MATHEMATICS OF OPERATIONS RESEARCH
Vol. 21, No. 3, August 1996
Printed in U.S.A.

COMPUTING THE NUCLEOLUS BY SOLVING A PROLONGED SIMPLEX ALGORITHM

JOS A. M. POTTERS, JOHANNES H. REIJNIERSE AND MICHEL ANSING

This paper describes a fast algorithm to find the nucleolus of any game with a nonempty imputation set. It is based on the algorithm scheme of Maschler et al. (1992) for the general nucleolus.

1. Introduction. Besides the Shapley value, the nucleolus is another major single point solution concept for TU-games. Briefly after the introduction by Schmeidler (1969), many attempts are made to find an algorithm to compute the nucleolus. From the beginning it was clear that the algorithm should consist of the solution of a series of linear programs and all solution methods we know of are doing this.

In a—as far as we know unpublished—manuscript of Kopelowitz (1967) we find the first algorithm of this type. The number of linear programs to be solved can increase to 2^{n-1} (in this paper the character n denotes the number of involved players), but in general it is much smaller. Kohlberg (1972) gives one linear program with $\mathcal{O}(n)$ variables, $(2^n)!$ constraints and huge coefficients. Owen (1974) also uses only one linear program; he needs $\mathcal{O}(2^n)$ variables and $\mathcal{O}(4^n)$ constraints. In Maschler, Peleg and Shapley (1979) we find an algorithmic scheme that requires the solution of $\mathcal{O}(4^n)$ linear programs with $\mathcal{O}(2^n)$ rows and columns. In this scheme the coefficients of the linear programs have only the values -1, 0 or 1. Dragan (1981) uses linear programs with only $\mathcal{O}(n)$ rows and $\mathcal{O}(2^n)$ columns. Although he claims to use only $n-1$ programs, we have the opinion that his procedure may use more. In his paper the nucleolus is the allocation that nowadays is called the prenucleolus. More recently, Sankaran (1991) proposed an algorithm based on Maschler, Peleg and Shapley (1979) that uses $\mathcal{O}(2^n)$ iterations. Finally, Solymosi (1993) gives in his dissertation an algorithm that uses at most $n-1$ linear programs with only $\mathcal{O}(n)$ rows and $\mathcal{O}(2^n)$ columns.

Our algorithm is based on an algorithmic scheme of Maschler et al. (1992) for the general nucleolus. It finds the nucleolus after solving at most $n-1$ linear programs with at most 2^n+n-1 rows and 2^n-1 columns. The initial values of the coefficients are -1, 0 or 1. The matrices used in the programs are very scarce: at most $n+1$ columns have more than one nonzero entry. This makes the number of coefficients to be stored of the same order as in the algorithms of Dragan (1981) and Solymosi (1993).

The programs we solve have such a great similarity that we can speak of one *prolonged* simplex algorithm. The term 'prolonged simplex algorithm' means that the usual series of pivot operations (Gauss eliminations) and deletion of elementary rows or columns is interrupted by the introduction of a new variable (that is, a new column). This happens exactly once in each of the linear programs. This answers a question, raised in Dragan (1981), affirmatively.

Received September 6, 1994; revised February 16, 1995.
AMS 1991 subject classification. Primary: 90D12; Secondary: 90C05.
OR/MS Index 1978 subject classification. Primary: Games/Nucleolus; Secondary: Programming/Linear.
Key words. TU-game, nucleolus, simplex method.

0364-765X/96/2103/0757/$01.25

To give an idea about the performance of the algorithm: the computation of the nucleolus in Sankaran's 5-person example requires only 11 pivot operations and approximately 2100 elementary calculations. On a SPARC/SUN/10/41 station it takes only 0.07 seconds of user time.

The outline of this paper is as follows. First we repeat the necessary definitions, in particular we recall the definition of the general nucleolus. We also give the algorithmic scheme of Maschler et al. (1992), that will be the basis of our algorithm. In §3 we transform this algorithmic scheme into an (implementable) algorithm. In §4 we investigate the performance of the algorithm and prove that the algorithm requires only $n - 1$ loops (linear programs). Section 5 gives an explicit calculation in a 3-person example. Finally we show some empirical results and a generalization of the algorithm.

2. Preliminaries. Let (N, v) be a zero-normalized *TU*-game with $v(N) > 0$. Furthermore, we assume that $v(S) \geq 0$ for all coalitions S. This is a harmless assumption as changing negative coalition values into zero does not change the nucleolus of a zero-normalized *TU*-game.

