International Journal of Game Theory (1996) 25: 219–244

The Kernel/Nucleolus of a Standard Tree Game

D. GRANOT[1]

The University of British Columbia, Vancouver, Canada

M. MASCHLER

Institute of Mathematics, The Hebrew University, Givat Ram, 91904 Jerusalem, Israel

G. OWEN[2]

Naval Post Graduate School, Monterey, California, USA

W. R. ZHU[3]

The University of British Columbia, Vancouver, Canada

Abstract: In this paper we characterize the nucleolus (which coincides with the kernel) of a tree enterprise. We also provide a new algorithm to compute it, which sheds light on its structure. We show that in particular cases, including a chain enterprise one can compute the nucleolus in $O(n)$ operations, where n is the number of vertices in the tree.

1 Introduction

In this paper we consider cost allocation problems defined on a *tree enterprise.* Such problems were essentially considered by several authors (Bird [1976], Claus and Granot [1976], Megiddo [1978], Granot and Huberman [1981], [1984], Granot and Granot [1992]). A special case, when the tree is a chain,[4] was studied by Littlechild [1974], Littlechild and Owen [1977], and Littlechild and Thompson [1977]. Megiddo [1978] produced an algorithm to compute the *nucleolus* of the tree enterprise[5] in $O(n^3)$ operations, where n is the number of vertices of the tree. By using efficient mergeable heaps Galil [1980] has reduced the number of operations needed for Megiddo's procedure to $O(n \log n)$.

In this paper we study the structure of the nucleolus[6] of the class of these enterprises; namely, we characterize the shape of the nucleolus in terms that are

[1] Partially supported by Natural Science and Engineering Council of Canada, grant A4181

[2] Supported by National Science Foundation, grant DMS-9116416.

[3] Partially supported by Natural Science and Engineering Council of Canada, grant A3998.

[4] I.e. all arcs are located along a line.

[5] Littlechild [1974], Littlechild and Owen [1977], and Littlechild and Thompson [1977] compute the nucleolus in the chain case.

[6] Since the corresponding games are convex, the nucleolus coincides with the kernel of the tree enterprise.

0020–7276/96/2/219–244 $ 2.50

common to all the enterprises. Roughly, it is obtained by dividing the "debt" at each vertex between the residents at the vertex and the arcs leaving that vertex. Each resident is charged his part of the debt and the same part of the debt is added to the cost of each arc leaving the vertex, to form a new debt. All the above is coupled with a process called "elimination of an arc", which is performed at some arcs at which a player at the head of an arc is charged more than a player at the tail of the arc. The hardest part of our analysis is to determine what arcs should be eliminated. This determination leads to a new and interesting algorithm to compute the nucleolus, which sheds more light on its structure.

This paper is organized as follows: In Section 2 we prepare the background and introduce the necessary notation. In Section 3 we set up a system of equations and prove that they determine the nucleolus. One interesting aspect here is the fact that the number of equations is $O(|N|)$, rather than $O(|N|^2)$, where $|N|$ is the number of players. Section 4 provides a characterization of the nucleolus for the class of the tree enterprises. Section 5 provides examples which show that the above characterization is not sufficient in order to get to the nucleolus. One has to determine which arcs to eliminate. This is done in Section 6. In this Section we also present our alternative algorithm to compute the nucleolus. In Section 7 we show that further short cuts can still be made in the algorithm. In particular, we show that in the case of a chain enterprise we can compute the nucleolus in $O(n)$ operations.

2 The Tree Enterprise and its Game

In this paper we consider a *tree enterprise* $\mathscr{E} := (V, E, a, b, N)$. Here (V, E) is a finite directed tree with vertex set V and arc set E. The set V contains a distinguished node 0, called *the root* of the tree.[7] The function $a: E \to \mathfrak{R}$, called *the arc-cost-function*, associates with each arc e a cost $a(e)$, interpreted as the cost to construct e. The function $b: V \to \mathfrak{R}$, called the *vertex-cost-function*, associated with each vertex v a cost $b(v)$, interpreted as the cost to construct v. All arcs are directed away from 0 and the construction of arcs not in the tree are regarded infeasible (or too costly). Other interpretations are, of course, also possible. $N = \{1, 2, \ldots, n\}$ is a finite set of *players*, each "located" at ("resides" at, "occupies") a specific vertex. We denote by v^i the vertex occupied by player i and we denote by e^i the arc entering v^i. We allow for several players to occupy the same vertex, in which case we call them *neighbors*. We also allow non-occupied vertices, in which case we call them *public vertices*. However, *we assume that each player occupies exactly one vertex*. Note that a and b may take negative values, interpreted as proceeds paid

[7] By "a directed tree" we mean a directed graph such that to each vertex there is a unique path from the root to that vertex.

to whoever constructs the corresponding arc, or vertex. The objective of the players is to connect themselves to the root (think of a cablevision network).

With each enterprise $\mathscr{E}$ we can associate a *cost game* $\Gamma_{\mathscr{E}} := (N; c)$, where, for each coalition S we define

$$c(S) = \text{minimal cost needed to join all members of } S \text{ to the root via a connected subgraph of } (V, E) \tag{2.1}$$

Remark 2.1: If a subgraph contains an arc, it is automatically assumed that it contains also its endpoints, unless stated otherwise. Thus, when S chooses a subgraph it has to pay all the costs of its arcs and their endpoints.

Remark 2.2: As the costs are the costs to construct the arcs and vertices. If several players "use" an arc, or a vertex, they have to pay only once for the construction.

Remark 2.3: Since a and b may take negative values, the optimal subgraph for a coalition need not be unique, nor need it be the *natural subgraph*[8] of S. It may be worthwhile for the members of S to construct additional acrs and vertices, in which case we say that it is worthwhile for S to *swallow* these additional parts.

We shall denote by (V^S, E^S) the optimal subgraph for coalition S and if there are several optimal subgraphs (because of subtrees that cost noting, if added) then, unless specified otherwise, (V^S, E^S) will denote the unique subgraph which is maximal under inclusion.

Remark 2.4: The empty coalition need not have a zero worth. By convention, it has to pay the cost of the root and, if it is worthwhile, it can swallow a connected subtree, connected to the origin.

We shall encounter general trees as described above, but actually the main purpose of this paper is to determine the structure of the kernel and nucleolus of a game corresponding to a *standard tree enterprise*, defined subsequently.[9]

Definition 2.5: A tree enterprise $\mathscr{E} = (V, E, a, b, N)$ is called *a standard tree enterprise* if

(i) $b(v) = 0$ for each v in V,
(ii) $a(e) \geq 0$ for each e in E,
(iii) The root is not occupied,

[8] Namely, the subgraph consisting of those arcs and vertices that must be traversed in order to join all members of S to the root.

[9] Henceforth, we shall refer to the kernel/nucleolus of the tree enterprise and mean the kernel/nucleolus of the corresponding game. A similar convention will apply to other solution concepts, such as the core.

(iv) All vertices are occupied,
(v) Only one arc leaves the root.

Standard trees, in which every vertex except the root is occupied by exactly one player were discussed in Megiddo [1978] and Granot and Huberman [1981]. Because $b(v)=0$, standard tree enterprises will be denoted $\mathscr{E}=(V,E,a,N)$. The following theorem enables us to deduce several facts, known from the literature.

Theorem 2.6:[10] The game associated with a standard tree enterprise is a convex game;[11] i.e.,

$$c(T\cup\{i\})-c(T)\leq c(S\cup\{i\})-c(S), \tag{2.2}$$

whenever $S\subset T$ and $i\notin T$ (see Shapley [1971]).

Proof: Let R be an arbitrary coalition. Because the tree is standard, $c(R)$ is the cost of the natural subgraph for R (Remark 2.3). We shall denote this subgraph[12] by (V^R,E^R). Thus,[13] $c(R)=a(E^R)$. Let $S\subset T$ and $i\notin T$. Note that $(V^{T\cup\{i\}},E^{T\cup\{i\}})$ contains (V^T,E^T), because in a directed tree, there is a unique path from the root to every vertex. Similarly, $(V^{S\cup\{i\}},E^{S\cup\{i\}})$ contains (V^S,E^S). The set $E^{T\cup\{i\}}\backslash E^T$ is the set of additional arcs needed to connect i to the root, given that T is already connected to the root. Thus,

$$c(T\cup\{i\})-c(T)=a(E^{T\cup\{i\}}\backslash E^T). \tag{2.3}$$

Similarly,

$$c(S\cup\{i\})-c(S)=a(E^{S\cup\{i\}}\backslash E^S). \tag{2.4}$$

Relation (2.2) now follows from (2.1), because $a\geq 0$ and $E^{S\cup\{i\}}\backslash E^S\supseteq E^{T\cup\{i\}}\backslash E^T$. □

Corollary 2.7: A standard tree enterprise has a nonempty core – in fact, a regular core (Shapley [1971]). Its kernel[14] consists of a unique point in the core and therefore coincides with its nucleolus.

