Math. Meth. Oper. Res. (2006) 63: 427–433
DOI 10.1007/s00186-006-0065-5

ORIGINAL ARTICLE

Ulrich Faigle · Walter Kern · Jeroen Kuipers

Computing an element in the lexicographic kernel of a game

Received: 9 March 2005 / Accepted: 12 July 2005 / Published online: 25 May 2006

Abstract The lexicographic kernel of a game lexicographically maximizes the surplusses s_{ij} (rather than the excesses as would the nucleolus) and is contained in both the least core and the kernel. We show that an element in the lexicographic kernel can be computed efficiently, provided we can efficiently compute the surplusses $s_{ij}(x)$ corresponding to a given allocation x. This approach improves the results in Faigle et al. (in Int J Game Theory 30:79–98, 2001) and allows us to determine a kernel element without appealing to Maschler transfers in the execution of the algorithm.

Keywords Kernel · Lexicographic · Computational complexity

AMS Classification 90C27 · 90D12

1 Introduction

The *kernel* of a cooperative game, introduced by Davis and Maschler (1965), is a perhaps less intuitive solution concept than Schmeidler's (1969) *nucleolus*. Yet, it is

We thank the referees for clarifying the presentation of the proof of Proposition 2.1.

U. Faigle (✉)
Zentrum für Angewandte Informatik, Universität zu Köln,
Weyertal 80, 50931 Koln, Germany
E-mail: faigle@zpr.uni-koeln.de

W. Kern
Faculty of Electrical Engineering, Mathematics and Computer Science, University of Twente,
P.O. Box 217, 7500 AE, Enschede, Netherlands

J. Kuipers
Department of Mathematics, Maastricht University,
P.O. Box 616, 6200 MD, Maastricht, Netherlands

closely related to the idea of the nucleolus. In fact, the nucleolus is a special element in the kernel, which lexicographically maximizes the excesses of a game, while the kernel generally tries to balance the surplusses (namely the relative excesses) of players against one another. Thus G. Kalai (see Ex. 4.9 in Maschler and Peleg 1976; Yarom 1981) suggested to combine the characteristic features of the kernel and the nucleolus and study the *lexicographic kernel*, which lexicographically maximizes the vector of surplusses.

It is interesting to note, however, that the lexicographic kernel does not necessarily contain the nucleolus, (cf. Maschler and Peleg 1976; Yarom 1981). On the other hand, there are many games whose nucleolus coincides with the lexicographic kernel (see, e.g. Faigle et al. 2001). This motivates us to investigate the computational aspects of the lexicographic kernel in its own right.

The lexicographic kernel is a polytope which (similar to the nucleolus) is contained in the intersection of the least core with the kernel. In contrast to the nucleolus, it is a "geometrical locus" (Yarom 1981) in the sense that it is completely determined by the (least) core as a subset of the euclidian space. Yarom (1981) investigates continuity aspects of this solution concept and presents a bargaining procedure converging to the lexicographic kernel.

The purpose of our work here is to present an algorithm that computes an element in the lexicographic kernel (in a finite number of steps). Our approach is closely related to the one we used in Faigle et al. (2001) for finding an element in the kernel. Under weak assumptions, concerning the efficient computability of the surplusses, the algorithm is efficient (i.e. has polynomial running time). The algorithm presented here offers a substantial improvement with respect to the algorithm (and its analysis) derived in Faigle et al. (2001): As the lexicographic kernel is a subset of the kernel, we are now able to completely eliminate Maschler's *transfer steps* (cf. Stearns 1968) for the computation of a kernel element.

2 Basic definitions

We consider *(cooperative) games* (N, c), where $N = \{1, \ldots, n\}$ is the set of players and $c : 2^N \to \mathbb{R}$ a *cost function*, assigning a cost $c(S)$ to every *coalition* $S \subseteq N$. We assume throughout that $c(\emptyset) = 0$.

Remark *Mutatis mutandis*, our results below hold of course also for cooperative profit games. We prefer the cost model for its intuitive appeal.