For an allocation $x \in \mathbb{R}^N$ and $S \subseteq N$ we denote $x(S) = \Sigma_{i \in S} x_i$. The *imputation set* of v or $I(v)$ is defined by:

$$I(v) = \{x \in \mathbb{R}^N_+ | x(N) = v(N)\}.$$

For each coalition $S \subseteq N$ we introduce the *excess function* $E_S : I(v) \to \mathbb{R}$ defined by $E_S(x) = v(S) - x(S)$. The *excess map* $E : I(v) \to \mathbb{R}^{2^N}$ is given by $E(x) = \{E_S(x)\}_{S \geq N}$. Furthermore, we need the map $\theta : \mathbb{R}^{2^N} \to \mathbb{R}^{2^n}$ that orders the coordinates of a vector (in $\mathbb{R}^{2^N}$) in a weakly decreasing order. The *nucleolus* is defined to be the set of imputations where the function $\theta \circ E$ restricted to the imputation set attains its lexicographical minimum.

In Schmeidler (1969) it has been proved that the nucleolus of a game (with a nonempty imputation set) contains precisely one element. This element will be denoted by Nu(v) or Nu.

In Maschler et al. (1992) the *general nucleolus* is defined along the same lines. Let $S \subseteq 2^n \setminus \{\phi, N\}$ be a collection of coalitions and let Π be a nonempty polytope in the imputation set of (N, v). We introduce the excess map $E_S = \{E_S\}_{S \in \mathcal{S}}$ and the coordinate ordering map $\theta_{\mathcal{S}} : \mathbb{R}^{\mathcal{S}} \to \mathbb{R}^{|\mathcal{S}|}$. If $\preceq_{\text{lex}}$ denotes the lexicographical ordering on $\mathbb{R}^{|\mathcal{S}|}$, the general nucleolus Nu($\Pi, \mathcal{S}$) is defined by

$$\text{Nu}(\Pi, \mathcal{S}) = \{x \in \Pi | \theta_{\mathcal{S}} \circ E_{\mathcal{S}}(x) \preceq_{\text{lex}} \theta_{\mathcal{S}} \circ E_{\mathcal{S}}(x') \text{ for all } x' \in \Pi\}.$$

For the special case $\mathcal{S} = \phi$, we define Nu $(\Pi, \phi) = \Pi$. It is easy to see that, if $\Pi = I(v)$ and $\mathcal{S} = 2^N \setminus \{\phi, N\}$, then Nu($\Pi, \mathcal{S}$) equals the nucleolus of (N, v).

In Maschler et al. (1992) one finds—up to minor changes—the following algorithmic scheme. The algorithm we describe in this paper will be an elaboration of this procedure.

Initialization: $\Pi := I(v)$, $\mathcal{S} := 2^N \setminus \{\phi, N\}$.
Step 1: Replace Π by $\{x \in \Pi | \bigvee_{S \in \mathcal{S}} E_S(x)$ is minimal$\}$.
Step 2: Delete from $\mathcal{S}$ a number (at least one) of coalitions with constant excess on Π.
If $\mathcal{S} \neq \phi$, return to step 1.

Compared with algorithm 2.4 in Maschler et al. (1992) we replaced "Remove *all* functions which are constant on Π," by "Remove *at least one* coalition with constant

excess on Π." We made this change because it is easy to find at least one constant excess function (cf. §3), but it may require much effort to find all of them. This minor change does not affect the proof (in Maschler et al. (1992)) of the fact that this algorithm stops and generates the nucleolus.

3. The transformation into an algorithm. The algorithmic scheme at the end of the previous section contains commands like "replace a polytope Π by an other one." A computer can only handle polytopes if they are given in an implementable form. Therefore, we define a *description of a pair* $(\Pi, \mathcal{S})$, where $\Pi \subseteq I(v)$ is a polytope and $\mathcal{S} \subseteq 2^N \setminus \{\phi, N\}$.

DEFINITION. A description of $(\Pi, \mathcal{S})$ is a system of inequalities and equations

$$x \in \mathbb{R}^N_+, \qquad y \in \mathbb{R}^{\mathcal{S}}_+, \qquad Ax + By = d \geq 0,$$

such that:

(1) There exists a solution (x, y) of the system if and only if $x \in \Pi$.

(2) There are q basic variables (q is the number of rows of A and B). In other words, there are q variables that occur in exactly one equation, each in a different one.

(3) For every point $x \in \Pi$, there is exactly one solution (x, y) of the system.

In practice, the variables y_S, $(S \in \mathcal{S})$ measure the difference between the last found maximal excess c and the excess of the current point $x \in \Pi$:

$$y_S = c - (v(S) - x(S)).$$

In the initialization step the constant c will be

$$\max_{S \in \mathcal{S}} v(S) = \max_{x \in I(v)} \max_{S \in \mathcal{S}} E_S(x).$$

We start the translation of the algorithmic scheme with the initialization step.

