is an approximation for q (we initiate with $r = a_1/b_1$) then we construct the subtrees S_i^r. By the lemma, these could be constructed according to the formula

$$S_i^r = \{i\} \cup \bigcup\{S_j^r : j \in K_i, \gamma(S_j^r) \leqslant r\}.$$

Since node 1 is r-active $(r \geqslant q)$, it follows by the lemma that $\gamma(S_1^r) \leqslant r$. Moreover, if $\gamma(S_1^r) = r$ then by the theorem $r = q$; otherwise, $\gamma(S_1^r)$ is a better approximation for q, i.e., $q \leqslant \gamma(S_1^r) < r$. A detailed description follows.

Algorithm

0. Initiate with $r = a_1/b_1$, and $S = N$.

1. For every leaf $i \in S$, define labels $\alpha_i = a_i$, $\beta_i = b_i$ and delete i from S if $\alpha_i/\beta_i > r$.

2. If i is unlabeled and every son $j \in S$ of i is labeled, then define labels $\alpha_i = a_i + \alpha(S \cap K_i)$, $\beta_i = b_i + \beta(S \cap K_i)$. If $\alpha_i/\beta_i > r$ then delete all $j \in S \cap T_i$ from S. Repeat this step until node 1 is labeled.

3. If $\alpha_1/\beta_1 = r$ then terminate; otherwise (necessarily $\alpha_1/\beta_1 = \gamma(S_1^r) < r$) define $r = \alpha_1/\beta_1$, erase all labels and go to 1.

When the algorithm terminates, S is a concentric subtree such that $\gamma(S) = q$. The algorithm must terminate within m iterations since at least one node is deleted during an iteration which ends with $\alpha_1/\beta_1 < r$ (except, perhaps, the first iteration). The number of operations per deletion is not greater than $O(m)$. Thus, the algorithm terminates within $O(m^2)$ operations.

References

[1] Bird, C. G. (1976). On Cost Allocation for a Spanning Tree: A Game Theoretic Approach. *Networks* **6** 335–350.

[2] Claus, A. and Granot, D. (June 1976). Game Theory Application to Cost Allocation for a Spanning Tree. Working Paper No. 402, Faculty of Commerce and Business Administration, University of British Columbia.

[3] Granot, D. and Huberman, G. (June 1976). On the Core of a Minimum Spanning Tree Game. Working Paper No. 403, Faculty of Commerce and Business Administration, University of British Columbia.

[4] Jeroslow, R. (1973). The Simplex Algorithm with the Pivot Rule of Maximizing Criterion Improvement. *Discrete Math.* **4** 367–377.

[5] Klee, V. and Minty, G. J. (1972). How Good Is the Simplex Algorithm? *Inequalities* **3** 159–175. Academic Press, New York.

[6] Kopelowitz, A. (1967). Computation of the Kernels of Simple Games and the Nucleolus of n-Person Games. Res. Memo 31, Research Program in Game Theory and Mathematical Economics, Dept. of Mathematics, Hewbrew University of Jerusalem.

[7] Littlechild, S. C. (1974). A Simple Expression for the Nucleolus in a Special Case. *Internat. J. Game Theory* **3** 21–29.

[8] ——— and Owen, G. (1973). A Simple Expression for the Shapley Value in a Special Case. *Management Sci.* **20** 370–372.

[9] ——— and Thompson, G. F. (December 1973). Aircraft Landing Fees: A Game Theory Approach. Working Papers Series 16, Management Center, University of Aston, Birmingham.

[10] Maschler, M. and Peleg, B. (1956). A Characterization, Existence Proof and Dimension Bounds for the Kernel of a Game. *Pacific J. Math.* **18** 289–328.

[11] Megiddo, N. (1978). Cost Allocation for Steiner Trees. *Networks* **8** 1–6.

[12] Owen, G. (1968). *Game Theory*. W. B. Saunders Co., Philadelphia.

[13] Schmeidler, D. (1969). The Nucleolus of a Characteristic Function Game. *SIAM J. Appl. Math.* **17** 1163–1170.

[14] Shapley, L. S. (1953). A Value for n-Person Games. *Contributions to the Theory of Games* **II** 307–317. H. W. Kuhn and A. W. Tucker, eds. Annals of Mathematics Studies, No. 28. Princeton University Press, Princeton.

[15] Suzuki, M. and Nakayama, M. (1976). The Cost Allocation of Cooperative Resource Development: A Game Theoretical Approach. *Management Sci.* **22** 1081–1086.

[16] Tarjan, R. (1972). Depth-First-Search and Linear Graph Algorithms. *SIAM J. Computing* **2** 146–160.

DEPARTMENT OF STATISTICS, TEL AVIV UNIVERSITY, TEL AVIV, ISRAEL