Remark 2.8: Of the Conditions of Definition 2.5, only the first and the second were employed; therefore, Theorem 2.6 holds for a wider class of trees, in which

[10] This theorem was mentioned in Granot and Huberman [1982].
[11] Actually, a concave game. We shall use the terminology which refer to the games $(N;-c)$. Similarly, we shall say "core", "nucleolus", etc. instead of "anti-core", "anti-nucleolus", etc.
[12] Contrary to the general convention, when the optimal subgraph is not unique.
[13] $a(E^R)=\sum\{a(e):e\in E^R\}$.
[14] Which is also its prekernel (see (3.1)–(3.2)), because a convex game is 0-monotonic (Maschler and Peleg [1967]).

the other conditions may be violated. In fact, it is enough to require that for each vertex v^i, $b(v^i) + a(e^i) \geq 0$.

Remark 2.9: Conditions (iii) and (v) in Definition 2.5 are required only for simplification purpose. If the root is occupied, we can always add a zero-cost arc from a new unoccupied root to this root without changing the characteristic function. If several arcs leave the root, then, as has been observed by Megiddo [1978], the nucleolus is simply the cartesian product of the nucleoli of the various branches emanating from the root.[15]

3 The Equations for the Kernel/Nucleolus of a Standard Tree

We achieve the task at the head of this section by employing *the reduced game property* that is satisfied by the prekernel (see, e.g. Peleg [1986]), when applied to certain 2-person *reduced games*. The reduced game property will help us get some equations that must be satisfied by the kernel/nucleolus. We then prove that these are sufficient. Later we shall construct an algorithm for computing the kernel/nucleolus and prove its validity by showing that its outcome satisfies these equations. (The reduced game property will again be employed during the development of the algorithm.)

We remind the reader that the prekernel of a cost game $(N; c)$ is the set

$$\{x \in \mathfrak{R}^n : x(N) = c(N) \text{ and } s_{kl}(x) = s_{lk}(x), \text{ for all } k, l \in N, k \neq l\}, \tag{3.1}$$

where[16]

$$s_{kl}(x) = \min\{c(S) - x(S) : S \ni k, S \not\ni l\}. \tag{3.2}$$

Notation 3.1:

(1) An ordered pair of players (i, j) is a pair of adjacent[17] players i and j, where j follows i. We shall denote by e_{ij} the arc from v^i to v^j, and also refer to it as the arc entering v^j.

(2) Let (i, j) be a pair of adjacent players. We denote by B_{ij} the subtree of (V, E) rooted at v^i and whose vertex set consists of v^i, and all vertices of the tree, such that the paths from the root to these vertices contain e_{ij}. B_{ij} will be referred to as the branch at v^i, in the direction of j. We shall also denote by B_{il} the branch

[15] This can also be proved with the help of Theorem 3.12. Granot and Huberman [1981] proved a more general result.

[16] $c(S) - x(S)$ is called the *excess* of the coalition S at x.

[17] Namely, occupying the endpoints of an arc.

B_{ij}, if l is located somewhere after j, where j is adjacent to i and is on the path from i to l. We shall refer to B_{il} as the branch at v^i in the direction of l.

(3) We denote by T_{ij} [or by T_{il}, see above] the subtree rooted at the original root, consisting of v^i and all the arcs and vertices not in B_{ij}. We shall refer to it as the trunk at v^i, away from j [from l].

(4) For an arbitrary vertex v, we denote by $x(v)$ the expression $\sum\{x_v : v \text{ resides at } v\}$. For a subtree W, we denote by $x(W)$ the expression $\sum\{x(v): v \text{ is a vertex in } W\}$.

(5) For a subtree W, we denote by $a(W)$ the expression $\sum\{a(e): e \text{ is an arc in } W\}$.

The following lemma is taken from Granot and Maschler [1994]. Essentially it appeared in Megiddo [1978] and Granot and Huberman [1984].

Lemma 3.2: If x is a core point of a standard tree enterprise $\mathscr{E}$ (Definition 2.5), then $x \geq 0$ and (see Notation 3.1) for a pair of adjacent players (i, j),

$$x(B_{ij}) - x(v^i) - a(B_{ij}) \geq 0. \tag{3.3}$$

The left side of (3.3) is the amount the players residing in B_{ij} and not in v^i pay, under x, above the cost of the branch. Thus, at a core point these players pay at least the cost of the branch.

The next lemma is a special case of a basic theorem in Granot and Maschler [1994].

Lemma 3.3: Let x be a core point of a game $\Gamma_{\mathscr{E}}$ corresponding to a standard tree enterprise $\mathscr{E} = (V, E, a, N)$. Let (i, j) be a pair of adjacent players and let $(\{i, j\}; \tilde{c}^x_{\{i,j\}})$ be the reduced game[18] on $\{i, j\}$ at x. Then this reduced game is the game corresponding to the reduced enterprise $\tilde{\mathscr{E}} = (V, E, a, \tilde{b}, \{i, j\})$, where

$$\begin{cases} \tilde{b}(v) = -x(v), & \text{if } i \text{ and } j \text{ do not reside in } v, \\ \tilde{b}(v) = -x(v) + x_i, & \text{if } i \text{ resides in } v, \\ \tilde{b}(v) = -x(v) + x_j, & \text{if } j \text{ resides in } v. \end{cases} \tag{3.4}$$

Remark 3.4: Lemmas 3.2 and 3.3 can be extended to general enterprises (V, E, a, b, N).

Thus, the reduced enterprise on $\{i, j\}$, at a core point x, is obtained from the original tree enterprise by removing all the players other than i and j and reducing by x_p the (originally zero) cost of the vertex where p resides, $p \neq i, j$. The reduced enterprise is no longer a standard tree.

[18] The reduced game on a coalition S, at a core point x, of a cost game $(N; c)$ is a game $(S; \tilde{c}^x_S)$, where $\tilde{c}^x_S(R) = \min\{c(R \cup Q) - x(Q): Q \subseteq S^c\}$, all $R \subseteq S$.

Lemma 3.5: Let x be a core point of a standard tree enterprise. Let (i, j) be a pair of adjacent players. The characteristic function of the reduced game on $\{i, j\}$, at x, is given by[19]

$$\begin{cases} \tilde{c}^x_{\{i,j\}}(i, j) = x_i + x_j, \\ \tilde{c}^x_{\{i,j\}}(j) = x_i + x_j, \\ \tilde{c}^x_{\{i,j\}}(i) = \min\{x_i + x_j, a(T_{ij}) - x(T_{ij}) + x_i\}, \\ \tilde{c}^x_{\{i,j\}}(\varnothing) = 0 \end{cases} \tag{3.5}$$

(see Notation 3.1).

Note the simple structure of the reduced game: Two coalitions have the same worth and only two options are available to the third.

Proof: Both coalitions $\{i, j\}$ and $\{j\}$ have to pay the cost of the path from the root to j. By Lemmas 3.2 and 3.3, it is worth their while to swallow every arc and vertex of the tree; hence, their worth is $c(N) - x(N\backslash\{i, j\}) = x_i + x_j$. As to coalition $\{i\}$, again, by Lemmas 3.2 and 3.3, we need to consider only two options: either to swallow v^j, knowing that x_j will not be paid back, and then to continue and swallow all the tree, or swallow only the trunk T_{ij}. □

Lemma 3.6: The prekernel of a 2-person cost game $(\{i, j\}; c)$, where $c(\varnothing) = 0$, $c(i) = a, c(j) = c(i, j) = b$, consists of the unique point $((a/2), b - (a/2))$.

Proof: The results follow from (3.1) and (3.2). □

Lemma 3.7: Let x be the kernel/nucleolus point of a standard tree enterprise $\mathscr{E}$ (Definition 2.5), then, for every pair of adjacent players (i, j) and every pair of neighbors $\{p, q\}, x$ must satisfy

$$\begin{cases} x_i = a(T_{ij}) - x(T_{ij}), & \text{if } x_j \geq a(T_{ij}) - x(T_{ij}), \\ x_i = x_j, & \text{if } x_j \leq a(T_{ij}) - x(T_{ij}), \\ x_p = x_q, \\ x(N) = c(N). \end{cases} \tag{3.6}$$

Proof: By Corollary 2.7 x is also the prekernel point. The last equation follows from (3.1). The previous one follows from the fact that p and q are symmetric players (substitutes) (see Maschler and Peleg [1966]). The first two equations follow from (3.5) and from the reduced game property of the prekernel.[20] Indeed, by Corollary 2.7, Lemma 3.6 and (3.5),

$$\begin{cases} x_i = [a(T_{ij}) - x(T_{ij}) + x_i]/2, & \text{if } x_i + x_j \geq a(T_{ij}) - x(T_{ij}) + x_i, \\ x_i = [x_i + x_j]/2, & \text{if } x_i + x_j \leq a(T_{ij}) - x(T_{ij}) + x_i. \end{cases} \tag{3.7}$$

Equations (3.6) are equivalent to (3.7). □

[19] We sometimes omit curly braces that should surround members of a coalition.