The initialization. Let $\mathcal{S} = 2^N \setminus \{\phi, N\}$ and let $\Pi = I(v)$. Let $c = \max_{S \in \mathcal{S}} v(S)$ and let $i' \in N$ be an arbitrary player. We use the following description of $(\Pi, \mathcal{S})$:

$$x \in \mathbb{R}^N_+, \quad y \in \mathbb{R}^{\mathcal{S}}_+$$

$$x(N) = v(N)$$

$$y_S - x(S) = c - v(S) \qquad (S \in \mathcal{S}, i' \notin S)$$

$$y_S + x(N \setminus S) = c + v(N) - v(S) \qquad (S \in \mathcal{S}, i' \in S).$$

Notice that the variables $\{y_S\}_{S \in \mathcal{S}}$ and the variable $x_{i'}$ occur in exactly one equation. Also the other conditions for a description of $(\Pi, \mathcal{S})$ are satisfied.

Step 1. In this step we replace the description of the current pair $(\Pi, \mathcal{S})$ into a description of the new pair $(\Pi', \mathcal{S})$ where $\Pi' := \{x \in \Pi | \vee_{S \in \mathcal{S}} E_S(x) \text{ is minimal}\} = \{x \in \Pi | \wedge_{S \in \mathcal{S}} y_S(x) \text{ is maximal}\}$. Notice that y_S can be seen as an affine function of x by condition (3) of a description.

Let $x, y \geq 0$, $Ax + By = d$ be the current description of $(\Pi, \mathscr{S})$. To find the subset of Π where $\bigwedge_{S \in \mathscr{S}} y_S$ is maximal, we solve the linear program:

$$\text{maximize } t \text{ under the conditions}$$

$$x \geq 0, \quad t \geq 0$$

$$Ax + By = d$$

$$t \leq y_S \qquad (S \in \mathscr{S}).$$

At this point a new variable $t \geq 0$ (and thus a new column) is introduced. To save a large number of slack variables and equations, we substitute $z_S + t := y_S$ and get the following problem:

$$\text{maximize } t \text{ under the conditions}$$

$$x \geq 0, \quad z \geq 0, \quad t \geq 0$$

$$Ax + Bz + (Be_{\mathscr{S}})t = d,$$

in which $e_{\mathscr{S}} \in \mathbb{R}^{\mathscr{S}}$ is the indicator vector of $\mathscr{S}$.

We solve the linear program with a slight variation of the simplex method (see e.g. Nemhauser and Wolsey (1988, pp. 30–41)). First we bring the t-variable into the basis (by one pivot operation). The current tableau has a basis and the t-column is a basis column. One row, say the jth row, contains the variable t. When we start the maximization of t, we firstly eliminate the t-variable from the objective function by subtracting the jth equation. After this, we pivot in columns in which the objective function has a positive entry. Notice however that the objective function and the jth row have in each column coefficients that add up to zero. Therefore, if the objective function has a positive entry in some column, the jth row has a negative entry in this column. This means that the pivot is never in the jth row and that the t-variable remains in the basis. Hence, we might as well omit the objective function: removing positive entries from the objective function is the same as removing negative entries from the jth row. As the objective function attains its maximum and the simplex method finds a maximum in finitely many steps, we can also remove the negative coefficients from the jth row in (the same) finitely many steps.

Suppose that after the maximization of t the jth equation is

$$\sum A_{ji} x_i + \sum B_{jS} z_S + t = d_j.$$

As we still have a basis (corresponding with a feasible point (x, z, t) with $t = d_j$), we have another description of the pair $(\Pi, \mathscr{S})$. Since the jth equation has only nonnegative coefficients, d_j is the highest value that t can attain under the condition that $x \in \Pi$. Furthermore, all variables with a positive coefficient in the jth equation (except for t) are bound to vanish in order that t attains its maximal value. So, we replace the jth equation by the *set of equations* (splitting the jth equation)

$$x_i = 0 \quad \text{for all } i \text{ with } A_{ji} > 0,$$

$$z_S = 0 \quad \text{for all } S \text{ with } B_{jS} > 0,$$

$$t = d_j =: \hat{t}.$$

If we replace the coefficients of these variables in the other equations by 0 (this can be understood as a very elementary pivot operation), we have a basis again. After these actions we have obtained a system of linear equations with the property:

The point (x, z, t) is a solution of the system if and only if $x \in \Pi$, $t = \hat{t}$ is maximal and $z_S = (c - \hat{t}) - E_S(x)$ for all $S \in \mathcal{S}$.

If we delete the equation $t = \hat{t}$ and rename the variables z_S by y_S $(S \in \mathcal{S})$, we have a description of the pair $(\Pi', \mathcal{S})$. Note that y_S now measures the difference between the (new) current highest excess $c - \hat{t}$ and $E_S(x)$. Hence, we have finished step 1.

Step 2. In this step we delete coalitions that *are found* to have constant excess functions on the newly constructed set Π. As we only have the present description at our disposal, the current system of equations must *show* that one or more excess functions are constant. Therefore, we introduce

DEFINITION. The kth equation (row) is called *elementary* if there is a unique coalition $S \in \mathcal{S}$ such that $B_{kS} \neq 0$ and $A_{ki} = B_{kS'} = 0$ $\forall S' \neq S$ and $i \in N$. We call this coalition S the *index* of row k. So, an elementary equation looks like $y_S = d_k$.

LEMMA 1. *After step* 1, *the tableau contains at least one elementary row.*

PROOF. Immediately after the substitution $z_S + t := y_S$ we have the set of equations $Ax + Bz + (Be_{\mathcal{S}})t = d$. For each row in this set the sum of the coefficients of the z-variables equals the coefficient of t and *this property is not affected by pivot operations*. Therefore, after the optimization, the jth row still obeys the property. Hence, $\sum B_{jS} = 1$ and there is at least one $S \in \mathcal{S}$ with $B_{jS} > 0$. Each of these variables occurs in an elementary row after splitting the equation $\sum A_{ji}x_i + \sum B_{jS}z_S + t = d_j$ in the previous step. □

If row k is elementary, then z_S is constant over Π, where S is the index of k. We delete S from $\mathcal{S}$ and remove row k and the column corresponding to y_S. Note that after this action there is still a basis; we remove each time one *basis* column (variable) and one row containing this variable. This is all we do in step 2.

Summarizing, we have the following algorithm:

Initialization. Let (N, v) be a zero-normalized *TU*-game with $v(N) > 0$ and $v(S) \geq 0$ for all $S \subseteq N$. Let $\Pi = I(v)$ and $\mathcal{S} = 2^N \setminus \{\phi, N\}$. Furthermore, let $c = \max_{S \in \mathcal{S}} v(S)$ and take $i' \in N$. Let $(\Pi, \mathcal{S})$ be described by:

$$x \in \mathbb{R}^N_+, \quad y \in \mathbb{R}^{\mathcal{S}}_+$$

$$x(N) = v(N)$$

$$y_S + x(N \setminus S) = v(N) + c - v(S) \qquad (i' \in S \in \mathcal{S})$$

$$y_S - x(S) = c - v(S) \qquad (i' \notin S \in \mathcal{S})$$

while $\mathcal{S} \neq \phi$ **do.**

Let $Ax + By = d$ be the present description of $(\Pi, \mathcal{S})$.

(a) Maximize t under the conditions

$$t \geq 0, \quad x \geq 0, \quad y \geq 0$$

$$Ax + By + (Be_{\mathcal{S}})t = d;$$

Start with a pivot operation in the t-column;

(b) If $(e_jA)x + (e_jB)y + t = d_j$ is the equation containing t;

(i) For all $S \in \mathcal{S}$ with $B_{jS} > 0$: $\mathcal{S} \leftarrow \mathcal{S} \setminus \{S\}$; remove column S;

(ii) For all $i \in N$ with $A_{ji} > 0$: replace column x_i by the 0-column and add the equation $x_i = 0$;

(iii) Delete row j (which has become "$t = d_j$");

(iv) For all elementary rows k with index S: $\mathcal{S} \leftarrow \mathcal{S} \setminus \{S\}$; remove column S and row k;

(c) The resulting tableau is $Ax + By = d$.

4. Performance of the algorithm. In this section we investigate the complexity of the algorithm of §3. We discuss the following characteristics of the algorithm:

(a) the number of linear programs to be solved,

(b) the sizes of these programs,

(c) the values of the coefficients of the programs,

(d) the total number of pivot steps and the number of elementary calculations in one pivot step.

(a) In this section we will prove that our algorithm terminates after solving at most $n - 1$ linear programs, where n equals the number of players. This is, in fact, the main result of this section. The other algorithm we know of that uses at most $n - 1$ linear programs, the algorithm of Solymosi (1993), obtains this feature by a (not explicitly given) subroutine between each program that "settles" coalitions that are linearly dependent of coalitions with a constant excess. This happens automatically in our algorithm.

(b) The sizes of the linear programs are large; the first program has $2^n - 1$ rows and $2^n + n$ columns (including the column with constraints). Each program has a strictly smaller size than its predecessor. Because we have a basis all the time, most of the columns are unit vectors and only the $(n + 1)(2^n - 1)$ coefficients in the nonbasic columns must be stored individually. Hence, the number of coefficients that has to be stored is of the same order as in the algorithms of Dragan (1981) and Solymosi (1993).

(c) In the beginning all coefficients (outside the constraint column) are -1, 0 or 1.