[20] Namely, if $(S; \tilde{c}^x_S)$ is the reduced game on S, at x, where x is a prekernel point then $x^S := \{x_i\}_{i \in S}$ is a prekernel point of the reduced game.

Corollary 3.8: If (i, j) is a pair of adjacent players in a standard tree game and x satisfies (3.6), then $x_j \geq x_i$.

The last conclusion makes sense intuitively. Player j needs all the arcs that are used by player i and an additional arc. He should therefore pay at least as much as player i.

Corollary 3.9: Equations (3.6) are equivalent to each of the following:

$$\begin{cases} x_i = a(T_{ij}) - x(T_{ij}) \text{ and } x_j \geq a(T_{ij}) - x(T_{ij}), & \text{if } x_j \geq x_i, \\ x_i = x_j \text{ and } x_j \leq a(T_{ij}) - x(T_{ij}), & \text{if } x_j \leq x_i, \\ x_p = x_q, \\ x(N) = c(N). \end{cases} \tag{3.8}$$

$$\begin{cases} x_i = x(B_{ij}) - a(B_{ij}) - x(v^i) \text{ and } x_j \geq x(B_{ij}) - a(B_{ij}) - x(v^i), & \text{if } x_j \geq x_i, \\ x_i = x_j \text{ and } x_j \leq x(B_{ij}) - a(B_{ij}) - x(v^i), & \text{if } x_j \leq x_i, \\ x_p = x_q, \\ x(N) = c(N). \end{cases} \tag{3.9}$$

Proof: Suppose (3.6) holds. By Corollary 3.8, $x_j < x_i$ cannot hold. If $x_j > x_i$ then $x_i = a(T_{ij}) - x(T_{ij})$ and $x_j > a(T_{ij}) - x(T_{ij})$. If $x_i = x_j$ then $x_j > a(T_{ij}) - x(T_{ij})$ would imply $x_j > x_i$, which is a contradiction. Thus, (3.8) is satisfied. We omit the proof that (3.8) implies (3.6). Equations (3.9) follow since $c(N) = a(B_{ij}) + a(T_{ij})$ and $x(N) + x(v^i) = x(B_{ij}) + x(T_{ij})$. □

Corollary 3.10: Let x be the kernel point of a tree enterprise $\mathscr{E}$. Let (i, j) be a pair of adjacent players (Notation 3.1). Then

$$x_i \leq x(B_{ij}) - a(B_{ij}) - x(v^i); \tag{3.10}$$

i.e., a player residing at v^i never pays more than the amount the players after him on a branch B_{ij} pay above the cost of the branch.

Proof: Formula (3.9). □

Lemma 3.11: A payoff vector x satisfying (3.6), or equivalently (3.8) or (3.9), is nonnegative.

Proof: Suppose not, then, by Corollary 3.8 $x_1 < 0$, where v^1 is adjacent to the root and the charges x_k to players along each path are negative until they start to become nonnegative, if at all. The players i who pay a negative amount pay less than the cost of the arcs e^i, because the cost function a is nonnegative. As soon as we arrive at a pair (i, j) of adjacent players, in which $x_i < 0, x_j \geq 0$, by the first

equation of (3.9), the players in B_{ij} other than those residing at v^i, pay together less than the cost of their branch by the amount $|x_i|$. This happens on every branch $B_{i,j}$ for which $x_i < 0$ and $x_j \geq 0$, so altogether, $x(N) < c(N)$, in contradiction with (3.9). □

Equations (3.6) (or (3.8), or (3.9)), express the fact that a prekernel point satisfies $s_{ij}(x) = s_{ji}(x)$ for each pair of adjacent players and that it is a preimputation in which neighbors pay the same amount. Equations (3.1), however, require that $s_{kl}(x) = s_{lk}(x)$ for *each* pair of distinct players. Fortunately, for a standard tree enterprise the additional requirements are satisfied automatically, as the following theorem shows.

Theorem 3.12: If x satisfies (3.6) (or, equivalently (3.8), or (3.9)), then x is the prekernel point, and therefore, the kernel/nucleolus point of the standard tree enterprise.

For the proof we need the following two lemmas.

Lemma 3.13: If x satisfies (3.6), then for every $k, l, k \neq l$, there exists a subset S, containing k but not l, for which $s_{k,l}(x) := \min\{c(R) - x(R): k \in R, l \notin R\} = c(S) - x(S)$, such that either $S = N \backslash \{l\}$, or $(V^S, E^S) = T_{ql}$ for some player q.

Proof: Let S be such a coalition, for which s_{kl} is achieved. Formally, (V^S, E^S) can employ vertices not occupied by members of S, but since $x \geq 0$ (Lemma 3.11), we may assume that S contains every player i, when $v^i \in V^S$, with exception of l, (even if $v^l \in V^S$). Suppose that $q \in S$ and let B_{qr} be a branch *not in the direction of* l. Since $0 \leq x_q \leq x(B_{qr}) - a(B_{qr}) - x(v^q)$, we shall diminish the excess if we include B_{qr} in (V^S, E^S). Thus, we can assume that S contains every B_{qr}, not in the direction of l, if it contains q. Moreover, if $l \in V^S$ then (V^S, E^S) is the full tree and $S = N \backslash \{l\}$. If $l \notin V^S$ then, in view of the above, and since (V^S, E^S) is a connected graph, connected to the root, S must be of the form T_{ql} for some q in the path from the root to l. □

Lemma 3.14: Let x satisfy (3.6). Consider a path from some v_1 to some v_{r+1} whose vertices are $v_1, v_2, \ldots, v_q, v_{q+1}, \ldots, v_r, v_{r+1}$ and let $m_1, m_2, \ldots, m_q, m_{q+1}, \ldots, m_r, m_{r+1}$ be players located at the vertices of the path, respectively. Suppose that $x_{m_1} = x_{m_2} = \cdots = x_{m_q} < x_{m_{q+1}} \leq \cdots \leq x_{m_r} \leq x_{m_{r+1}}$ under these conditions,

$$a(T_{m_q m_r}) - x(T_{m_q m_r}) \leq a(T_{m_1 m_q}) - x(T_{m_1 m_q}), \tag{3.11}$$

$$a(T_{m_q m_r}) - x(T_{m_q m_r}) < a(T_{m_r m_{r+1}}) - x(T_{m_r m_{r+1}}), \tag{3.12}$$

Consequently, the minimum of $a(T_{m_p m_{p+1}}) - x(T_{m_p m_{p+1}})$, $1 \leq p \leq r$, along this path is achieved when $T_{m_p m_{p+1}} = T_{m_q m_{q+1}}$.

Proof: Since $x_{m_q} < x_{m_{q+1}}$, it follows from (3.9) that $x_{m_q} = x(B_{m_q m_r}) - a(B_{m_q m_r}) - x(v^{m_q})$ and $x_{m_1} \leq x(B_{m_1 m_q}) - a(B_{m_1 m_q}) - x(v^{m_1})$. Thus,

$$\begin{aligned} a(T_{m_q m_r}) - x(T_{m_q m_r}) &= a(T_{m_1 m_q}) - x(T_{m_1 m_q}) \\ &\quad + a(B_{m_1 m_q}) - x(B_{m_1 m_q}) + x(v^{m_1}) \\ &\quad - [a(B_{m_q m_r}) - x(B_{m_q m_r}) + x(v^{m_q})] \\ &= a(T_{m_1 m_q}) - x(T_{m_1 m_q}) \\ &\quad + a(B_{m_1 m_q}) - x(B_{m_1 m_q}) + x(v^{m_1}) + x_{m_q} \\ &\leq a(T_{m_1 m_q}) - x(T_{m_1 m_q}) - x_{m_1} + x_{m_q} \\ &= a(T_{m_1 m_q}) - x(T_{m_1 m_q}). \end{aligned}$$

This proves (3.11).