(d) In general, the number of elementary calculations in one pivot step is of the order of the size of the tableau, that is about 4^n in our case. But, from the scarcity of non-trivial columns we infer that one pivot step requires only $\simeq (n + 2) \cdot 2^n$ elementary calculations. Furthermore, we only have to store at most $n + 1$ nontrivial columns and the places where the basic columns have their one. This gives the process a length of $n \cdot 2^n \cdot p$, where p equals the number of pivot operations. In general, simplex algorithms have a bad worst case performance but a rather good average complexity. We did not investigate if this special class of linear programs behaves better.

In the next section we gather some empirical evidence showing that the number of pivot steps does not increase too dramatically with the number of players. The rest of this section will be used to prove that the solution of at most $n - 1$ linear programs is required for finding the nucleolus.

THEOREM. *The algorithm terminates after solving at most $n - 1$ linear programs.*

The subtlety of the proof comes from the distinction we have to make between variables that *are* constant on the current set Π and the variables that are *detected to be* constant. In fact, we prove that all variables that are constant on the *linear variety* defined by $Ax + By = d$ (without the constraints $x \geq 0$ and $y \geq 0$) are detected by the algorithm. Let $\mathcal{B}$ be the collection of coalitions S that the algorithm has detected

to have constant excess on Π. In the beginning $\mathscr{B}$ consists of the coalitions N and ϕ only. During the algorithm the collection $\mathscr{B}$ increases with at least one coalition in each loop but we have to do better.

Suppose that we are starting a new loop of the algorithm. The present description is $Ax + By = d$, $x \geq 0$ and $y \geq 0$. Let $\mathscr{B}$ be the present collection of coalitions with detected constant excess and let c be the last found maximal excess. The collection $\mathscr{S}$ is the set of coalitions of which the y-variables are still present. Using these data we can introduce two linear varieties Λ and $\Lambda_{\mathscr{B}}$.

The linear variety Λ is the solution of the linear equations $Ax + By = d$ (without the inequalities $x \geq 0$ and $y \geq 0$). At the beginning of the algorithm the linear variety Λ is the set

$$\{(x, y) \in \mathbb{R}^N \times \mathbb{R}^{\mathscr{S}} | x(N) = v(N),\ y_S = c - E_S(x),\ S \in \mathscr{S}\}.$$

It has dimension $n - 1$. The linear variety $\Lambda_{\mathscr{B}}$ is defined by the following linear equations:

$$x \in \mathbb{R}^N,\ y \in \mathbb{R}^{\mathscr{S}}$$

$$x(S) = \mathrm{Nu}(S) \qquad (S \in \mathscr{B})$$

$$y_S = c - (v(S) - x(S)) \qquad (S \in \mathscr{S}).$$

At the start of the algorithm the varieties Λ and $\Lambda_{\mathscr{B}}$ coincide.

CLAIM 1. *Each time a new loop starts the linear varieties Λ and $\Lambda_{\mathscr{B}}$ are the same.*

PROOF. We must prove that, if we start a loop with equality of Λ and $\Lambda_{\mathscr{B}}$, we end the loop with equality, although the linear varieties have changed. The first action in the loop is the substitution $y_S = z_S + t$, $(S \in \mathscr{S})$. If we perform this substitution in both systems of equations we obtain

(1) $$Ax + Bz + (Be_{\mathscr{S}})t = d \quad \text{and}$$

$$x \in \mathbb{R}^N,\quad y \in \mathbb{R}^{\mathscr{S}}$$

(2) $$x(S) = \mathrm{Nu}(S) \qquad (S \in \mathscr{B})$$

$$z_S = (c - t) - (v(S) - x(S)) \qquad (S \in \mathscr{S}).$$

These systems of equations have also the same solution space.

The next actions are pivot operations to maximize t. Pivot operations change the appearance of the equations but not the solution space. When we have found the maximal value $\hat{t}$ and the jth equation (the one containing the variable t) is

$$\sum A_{ji} x_i + \sum B_{jS} z_S + t = \hat{t},$$

we add the equations $x_i = 0$ if $A_{ji} > 0$ and the equations $z_S = 0$ if $B_{jS} > 0$ to the system of linear equations. *We do the same in system* (2). Both systems keep having the same solution spaces. Now we add to $\mathscr{B}$ the coalitions $\{i\}$ if $A_{ji} > 0$ an the coalitions S with $B_{jS} > 0$. So, in system (2) we find $x_i = 0 = Nu_i$ in the first case; we find the equations $z_S = 0$ and (the old equation) $z_S = (c - t) - E_S(x)$ in the second case. Then we find by elimination: $x(S) = v(S) - (c - t)$ and $z_S = 0$ in (2). When we

substitute $t = \hat{t}$, the solution spaces remain the same and the equation(s) $x(S) = v(S) - (c - t)$ become(s) $x(S) = v(S) - (c - \hat{t}) = Nu(S)$, as the nucleolus is a point of the solution space of the present system (2). From the system (1) the by now trivial jth equation is deleted and we are at the end of step 1.

In step 2 we delete the elementary equations of the form $z_S = d_k$. Among these equations are the equations $z_S = 0$ we added to the system. In system (2) we skip these variables too and change thereby the collection $\mathscr{S}$. If we *find* other elementary equations $z_S = d_k$, we know that the variable z_S is constant in the solution space of (2). As we also have the equality $z_S = (c - \hat{t}) - E_S(x)$, we find that $x(S) = d_k - (c - \hat{t}) + v(S)$ holds on the solution space. This constant value must be $Nu(S)$. Therefore, at the end of the loop we have new collections $\mathscr{B}$ and $\mathscr{S}$ but still we have the equality $\Lambda = \Lambda_{\mathscr{B}}$. Note that also c obtains a new value $(c - \hat{t})$. □

The proof of the theorem is based on the fact that, in each loop, *all* coalitions S with constant excess on $\Lambda_{\mathscr{B}}$ (*not on* Π) are detected by an elementary equation. The excess function E_T is certainly constant on $\Lambda_{\mathscr{B}}$ if the characteristic vector of T is a linear combination of the characteristic vectors of coalitions in $\mathscr{B}$. Therefore we define $\mathrm{cl}(\mathscr{B})$: If $\mathscr{B}$ is a nonempty collection of coalitions, $\mathrm{cl}(\mathscr{B})$ consists of the coalitions T for which there is a linear expression $e_T = \sum_{S \in \mathscr{B}} \lambda_S e_S$.

CLAIM 2. *At the end of every loop* $\mathscr{B} = \mathrm{cl}(\mathscr{B})$.

PROOF. Suppose that, at the start of a loop, we have $\mathscr{B} = \mathrm{cl}(\mathscr{B})$. The present linear variety $\Lambda_{\mathscr{B}}$ is the solution of the system of linear equations

$$Ax + Bz + (Be_{\mathscr{S}})t = d \tag{1}$$

and also of the system of linear equations

$$\begin{aligned} &x \in \mathbb{R}^N, \quad y \in \mathbb{R}^{\mathscr{S}} \\ &x(S) = \mathrm{Nu}(S) \qquad (S \in \mathscr{B}) \\ &z_S = (c - t) - (v(S) - x(S)) \qquad (S \in \mathscr{S}). \end{aligned} \tag{2}$$

The first change of $\mathscr{B}$ occurs at the moment we split up the jth equation. If x_i has a positive coefficient in this equation we add the coalition $\{i\}$ to $\mathscr{B}$; if y_S has a positive coefficient, we add S to $\mathscr{B}$. Let $\mathscr{B}'$ be the extended collection. We have to prove that, for coalitions $T \in \mathrm{cl}(\mathscr{B}') \setminus \mathscr{B}$, there is an elementary equation of the form $z_T = d$ in the present set of equations $Ax + Bz = d$.

If $T \in \mathscr{B}' \setminus \mathscr{B}$, we made an equation $z_T = 0$ (already of the right type) or an equation $x_i = 0$. If we have $x_i = 0$, we find in system (2) also the equation $z_{\{i\}} = (c - \hat{t}) + x_i$ and therefore, that $z_{\{i\}}$ is constant on $\Lambda_{\mathscr{B}'}$. If $T \in \mathrm{cl}(\mathscr{B}') \setminus \mathscr{B}'$, we have

$$x(T) = \sum_{S \in \mathscr{B}'} \lambda_S x(S) = \sum_{S \in \mathscr{B}'} \lambda_S \mathrm{Nu}(S)$$

and this linear function is constant on $\Lambda_{\mathscr{B}'}$. From the equation $z_T = (c - \hat{t}) - (v(T) - x(T))$ of (2) we infer that z_T is constant on the linear variety $\Lambda_{\mathscr{B}'}$. Then the equations $z_{\{i\}} = \textit{constant}$ and the equations $z_T = \textit{constant}$ are linear combinations of the equations in $Ax + Bz = d$. As this system has a basis, every nontrivial linear combination of the equations contains as many basic variables as these are nonzero coefficients in the linear combination. Hence, to obtain the equation $z_{\{i\}} = d$ or $z_T = d$, there must be an equation of this form among the equations in $Ax + Bz = d$.