Again, by (3.9),

$$\begin{aligned} a(T_{m_r m_{r+1}}) - x(T_{m_r m_{r+1}}) &= a(T_{m_q m_r}) - x(T_{m_q m_r}) \\ &\quad + a(B_{m_q m_r}) - x(B_{m_q m_r}) + x(v^{m_q}) \\ &\quad - [a(B_{m_r m_{r+1}}) - x(B_{m_r m_{r+1}}) + x(v^{m_r})] \\ &= a(T_{m_q m_r}) - x(T_{m_q m_r}) - x_{m_q} \\ &\quad - [a(B_{m_r m_{r+1}}) - x(B_{m_r m_{r+1}}) + x(v^{m_r})] \\ &\geq a(T_{m_q m_r}) - x(T_{m_q m_r}) - x_{m_q} + x_{m_r} \\ &> a(T_{m_q m_r}) - x(T_{m_q m_r}), \end{aligned}$$

because $x_{m_r} > x_{m_q}$. This proves (3.12)

Realizing that v_r could be any vertex beyond v_q and that v_1 could be any vertex prior to v_q, we conclude that the minimum of $a(T_{m_p m_{p+1}}) - x(T_{m_p m_{p+1}}), 1 \leq p \leq r$, along our path is indeed reached[21] when $p = q$. □

Proof of Theorem 3.12: Let k and l be two distinct players. Consier the paths from the root to v^k and to v^l and let v^m be the last vertex which is common to both paths. (We allow $m = k$, or $m = l$.) We wish to determine a coalition S of lowest excess $c(S) - x(S)$, containing k and not containing l and to determine its excess $s_{kl}(x)$ (see (3.2)). This coalition certainly contains m, so, by Lemma 3.13, we can assume that (V^S, E^S) is either the full tree, or $(V^S, E^S) = T_{ql}$, where q is a player located on the path from m to $l, m_q \neq l$. Let $v^{m_1} := v^m, v^{m_2}, \ldots, v^{m_t} := v^l$ be the vertices of the path from m to l. If $x_{m_1} = x_{m_2} = \cdots = x_{m_r} < m_{x_{r+1}} \leq \cdots \leq x_l$ then, by Lemma 3.14, we can assume that $q = m_r$. In this case, by (3.8), $s_{kl}(x) = a(T_{m_r l}) - x(T_{m_r l}) =$

[21] We remind the reader that $T_{m_p m_r}$ and $T_{m_p m_{p+1}}$ are the same trunk.

$x_{m_r} = x_m$. If $x_{m_1} = x_{m_2} = \cdots = x_l$ we claim that we can assume that (V^S, E^S) is the full tree. Indeed, if $(V^S, E^S) = T_{ql}$ and we add to it the rest of the tree, then we subtract from the excess the amount $x(B_{ql}) - a(B_{ql}) - x(v^l) - x_l$, where x_l was subtracted, because, by definition, $l \notin S$. By (3.9), this expression is not smaller than $x_q - x_l = 0$. It follows that $s_{kl}(x) = c(N) - x(T) + x_l = x_l = x_m$. We have proved that in both cases $s_{kl}(x) = x_m$. Interchanging k with l we obtain that also $s_{lk}(x) = x_m$. This proves that x is the prekernel point and therefore, the kernel/nucleolus point. □

4 The Proto-Nucleolus and the Structure of the Kernel/Nucleolus of the Standard Tree Game

In this section we shall modify the equations (3.9) and show that the modified version has a unique solution. We shall then show in what respect to nucleolus differs from this solution.

Theorem 4.1: Let $\mathscr{E}$ be a standard tree game. The system of equations

$$\begin{cases} x_i = x(B_{ij}) - a(B_{ij}) - x(v^i), & \text{for all pairs } (i,j) \text{ of adjacent players,} \\ x_p = x_q, & \text{whenever } p \text{ and } q \text{ are neighbors,} \\ x(N) = c(N), \end{cases} \tag{4.1}$$

has a unique solution, called the proto-nucleolus of $\mathscr{E}$.

Convention and Notation 4.2: We assume that the vertices of (V, E) are numbered $v_0 :=$ the root, $v_1, v_2, \ldots, v_n$ in such a way that the path from the root to a vertex v_k in (V, E) does not pass through a vertex of higher index. Figure 1 provides an example of such a labeling.

For a vertex v, we denote by dv the number of residents at v plus the number of arcs leaving v. We call dv *the degree of* v. Clearly, $dv \geq 2$ if v is not an endpoint and not the root of (V, E). We denote by $i(v)$ a player residing at v and by $e(v)$ the arc entering v.

We shall now describe an algorithm, and later prove that it yields the unique solution to equations (4.1).

Algorithm 4.3: To obtain the proto-nucleolus. We process all vertices in depth first search order.

Step 0. Initialize by charging each resident of v_1 the amount $x_{i(v_i)} = a(e(v_1))/dv_1$. Proceed to v_2, if v_2 exists; otherwise terminate.

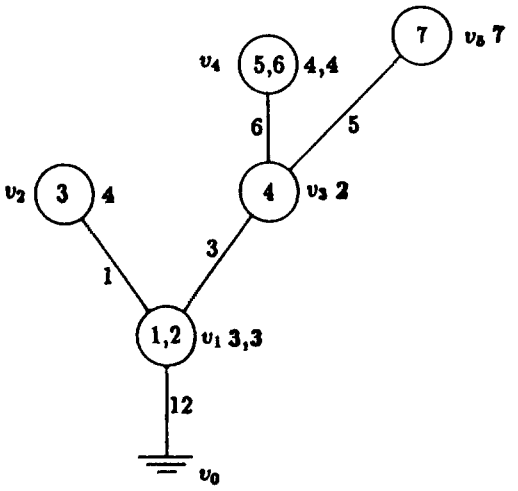

Fig. 1. Computing the proto-nucleolus

Step 1. Suppose vertex v_h has been the last to be processed. If vertex v_k follows v_h (i.e. $(v_h, v_k) \in E$) then process vertex v_k by charging each resident of v_k the amount $[x_h + a(e(v_k))]/dv_k$. If, on the other hand, v_h is an endpoint of (V, E) then backtrack, to find a vertex v_l, which has been processed and which has a follower vertex v_t that has not been processed. Proceed to process vertex v_t by charging each resident of v_t the amount $[x_l + a(e(v_t))]/dv_t$. If no such vertex v_l exists, terminate.

Verbally, by this algorithm each resident of a vertex v is charged $1/dv$ of the "debt" and this same amount is added as a debt to the cost of each arc leaving v. Figure 1 shows the resulting charges. In this and in the following figures, the residents of a vertex are encircled. The cost of arcs are placed near the arcs and the charges are typeset in bold.

Proof of Theorem 4.1: Denote the expression $x(B_{ij}) - a(B_{ij}) - x(v^i)$ by $s(B_{ij})$ and call it *the surplus of the branch* B_{ij} *at* x; for this is the amount that the players of B_{ij}, not residing at v^i pay under x, above the cost of the branch. Let $T(v^i)$ be the subtree of (V, E) consisting of v^i and all the vertices and arcs whose paths from the root do not cross v^i. Denote $a(T(v^i)) - x(T(v^i)) + x(v^i)$ by deficit $T(v^i)$ and call it *the deficit of* $T(v^i)$ *at* x. This is the amount that the players of $T(v^i)$, not residing at v^i, pay below the cost of $T(v^i)$; namely, the amount "pushed upward" by these players, that has to be paid in order to cover the costs of the tree. By (4.1), all the surpluses of the branches rooted at v^i as well as the amounts charged to the residents of v^i must add up to $x_i dv^i$, because $x_i = s(B_{ij})$ for each branch B_{ij}. This sum must cover the deficit of $T(v^i)$, at x, so

$$x_i dv^i = \text{deficit}\, T(v^i), \tag{4.2}$$

for each player i. Now, deficit $T(v_1) = a(e(v_1))$, so $x_{i(v_1)} = a(e(v_1))/dv_1$, in accordance with Step 0 of Algorithm 4.3. Consider now a vertex v_k that follows a vertex v_h. By (4.2),

$$\begin{aligned} x_{i(v_k)} dv_k &= \text{deficit } T(v_k) = a(T(v_k)) - x(T(v_k)) + x(v_k) \\ &= a(e(v_k)) + \text{deficit } T(v_h) - x_h[dv_h - 1] \\ &= a(e(v_k)) + x_h dv_h - x_h dv_h + x_h \\ &= x_h + a(e(v_k)), \end{aligned}$$

in accordance with Step 1 of the algorithm. □

In general, the proto-nucleolus is different from the nucleolus of the game. For example, the charges in Figure 1 violate Corollary 3.8. Accordingly we shall refer to arcs $e_{ij} := (v^i, v^j)$ for which $x_i > x_j$ as *bad arcs*. We shall now describe a procedure whereby one *eliminates a bad arc*, or, equivalently, one *condenses the players* residing at the endpoints of a bad arc.

Definition 4.4: Elimination of an arc/condensation of players. Let $\mathscr{E} = (V, E, a, N)$ be a standard tree enterprise and let (i, j) be a pair of adjacent players. By eliminating the arc e_{ij} [condensing the players[22] i and j] we mean a process whereby one deletes the arc, adds its cost to the cost of e^i, places all the players of v^j at the vertex v^i and replaces every arc (v^j, v^k) by an arc (v^i, v^k), having the same cost. Formally, after the condensation we obtain a standard tree enterprise $\tilde{\mathscr{E}}_{ij} := (\tilde{V}, \tilde{E}, \tilde{a}, N)$, where

(1) all residents of v^j now reside at v^i,
(2) $\tilde{V} = V \setminus \{v^j\}$
(3) $\tilde{E} = E \setminus \{e_{ij}, e_{jk} \text{ for all } k\text{'s adjacent to } j\} \cup \{e_{ik} : (j, k) \text{ were adjacent players in } \mathscr{E}\}$,
(4)

$$\tilde{a}(e) = \begin{cases} a(e), & \text{if } e \in E \cap \tilde{E}, e \neq e(v^i), \\ a(e(v^i)) + a(e(v^j)), & \text{if } e = e(v^i), \\ a(e_{jk}), & \text{if } e = e_{ik} \text{ and } (j, k) \text{ were adjacent players in } \mathscr{E}. \end{cases} \tag{4.3}$$

Note that if several arcs are eliminated consecutively, the resulting tree does not depend on the order of elimination. The following theorem and corollary show that if we knew in advance which pairs of adjacent players are charged the same amount in the nucleolus, we could have computed the nucleolus in $O(n)$ steps.

[22] We are abusing the language: Actually we condense all the players in v^j with all the players in v^i.

Theorem 4.5: Let I be a fixed set of pairs of adjacent players in a standard tree $\mathscr{E} = (V, E, a, N)$. The system of equations

$$\begin{cases} x_i = x(B_{ij}) - a(B_{ij}) - x(v^i), & \text{for all pairs of adjacent players } (i, j), \text{ not in } I, \\ x_i = x_j, & \text{for all pairs } (i, j) \text{ in } I, \\ x_p = x_q, & \text{whenever } p \text{ and } q \text{ are neighbors,} \\ x(N) = c(N), \end{cases} \tag{4.4}$$

has a unique solution. It is the proto-nucleolus of the tree obtained by condensing all the pairs in I.

Proof: By Theorem 4.1, it is sufficient to show that the system (4.4) is replaced by an equivalent system if we eliminate just one arc e_{ij}, for $(i, j) \in I$ and replace I by $I \setminus \{(i, j)\}$. Indeed, the equations in the last three rows remain unchanged and either the first equation, for a particular (i, j) in the first row remains unchanged or the equivalent equation (in view of the last one) $x_i = a(T_{ij}) - x(T_{ij})$ remains unchanged. □

Theorem 4.6: If the proto-nucleolus of a standard tree enterprise is such that $x_i \leq x_j$, whenever (i, j) are pairs of adjacent players, then the proto-nucleolus is the kernel/nucleolus point of the tree.

Proof: In this case equations (3.9) are satisfied. The result now follows from Theorem 3.12. □

Theorems 4.5 and 4.6 show that the structure of the nucleolus is the structure of a proto-nucleolus of a related tree, in which some pairs of adjacent players were condensed. Indeed, if I consists of all the adjacent pairs, who receive equal payments in the nucleolus, then, after condensing them, we obtain a tree whose proto-nucleolus charges (in both games, by Theorem 4.5) are nondecreasing along the paths. By Theorem 4.6, this proto-nucleolus is the nucleolus of both games.

Which pairs should be condensed – that will be studied in Sections 5–6. For future applications let us draw the following conclusion:

Corollary 4.7: If I is a subset of pairs of adjacent players that are charged equally at the nucleolus,[23] then the original tree and the tree obtained by condensing the pairs in I have the same kernel/nucleolus.

[23] These need not be all the pairs with this property.

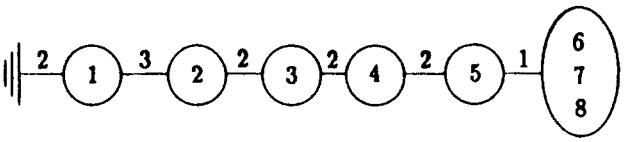

Fig. 2. Good arcs may turn out bad

5 Examples

It seems from Theorem 4.6 and from the discussion in the previous section that it is enough to eliminate bad arcs (i.e. arcs, e_{ij}, where $x_i > x_j$ in the proto-nucleolus) until one arrives at a game whose proto-nucleolus is equal to its nucleolus. Unfortunately, this is not the case! In this section we shall provide some examples which show what can go wrong.

Example 5.1: Consider the tree enterprise of Figure 2.

The proto-nucleolus of this tree enterprise turns out to be (1, 2, 2, 2, 2, 1, 1, 1) and arc e_{56} is bad. By eliminating this arc we derive a new tree enterprise, whose proto-nucleolus is (1, 2, 2, 2, 1.25, 1.25, 1.25 ,1.25). Now e_{45} is bad and after its elimination the proto-nucleolus becomes (1, 2, 2, 1.4, 1.4, 1.4, 1.4, 1.4). Now e_{34} is a bad arc, and after its elimination the proto-nucleolus is (1, 2, 1.5, 1.5, 1.5, 1.5, 1.5, 1.5). Still, arc e_{23} has to be eliminated and the final outcome is $(1, 1\frac{3}{7}, 1\frac{3}{7}, 1\frac{3}{7}, 1\frac{3}{7}, 1\frac{3}{7}, 1\frac{3}{7}, 1\frac{3}{7})$. This is the nucleolus of the resulting game (Theorem 4.6). It is *also* the nucleolus of the original game, because it satisfies, e.g., (3.9). This example shows that during the process of computing the proto-nucleolus, some calculations may turn out to be in vein.

Example 5.2: Consider the tree enterprise on the top left side of Figure 3, in which we also placed the proto-nucleolus.

We see that arcs e_{12}, e_{13} and e_{24} are bad. We proceed to eliminate e_{13} (top right) and still e_{12} and e_{24} are bad. So, we eliminate e_{12} (bottom left). Still, e_{14} is a bad arc and we eliminate it (bottom right). Finally we arrive at the nucleolus of the last tree, which is $(25\frac{5}{6}, 25\frac{5}{6}, 25\frac{5}{6}, 25\frac{5}{6}, 85\frac{5}{6}, 42\frac{11}{12}, 102\frac{11}{12})$. *It is not the nucleolus of the original tree*, as can be checked by examining (3.9). The nucleolus of the original tree is (25.8, 25.8, 25.9, 25.8, 85.8, 42.95, 102.95). We shall show in the next section that it can be obtain by an elimination first of e_{24}, followed by an elimination of e_{12}. The reader can try these eliminations now and see how the previously bad arc e_{13} now becomes a good one and should not be eliminated to begin with.

This example shows that the order in which elimination of arcs (condensation of players) is made makes a difference. Next section we shall show that processing

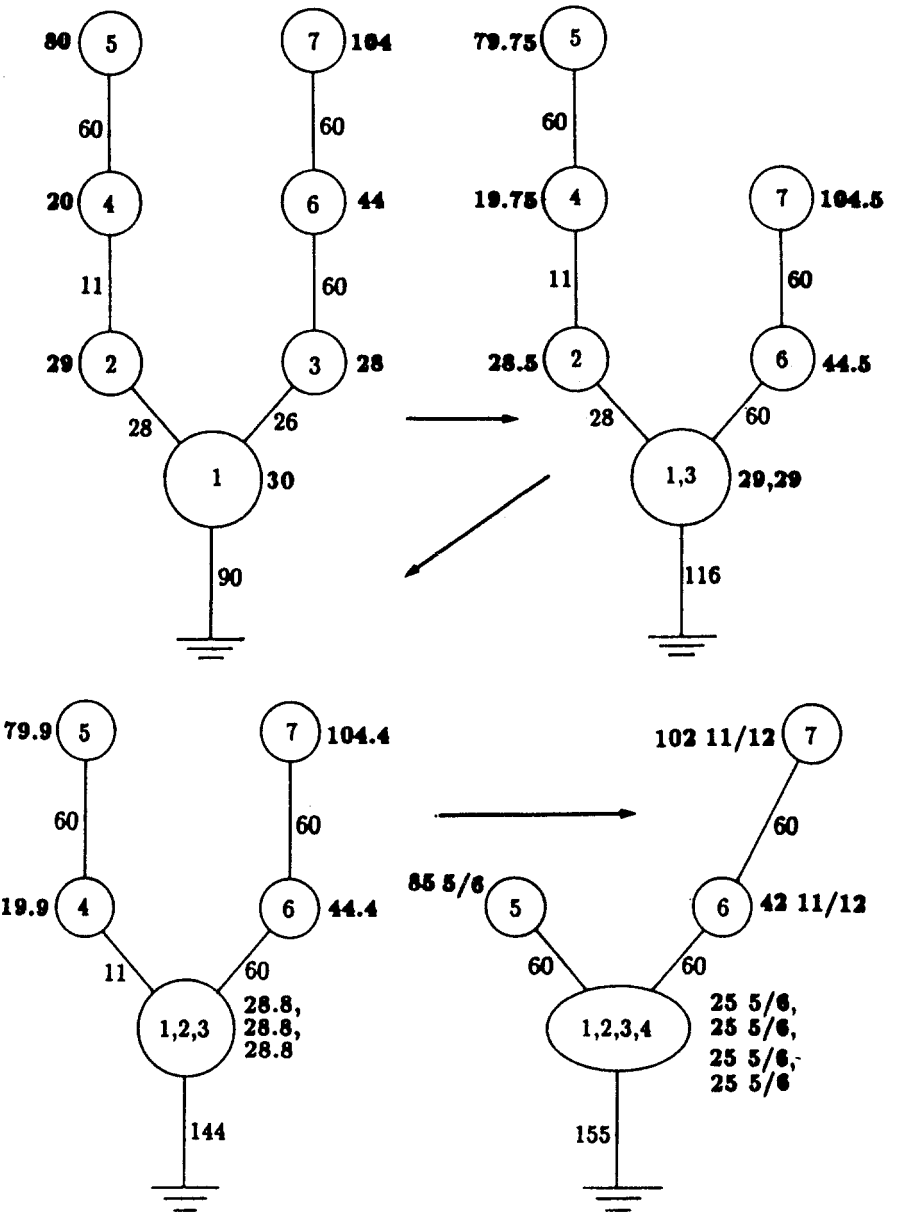

Fig. 3. Condensations that fail

the eliminations of bad arcs in a right order does indeed bring us to the nucleolus.