The variables $z_{\{i\}}$ and z_S occur in an elementary equation and are *detected to be constant*. □

PROOF OF THE THEOREM. The variable z_S that is detected to be constant when splitting the equation containing t, was not constant on the previous linear variety. If so, the equation $y_S = k$ would have been a linear combination of the equations in the previous system (2). This would have been detected in the previous loop. Therefore, the dimension of $\Lambda_{\mathscr{B}}$ decreases within each loop of the algorithm. This means that $\Lambda_{\mathscr{B}}$ is zero-dimensional after at most $n - 1$ loops. Because $\Pi \subseteq \Lambda$, the set Π also consists of one point after at most $n - 1$ loops. □

5. An example of the algorithm. Let $N = \{1, 2, 3\}$, $v(N) = 9$, $v(12) = v(13) = 5$, $v(23) = 1$ and $v(i) = 0$ for all $i \in N$. Then $c = \max_{S \neq N} v(S) = 5$. We took $i' = 3$. After introducing the t-column, the tableau is:

$$\left[\begin{array}{cccccccccc|c} t & y_{12} & y_{13} & y_{23} & y_1 & y_2 & y_3 & x_1 & x_2 & x_3 & d \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 9 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 9 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 13 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 5 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 5 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 14 \end{array}\right].$$

The first pivot is in the t-column. This must be in the second row:

$$\left[\begin{array}{cccccccccc|c} t & y_{12} & y_{13} & y_{23} & y_1 & y_2 & y_3 & x_1 & x_2 & x_3 & d \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 9 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 9 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 & 2 & 1 & 0 & 13 \\ 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 5 \\ 0 & -1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 5 \\ 0 & -1 & 0 & 0 & 0 & 0 & 1 & 2 & 2 & 0 & 14 \end{array}\right].$$

For the second pivot we choose a column with negative second coordinate, for instance the column corresponding to x_1. We pivot with the sixth row:

$$\left[\begin{array}{cccccccccc|c} t & y_{12} & y_{13} & y_{23} & y_1 & y_2 & y_3 & x_1 & x_2 & x_3 & d \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 1 & 4 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 5 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 2 & 0 & 4 \\ 0 & 1 & 0 & 1 & 0 & -2 & 0 & 0 & 1 & 0 & 3 \\ 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 5 \\ 0 & -1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 5 \\ 0 & 1 & 0 & 0 & 0 & -2 & 1 & 0 & 2 & 0 & 4 \end{array}\right].$$

The third pivot must be in the column corresponding to x_2, for instance with the last row:

$$\begin{array}{c|cccccccccc} & t & y_{12} & y_{13} & y_{23} & y_1 & y_2 & y_3 & x_1 & x_2 & x_3 & d \\ \hline & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 1 & 2 \\ & 1 & \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 7 \\ & 0 & -1 & 1 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ & 0 & \frac{1}{2} & 0 & 1 & 0 & -1 & -\frac{1}{2} & 0 & 0 & 0 & 1 \\ & 0 & -1\frac{1}{2} & 0 & 0 & 1 & 1 & -\frac{1}{2} & 0 & 0 & 0 & 3 \\ & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 5 \\ & 0 & \frac{1}{2} & 0 & 0 & 0 & -1 & \frac{1}{2} & 0 & 1 & 0 & 2 \end{array}.$$

At this moment there are no negative coefficients in the second row anymore, we finished part (a) of the algorithm for the first time. The collection $\mathscr{S}$ becomes $\{(13), (23), (1), (2)\}$. We remove the t-column, the y_{12}- and y_3-columns and the second row. There are no elementary rows. The tableau has become:

$$\begin{array}{c|ccccccc} & y_{13} & y_{23} & y_1 & y_2 & x_1 & x_2 & x_3 & d \\ \hline & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 \\ & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 1 \\ & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 3 \\ & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 5 \\ & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 2 \end{array}.$$

At this stage it occurs that the algorithm has not detected yet that, e.g., y_2 is constant over Π. Hence, the dimension of Λ is strictly larger than the dimension of Π. This completes step 2 and a new loop is started. We get the new t-column by summing up in each row the sum of the coefficients of the y-variables:

$$\begin{array}{c|ccccccccc} & t & y_{13} & y_{23} & y_1 & y_2 & x_1 & x_2 & x_3 & d \\ \hline & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 \\ & 2 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ & 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 1 \\ & 2 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 3 \\ & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 5 \\ & -1 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 2 \end{array}.$$

Pivot with the second row:

$$\begin{array}{c|ccccccccc} & t & y_{13} & y_{23} & y_1 & y_2 & x_1 & x_2 & x_3 & d \\ \hline & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 \\ & 1 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 \\ & 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 1 \\ & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 3 \\ & 0 & -\frac{1}{2} & 0 & 0 & \frac{1}{2} & 1 & 0 & 0 & 5 \\ & 0 & \frac{1}{2} & 0 & 0 & -\frac{1}{2} & 0 & 1 & 0 & 2 \end{array}.$$

Because there are no negative numbers in the second row, we are on an optimal point. After deleting the columns corresponding to y_{13} and y_2, all rows become elementary. The algorithm terminates. We derive from the tableau that $\mathrm{Nu}(v) = (5, 2, 2)$. The total number of pivot steps is 4.