6 An Algorithm for Computing the Kernel/Nucleolus of a Standard Tree Game

In this section we characterize those arcs that should be eliminated in order to reach the nucleolus. We empoly our characterization to construct an algorithm for computing the nucleolus, which is different from the one constructed in Megiddo [1978]. The characterization and the algorithm shed more insight into the structure of the nucleolus.

Unlike many other algorithms, including Megiddo [1978], that compute the nucleolus of a game via successive minimizations of highest excesses (maximizations of lowest excesses in a cost game), see e.g. Maschler [1992], our algorithm does nothing of this kind. It starts with the proto-nucleolus, which is often a kind of approximation of the nucleolus, and it generates k successive cost allocation vectors, $k \leq n$, the last one being the nucleolus.

As has been shown in the previous section, it all boils down to the question which players should be condensed. To make the right decision we shall have to study more deeply the process of condensation. The following well-known lemma will prove useful.

Lemma 6.1: If $b_i > 0$, $i = 1, 2, \ldots, t$ then

$$\frac{a_1 + a_2 + \cdots + a_t}{b_1 + b_2 + \cdots + b_t}$$

is a weighted average of the terms $r_i = (a_i/b_i)$; the weights being $b_i / \sum_{j=1}^{t} b_j$, $i = 1, 2, \ldots, t$.

Note that the weights are all positive, so, if $\min_{i=1}^{t} r_i < \max_{i=1}^{t} r_i$, the average is strictly between the minimum and the maximum.

Notation 6.2: For adjacent players (i, j), we denote by h_j the expression

$$h_j = \frac{a(e_{ij})}{dv^j - 1}, \tag{6.1}$$

(see Notation 4.2 for the definition of dv).

Note that h_j is not defined if v^j is an endpoint occupied by a single player; otherwise the denominator in (6.1) is a positive integer.

Lemma 6.3: Let x be the proto-nucleolus of a standard tree enterprise. Let (i, j) be a pair of adjacent players, then x_j is a weighted average of x_i and h_j, whenever h_j is defined.

Proof: By the construction of the proto-nucleolus,

$$x_j = \frac{x_i + a(e_{ij})}{dv^j}$$

$$= \frac{x_i + a(e_{ij})}{1 + (dv^j - 1)},$$

which is the average of $x_i/1$ and h_j (Lemma 6.1). □

Lemma 6.4: Let l be a successor of player i, who is not a single residend of an endpoint (so that h_l is well defined). Let x be the proto-nucleolus of the standard tree enterprise. Then, x_l is a weighted average of the numbers x_i and the h_q's, where q is taken over all vertices intermediate between i and l, including l, but not including i.

Proof: By induction on the number of arcs from i to l. The case of one arc was proved in Lemma 6.3. The passage from $t-1$ arcs to t arcs follows from the fact that a weighted average of weighted averages is a weighted average. □

Based on the previous lemma, we now show that the payments in the nucleolus are weighted averages of known quantities. Again, we shall be concerned with a standard tree enterprise $\mathscr{E} = (V, E, a, N)$ and its game $\Gamma_{\mathscr{E}} = (N; c)$, but *we shall assume that each endpoint is occupied by at least two players.* Such enterprises/games will be called enterprises/games of *class* A. For these enterprises h_j is defined for each j. We denote by 1 a player who resides at v^1 – the son of the root.

Lemma 6.5: Let x and z respectively be the proto-nucleolus and the nucleolus of a standard tree game $\Gamma_{\mathscr{E}}$ belonging to class A. Let i be an arbitrary player. With this notation, z_i is a weighted average of x_1 and h_j's, where j is taken over some, possibly all, players who do not reside at v^1.

Proof: By Corollary 4.7, the nucleolus z is the proto-nucleolus of some tree game $\tilde{\mathscr{E}}$, obtained from $\mathscr{E}$ by condensing arcs of a set I, $I \subset E$, where I is the set of pairs of adjacent players that are charged equally at the nucleolus. For an arbitrary player i, consider the vertex $\tilde{v}^i$ in $\tilde{\mathscr{E}}$ and let $\tilde{h}_i$ be the quantity that corresponds to this player. Let $J, J \subseteq I$, be the set of arcs in $\mathscr{E}$ that collapsed to $\tilde{v}^i$. Then

(i) If $\tilde{v}^i$ is not the son of the root, i.e. if $\tilde{v}^i \neq \tilde{v}^1$, then

$$\tilde{h}_i = \frac{a(e(v^i)) + \sum_{e_j \in J} a(e_j)}{dv^i - 1 + \sum_{e_j \in J}(dv_j - 1)},$$

where j is a fixed player in v_j, the tail node of e_j, for each e_j in J. Therefore, $\tilde{h}_i$ is a weighted average of h_i and the h_j's, $j \in J$ (Lemma 6.1).

(ii) If $\tilde{v}^i = \tilde{v}^1$ then

$$\tilde{x}_1 = \frac{a(e(v^1)) + \sum_{e_j \in J} a(e_j)}{dv^1 + \sum_{e_j \in J}(dv_j - 1)},$$

which is a weighted average of x_1 and the h_j's, $j \in J$ (Lemma 6.1).

The proof concludes by referring to Lemma 6.4. □

For the next lemma we adopt the notation:

$$s^x(B_{ij}) := x(B_{ij}) - a(B_{ij}) - x(v^i) = a(T_{ij}) - x(T_{ij}), \tag{6.2}$$

which holds for an imputation x and a pair of adjacent players (i, j).

Lemma 6.6: Let x and z be the proto-nucleolus and the nucleolus, respectively, of a standard tree game $\Gamma_{\mathscr{E}}$ belonging to the class A. For any pair (i, j) of adjacent players,

(1) $z_i \leq s^z(B_{ij})$, and $x_i = s^x(B_{ij})$;
(2) If $z_i < z_j$ then $z_i = s^z(B_{ij})$;
(3) $z_1 \leq x_1$;
(4) If $h_j \geq x_1$ for all j not residing at v^1, then $z_1 = x_1$ and $z_1 = s^z(B_{1j})$ for all j's adjacent to 1.

Proof: (1) is a reformulation of (3.10) ad (4.1). To prove (2) let $\tilde{\mathscr{E}}$ and I be as in the proof of the previous lemma. From $z_i < z_j$ it follows that $(v^i, v^j) \notin I$ (Corollary 4.7) and therefore (i, j) remain adjacent players in $\tilde{\mathscr{E}}$. By (1), above, $z_i = s^z(B_{ij})$, because the condensations leading to $\tilde{\mathscr{E}}$ do not change $s^z(B_{ij})$. We shall prove (3) by contradiction. Suppose $x_1 < z_1$ then, since $x_1 = a(e(v^1))/dv^1$, it follows from (1) that $a(e(v^1)) = x_1 dv^1 < z_1 dv^1 \leq z(v^1) + \sum\{s^z(B_{1j}) : j \text{ adjacent to } 1\} = z(N) - c(N) + a(e(v^1)) = a(e(v^1))$. This is a contradiction. (4) follows from Lemma 6.5. Indeed, z_1 is a weighted average of x_1 and some h_j's, j not residing at v^1. Thus, $h_j \geq x_1$ for all such j's implies that $z_1 \geq x_1$. But, by (3), $z_1 \leq x_1$, so equality holds. To complete the proof, suppose that for some $(1, j)$ in E, $s^z(B_{1,j})$ is not equal to z_1. Then, since $z(N)$ must cover the cost of the tree, $a(e(v^1)) = z(v^1) + \sum\{s^z(B_{1,j}) : j \text{ adjacent to } 1\} > z(v^1) + z_1 \cdot (\text{number of sons of } v^1) = dv^1 z_1 = dv^1 x_1 = a(e(v^1))$, which is a contradiction. □

Lemma 6.7: Let x be the proto-nucleolus of a standard tree game. If arc e_{ij} is a bad arc[24] then $x_i > x_j > h_j$. (The last inequality holds only if h_j is defined.) Conversely, if either $x_i > h_j$, or $x_j > h_j$ for a pair of adjacent players then e_{ij} is

[24] I.e. (i, j) are adjacent players and $x_i > x_j$.

a bad arc. If e_{ij} is a good arc, then $x_i \leq x_j \leq h_j$. (The last inequality holds only if h_j is defined.) Conversely, if (i, j) is a pair of adjacent players and either $x_i \leq h_j$, or $x_j \leq h_j$, then e_{ij} is a good arc.

Proof: We know that x_j is an average of x_i and h_j. If e_{ij} is a bad arc then $x_i > x_j$, hence, $x_j > h_j$. Conversely, if $x_i > h_j$ then $x_i > x_j > h_j$ and the same holds if $x_j > h_j$. In either case e_{ij} is a bad arc.

If e_{ij} is a good arc then $x_i \leq x_j$, so $x_j \leq h_j$. Conversely, if $x_i \leq h_j$ then $x_i \leq x_j \leq h_j$ and the same holds if $x_j \leq h_j$. In either case e_{ij} is a good arc. □

We shall be interested in bad arcs e_{ij} for which h_j is minimal. It will turn out that these should be eliminated.

Lemma 6.8: Let x be the proto-nucleolus of a standard tree game. Suppose

(a) arc e_{ij} is a bad arc,
(b) $h_j \leq h_l$ for all bad arcs e_{pl}.

Then:

(i) If v^k is a vertex for which $x_k \leq h_j$, all successors v^l of v^k satisfy $x_k \leq h_l$, provided that h_l is defined.
(ii) If v^k a vertex for which $x_k \geq h_j$, all successors v^l of v^k satisfy $h_l \geq h_j$, provided that h_l is defined.

Proof:

(i) By contradiction. Suppose $h_l < x_k$ for some successor l of k. Without loss of generality we assume that l is a first such successor; i.e.

$$h_q \geq x_k \quad \text{for all } q \text{ strictly intermediate between } k \text{ and } l. \tag{6.3}$$

Let p be the father of l. If $p = k$ then $x_p = x_k$. If $p \neq k$ then, by Lemma 6.4, x_p is a weighted average of x_k and the h_q's, for q intermediate between k and p. Thus, $x_p \geq x_k$, because of (6.3). In both cases, therefore, $x_p \geq x_k$. But $x_k > h_l$, so $x_p > h_l$ and e_{pl} is a bad arc (Lemma 6.7). Nevertheless, $h_l < x_k \leq h_j$, in contradiction to the minimality of h_j (condition (b)).

(ii) By contradiction. Suppose $h_l < h_j$ for some successor l of k. Again, we can assume that l is a first such player; i.e.

$$h_q \geq h_j \quad \text{for all } q \text{ strictly intermediate between } k \text{ and } l. \tag{6.4}$$

Let p be the father of l. If $p = k$ then $x_p = x_k$. If $p \neq k$ then x_p is a weighted average of x_k and the h_q's, for q intermediate between k and p. Now, $x_k \geq h_j$ and $h_q \geq h_j$, by (6.4), so $x_p \geq h_j$. But then $x_p > h_l$, because $h_j > h_l$, so, by

Lemma 6.7, e_{pl} is a bad arc and again we get a contradiction to the minimality of h_j. □

Notation 6.9: Let (i, j) be a pair of adjacent players in $\mathscr{E}$. Let y be a positive scalar. We denote by $\mathscr{E}^y(B_{ij})$ an enterprise defined on B_{ij}, whose cost function a' satisfies $a'(e(v^i)) = a(e(v^j)) + y$ and $a'(e) = a(e)$ for all other arcs in B_{ij} and whose player set is the set of the original players in B_{ij}, except those residing at v^i. Denote the game of this enterprise by $(B_{ij}, y; c')$. Note that c' is a function of y.

Lemma 6.10: Let z be the nucleolus point of a standard tree enterprise $\mathscr{E}$. Let (i, j) be a pair of adjacent players. The game $(B_{ij}, s^z(B_{ij}); c')$ is the reduced game at z, of the original game, reduced on the player set S consisting of the players in B_{ij} that do not reside at v^i (see footnote 18).

Proof: Let R be a subset of S. Denote by T_R the subtree of B_{ij} for which $c'(R)$ is attained. Then, $c'(R) = a'(T_R) = a(T_R) + s^z(B_{ij}) = a(T_R) + a(T_{ij}) - z(T_{ij}) = c(R \cup T_{ij}) - z(T_{ij}) = \min\{c(R \cup Q) - z(Q): Q \subseteq T_{ij}\}$. The last equality follows from Corollary 3.10 and Lemma 3.11. Indeed, for $Q \subseteq T_{ij}$, $T_{R \cup T_{ij}} \setminus T_{R \cup Q}$, is a union of branches B_{kl}, the surplus of each, at z, is non-negative, so $c(R \cup Q) - z(Q) \geq c(R \cup T_{ij}) - z(T_{ij})$. □

Corollary 6.11: Let (i, j) be a pair of adjacent players in a standard tree enterprise $\mathscr{E}$ and let z be its nucleolus. With this notation, the nucleolus of $\Gamma' := (B_{ij}, s^z(B_{ij}), c')$ is the restriction, z^S, of the nucleolus of $\mathscr{E}$ to the player-set S consisting of the players in B_{ij} that do not reside at v^i.

Proof: Γ_ε is convex; hence, its nucleolus coincides with its prekernel (Maschler, Peleg and Shapley [1972]). By the reduced game property (footnote 20) and Lemma 6.10, z^S is the prekernel of Γ' and, therefore, its nucleolus, because $\mathscr{E}$ is also a convex game. □

Corollary 6.12: Let (i, j) be a pair of adjacent players in a standard tree enterprise $\mathscr{E}$. Let x be the proto-nucleolus of $\mathscr{E}$. With this notation, the proto-nucleolus of (B_{ij}, x_i, c') is the restriction of x to the player-set S consisting of the players in B_{ij} that do not reside at v^i.

Proof: The proof is a direct consequence of Algorithm 4.3. In fact, the computation of the proto-nucleolus of (B_{ij}, x_i, c') proceeds exactly along the same steps as the computation of the proto-nucleolus of $\mathscr{E}$, done after reaching v^i. □

The next theorem will identify an adjacent pair (i, j) for which $z_i = z_j$ in the nucleolus point z.

Theorem 6.13: Let x and z be the proto-nucleolus and the nucleolus of a standard tree enterprise $\mathscr{E}$ of the class A. If (i, j) is a pair of adjacent players such that

a. e_{ij} is a bad arc,
b. h_j is minimal among all h_l, for bad arcs e_{pl},

then $z_i = z_j$.

Proof: Let us assume that the vertices of $\mathscr{E}$ are indexed so that root, $v_1, v_2, \ldots, v_k, v_k = v^i$, is the unique path from the root to vertex v^i. Since e_{ij} is a bad arc, $x_i > x_j > h_j$ (Lemma 6.7).

Consider v_1; if $x_1 < h_j$ then, by Lemma 6.8(i), all successors v_l of vertex v_1 satisfy $x_1 \leq h_l$. Thus, by Lemma 6.6(4), $z_1 = x_1 = s^z(B_{12})$. Now, by Corollaries 6.11 and 6.12, the proto-nucleolus and the nucleolus of $(B_{12}, z_1; c')$ are the restrictions of the proto-nucleolus and the nucleolus of $(N; c)$ to the player set consisting of players in B_{12} that do not reside at v_1, respectively. Thus, in order to prove that $z_i = z_j$, it is sufficient to prove it for the enterprise $(B_{12}, z_1; c')$. If in $(B_{12}, z_1; c')$ it happens that $x_2 < h_j$, we repeat the above arguments to conclude that $z_2 = x_2$ and that we can restrict ourselves to $(B_{23}, z_2; c')$. However, since $x_i > h_j$, after k steps, $k \leq i$, we shall end up with a game $(B_{k(k+1)}, z_k; c')$ in which $x_k \geq h_j$ and for which we have to show that $z_i = z_j$ for its nucleolus point z. Thus, without loss of generality, we can assume that $x_1 \geq h_j$. Then, by Lemma 6.8(ii), it follows that at all successors of v_l, $h_l \geq h_j$. By Lemma 6.5, z_i is a weighted average of x_1, and some h_l's. Since $x_1 \geq h_j$ and $h_l \geq h_j$ at all v_l, we conclude that $z_i \geq h_j$.

Suppose now by contradiction that $z_i \neq z_j$. Then, by Corollary 3.8, $z_i < z_j$, so that by Lemma 6.6(2), $z_i = s^z(B_{ij})$. Consider now the smaller sub-tree game $(B_{ij}, z_i; c')$ and let y be its proto-nucleolus, so

$$y_j = \frac{a(e(v_j)) + z_i}{dv_j - 1 + 1},$$

which is a weighted average of z_i and h_j. Since $z_i \geq h_j$, it follows that $y_j \leq z_i$. Now, by Corollary 6.11, the nucleolus of $(B_{ij}, z_i; c')$ is the restriction of z to the players in B_{ij}, not residing at v_i; therefore, by Lemma 6.6(1), $z_j \leq y_j \leq z_i$ in contradiction to the assumption that $z_i < z_j$. Thus $z_i = z_j$. □

Corollary 6.14: One can remove from Theorem 6.13 the restriction which limits the enterprises to be from class A.