6. Final comments.

Empirical results. To give an idea of the performance of the algorithm in practice, we give some empirical results. We defined the test games in the following way:

$v(i) = 0$ for all $i \in N$,

for all $S \subsetneq N$, $|S| \geq 2$, $v(S)$ is an arbitrary natural number between 1 and $100|S|$,

$v(N)$ is an arbitrary natural number between $100(n-2)$ and $100n$.

This list shows the results:

number of players:	3	4	5	6	7	8	9	10
number of tested games:	10	10	10	10	10	10	10	10
average number of pivots:	4	6.8	9.0	11.7	16.6	20.9	25.8	32.9
maximal number of pivots:	4	9	10	14	20	27	30	38
average number of programs:	1.4	2.3	2.3	2.2	2.8	1.9	1.9	1.9
maximal number of programs:	2	3	4	4	4	3	3	4
average user time in seconds:	0.04	0.05	0.06	0.10	0.21	0.56	1.67	5.27
maximal user time in seconds:	0.06	0.08	0.09	0.12	0.26	0.62	1.89	6.04

The general nucleolus. With minor changes the algorithm in this paper can be used for the calculation of the general nucleolus $\mathrm{Nu}(\Pi, \mathscr{S})$. Here, $\mathscr{S}$ can be any collection of coalitions of a zero-normalized game (N, v) and Π can be any nonempty polyhedral subset of the imputation set. Only the initialization has to be changed. Suppose the set Π is given by a system of inequalities: $\Pi = \{x \in \mathbb{R}^N_+ | Px \leq f\}$, in which P is a matrix with, say, p rows and n columns and f is an element of $\mathbb{R}^p$. First, we introduce slack-variables $z_k \geq 0$ $(k \leq p)$ and get a table of equalities: $Px + z = f$. The table has maximal rank p. Equations $e_k Px + z_k = f_k$ with $f_k < 0$ are multiplied by -1. We perform pivot operations until we have a basis (if $f \geq 0$, this is immediately the case). We get a system of the following form:

$$Px + Qz = f \geq 0, \quad x \in \mathbb{R}^N_+, \quad z \in \mathbb{R}^p_+,$$

in which p variables (forming the basis) occur in exactly one equation, each one in a different equation.

At this point we introduce the variables y_S $(S \in \mathscr{S})$ and the equations concerning the excesses of the coalitions in $\mathscr{S}$:

$$y_S - x(S) = c - v(S) \qquad (S \in \mathscr{S}).$$

To preserve a basis we eliminate the x-variables that occur in the basis of $Px + Qz = f$ from the equations $y_S - x(S) = c - v(S)$. For each basic variable x_i, we add the equation in $Px + Qz = f$ containing the x_i-variable to the equations $y_S - x(S) = c - v(S)$ with $i \in S$. We obtain a description of $(\Pi, \mathscr{S})$ with the sole difference that we have more slack variables than just the y-variables, namely the z-variables. Perform the algorithm as long as $\mathscr{S}$ is nonempty. The resulting system of equalities describes $\mathrm{Nu}(\Pi, \mathscr{S})$. Notice that the number of columns that has to be stored explicitly is still at most $n + 1$. We think that this algorithm uses at most $n - 1$ steps too.

References

Dragan, I. (1981). A procedure for finding the nucleolus of a cooperative n person game. *Z. Oper. Res.* **25** 119–131.

Kohlberg, E. (1972). The nucleolus as solution of a minimization problem. *SIAM J. Appl. Math.* **23** 34–39.

Kuipers, J. (1994). *Combinatorial methods in cooperative game theory*. Ph.D. Thesis, Maastricht.

Maschler, M., B. Peleg and L. Shapley (1979). Geometric properties of the kernel, nucleolus, and related solution concepts. *Math. Oper. Res.* **4** 303–338.

____, J. Potters and S. Tijs (1992). The general nucleolus and the reduced game property. *Internat. J. Game Theory* **21** 83–106.

Nemhauser, G. and L. Wolsey (1988). *Integer and Combinatorial Optimization.* John Wiley & Sons, New York.

Owen, G. (1974). A note on the nucleolus. *Internat. J. Game Theory* **3** 101–103.

Sankaran, J. (1991). On finding the nucleolus of an n-person cooperative game. *Internat. J. Game Theory* **19** 329–338.

Schmeidler, D. (1969). The nucleolus of a characteristic function game. *SIAM J. Appl. Math.* **17** 1163–1170.

Solymosi, T. (1993). *On computing the nucleolus of cooperative games*. Ph.D. Thesis, University of Illinois at Chicago.

____ and T. Raghavan (1994). An algorithm for finding the nucleolus of assignment games. *Internat. J. Game Theory* **23** 119–143.

J. A. M. Potters, J. H. Reijnierse and M. Ansing: Department of Mathematics, University of Nijmegen, Toernooiveld, 6525 ED Nijmegen, The Netherlands