Proof: Suppose $\mathscr{E}$ is such that player k is a sole occupant of an endpoint. Let (p, k) be the pair of adjacent players ending at k, then e_{pk} is not a bad arc, because the cost function is non-negative. Let $\hat{\mathscr{E}}$ be an enterprise obtained from $\mathscr{E}$ by deleting the arc e_{pk} and placing player k at the previous vertex, v^p. We claim that $\hat{\mathscr{E}}$ is strategically equivalent to $\Gamma_{\mathscr{E}}$, because only costs of coalitions containing k were modified by a decrease of $a(e_{pk})$. Since the nucleolus is covariant under strategic equivalence, the two nucleoli are equal, except for z_k that has been lowered by $a(e_{pk})$. Now k is no longer a sole occupant of an endpoint.

Note that neither h_p nor x_p changed during the passage from $\mathscr{E}$ to $\hat{\mathscr{E}}$ because the number of arcs emanating from p decreased by 1, but the number of occupants in v^p increases by 1. We can repeat this procedure for every player who is a sole occupant of an endpoint until finally we obtain a game of class A. If the conditions of Theorem 6.13 hold for the original game, they also hold for the last game so, by the theorem, $z_i = z_j$, because the same equality holds for the last game and player j did not move during the process. □

We are now in a position to provide an algorithm for computing the nucleolus of a standard tree enterprise.

The main algorithm 6.15:

Step 1. Compute the proto-nucleolus. Algorithm 4.3.
Step 2. If there are no bad arcs, stop.
Step 3. If there are bad arcs, choose a bad arc e_{ij}, for which $h_j = a(e_{ij})/(dv^i - 1)$ is minimal and eliminate this arc (ties may be broken arbitrarily) (Definition 4.4).
Step 4. Return to Step 1.

Theorem 6.16: Algorithm 6.15 leads to the nucleolus of the original game.

Proof: At every iteration, the number of arcs is reduced by one, so the algorithm stops after at most n iterations. By Corollary 4.6, we get at each iteration a tree enterprise, whose nucleolus coincides with the nucleolus of the previous tree enterprise. This is true, because we only condense pairs (i, j) of adjacent players for which we know, by Theorem 6.13 and Corollary 6.14, that the players receive equal amounts in the nucleolus of this tree. Consequently, all iterations yield trees with the same nucleolus. The proof then concludes by observing that when the process terminates there are no bad arcs; therefore, the proto-nucleolus of the final tree is equal to its nucleolus (Theorem 4.6). □

7 Shortcuts in Algorithm 6.15. The Case of a Chain Enterprise

Megiddo [1978] provides another algorithm to compute the nucleolus of a standard tree game. His algorithm calls for examining all closed sets with respect to various vertices.[25] In contrast, Algorithm 6.15 examines only bad arcs with respect to the proto-nucleolus. Thus, for example, if there are no bad arcs, the proto-nucleolus is equal to the nucleolus, and since it is computed in O(n) steps,

[25] A set of vertices S is said to be *closed with respect to a vertex* v if it contains v, if every vertex in S is a successor of v and for each vertex v_1 in S, S contains all the vertices on the path from v to v_1.

we have here a fast method to get to the nucleolus. This situation occurs, for example, in a binary tree enterprise in which each vertex is occupied by a single player and the costs are nonincreasing along paths.[26] In many cases there are only few bad arcs and only few condensations are required. In general, there is no need to recompute the proto-nucleolus from the beginning, because many of the entries x_i remain unchanged after a condensation. To identify a bad arc e_{ij} one does not have to compute x_j: It is enough to compare x_i with h_j (Lemma 6.7). Sometimes there is even no need to search for a minimal h_j. One such case is a chain[27] enterprise, first studied in Littlechild [1974], in Littlechild and Thompson [1977] and in Littlechild and Owen [1977] for the airport game.

Algorithm 7.1: Computing the kernel/nucleolus of a chain enterprise. Compute the proto-nucleolus in accordance with Algorithm 4.3. Each time you reach a bad arc eliminate it and if this elimination turns an immediate previous arc a bad one, eliminate it too[28], etc. By the time you have processed the last arc you have arrived at the nucleolus.

Proof: The proof rests on the observation that the order of condensations is not important, as long as one condenses the same players. now, suppose we proceed by Algorithm 6.15, compute the proto-nucleolus and eventually eliminate a bad arc e_{ij}, leaving behind a few other bad arcs. Before any condensation, the proto-nucleolus satisfies

$$x_i = \frac{x_u + a(e_i)}{dv^i} \quad \text{and} \quad x_j = \frac{x_i + a(e_j)}{dv^j}, \tag{7.1}$$

where (u, i) is a pair of adjacent players. After the condensation, the resulting vector $\tilde{x}$ satisfies

$$\tilde{x}_i = \tilde{x}_j = \frac{x_u + a(e^i) + a(e^j)}{dv^i + dv^j - 1} = \frac{x_i(dv^i - 1) + x_j dv^j}{(dv^i - 1) + dv^j} \tag{7.2}$$

and we see that $\tilde{x}_i = \tilde{x}_j$ is a weighted average of x_i and x_j, strictly between them, because $x_i > x_j$, $dv^i \geq 2$ and $dv^j \geq 1$. This means that x_i *decreased*. Thus, if $x_{i-1} < x_i$ then, after condensation it may happen that $x_{i-1} > \tilde{x}_i$; but if $x_{i-1} \geq x_i$ then, after condensation, $x_{i-1} > \tilde{x}_i$. Thus, condensation may turn a previous good arc bad, but it can never turn a previous bad arc good. This feature holds for all previous arcs. thus, working in accordance with Algorithm 6.15, we are sure that eventually we will eliminate all bad arcs prior to v^i (and, perhaps, additional arcs). If this is the case, we might as well eliminate a bad arc as soon as it is

[26] The verification of this fact is straightforward and we leave it to the reader.

[27] A *chain* is a tree whose arcs are located on a single path.

[28] As we did in Example 5.1

encountered, because by doing so we obtain a game having the same nucleolus as the original game (Corollary 4.7). □

Theorem 7.2: Algorithm 7.1 can be performed in $O(n)$ operations, where n is the number of vertices.

Proof: Passing in the upward direction is carried on in n steps, each of which requires a number of calculations which is constant, independent of n[29] (formula (7.1)). Each move in the downward direction involves a single comparison and possibly an elimination of an arc. By (7.2), the number of calculations needed for each elimination is a constant, and at most $n-1$ arcs are eliminated. If there is a comparison without elimination it is done just once, before proceeding in the upward direction. Thus, all calculations are done in linear time in n. □

References

Bird CG (1976) On cost allocation for a spanning tree: A game theoretic approach. Networks 6: 335–350

Claus A, Granot D (1976) Game theory application to cost allocation for a spanning tree. Working paper 402. Faculty of Commerce and Business Administration, University of British Columbia, Vancouver

Galil Z (1980) Applications of efficient mergeable heaps for optimization problems on trees. Acta Informatica 13: 53–58

Granot D, Granot F (1992) Computational complexity of a cost allocation approach to a fixed cost spanning forest problem. Mathematics of Operations Research 17: 765–780

Granot D, Huberman G (1981) Minimum cost spanning tree games. Mathematical Programming 21: 1–18

Granot D, Huberman G (1984) On the core and nucleolus of minimum cost spanning tree games. Mathematical Programming 29: 323–347

Granot D, Maschler M (1994) Spanning network games. Working paper, Faculty of Commerce and Business Administration, The University of British Columbia, Vancouver, BC, Canada.

Littlechild SC (1974) A simple expression for the nucleolus in a special case. International Journal of Game Theory 3: 21–29

Littlechild SC, Owen G (1977) A further note on the nucleolus of the 'airport game'. International Journal of Game Theory 5: 91–95

Littlechild SC, Thompson GF (1977) Aircraft landing fees: A game theory approach. The Bell Journal of Economics 8: 186–204

Maschler M (1992) The bargaining set, kernel, and nucleolus. In: Aumann RJ, Hart S (eds) Handbook of Game Theory. Vol. I. Elsevier Science Publishers B.V. Amsterdam North Holland 591–667

Maschler M, Peleg B (1967) The structure of the kernel of a cooperative game. SIAM Journal of Applied Mathematics 15: 569–604

Maschler M, Peleg B, Shapley LS (1972) The kernel and the bargaining set of convex games. International Journal of Game Theory 1: 73–93

[29] Fractions are maintained by storing separately the nominator and the denominator.

Megiddo N (1978) Computational complexity of the game theory approach to cost allocation for a tree. Mathematics of Operations Research 3: 189–196
Peleg B (1986) On the reduced game property and its converse. International Journal of Game Theory 15: 187–200
Shapley LS (1971) Cores of convex games. International Journal of Game Theory 1: 11–26

Received March 1994
Revised version November 1994