An (efficient) *allocation* is a vector $x \in \mathbb{R}^n$ satisfying $x(N) = c(N)$, where we use the standard shorthand notation

$$x(S) = \sum_{i \in S} x_i \qquad \text{for all } S \subseteq N.$$

We let $X = \{x \in \mathbb{R}^N \mid x(N) = c(N)\}$ denote the set of allocations. Relative to a given $x \in X$, the *excess* of a coalition $S \subseteq N$ is defined as

$$e(S, x) = c(S) - x(S).$$

The minimum (non-trivial) excess is then given by

$$e_{\min}(x) = \min_{S \neq \emptyset, N} e(S, x).$$

A related notion is the *surplus* $s_{ij}(x)$ of player i against player j, where

$$s_{ij}(x) = \min\{e(S, x) \mid S \subseteq N, i \in S, j \notin S\}.$$

With these notions, we are ready to introduce the following *solution concepts*. For $\varepsilon \in \mathbb{R}$, the ε-*core* is defined as

$$\varepsilon\text{-core}(c) = \{x \in X \mid e_{\min}(x) \geq \varepsilon\}.$$

Thus $\varepsilon = 0$ yields the well-known *core*. If ε is the unique maximum number for which the ε-core is nonempty, we obtain the so-called *least core*.

The *pre-kernel* $\mathcal{K}(c)$ is defined as

$$\mathcal{K}(c) = \{x \in X \mid s_{ij}(x) = s_{ji}(x) \quad \forall i \neq j\}.$$

Relative to any given allocation $x \in X$, let us arrange the surplusses $s_{ij}(x)$ in nondecreasing order to form the $n(n-1)$-dimensional vector $\theta(x)$. The *lexicographic kernel* is defined as the set of all allocations that lexicographically maximize $\theta(x)$ over all allocations, i.e.,

$$\mathcal{K}_{\text{lex}}(c) = \{x \in X \mid \theta(x) \succeq \theta(y) \text{ for all } y \in X\},$$

where $\theta(x) \succeq \theta(y)$ means that $\theta(x)$ is lexicographically larger than or equals $\theta(y)$.

Recall that the *pre-nucleolus* $\eta(c)$ is the unique allocation $x \in X$ that lexicographically maximizes the $(2^n - 2)$-dimensional vector obtained by arranging the non-trivial excesses $e(S, x)$, $\emptyset \neq S \neq N$, in nondecreasing order.

Remark An allocation $x \in X$ is called *individually rational* if $x(i) \leq c(\{i\})$ for all $i \in N$. Restricting oneself to the set

$$X^* = \{x \in X \mid x(i) \leq c(\{i\}) \quad \forall i \in N\}$$

of individually rational allocations, one arrives at slightly modified solution concepts. For example, the *nucleolus* resp. the *lexicographic kernel* is obtained by replacing X with X^* in the definitions above. From our point of view, however, there is no reason (other than tradition) to restrict ourselves to X^*, i.e. to distinguish between singleton coalitions and others a priori. We therefore work with the above "pre-solution concepts". It is straightforward to modify the algorithm we present in Sect. 4 so that it computes elements in the lexicographic kernel.

As is well-known, the (pre-)nucleolus is contained in the intersection of the (pre-)kernel with the least core. A similar relation was observed in Maschler and Peleg (1976) for the lexicographic kernel. To make this note more self-contained, we present it with an explicit proof.

Proposition 2.1 $\mathcal{K}_{\text{lex}}(c) \subseteq \mathcal{K}(c) \cap \textit{least core}\ (c)$.

Proof The inclusion $\mathcal{K}_{\text{lex}}(c) \subseteq$ least core (c), follows directly from the definitions. To prove $\mathcal{K}_{\text{lex}}(c) \subseteq \mathcal{K}(c)$, assume $x \in \mathcal{K}_{\text{lex}}(c)$ and order the surplusses non-decreasingly:

$$s_{i_1 j_1}(x) \leq \cdots \leq s_{i_m j_m}(x).$$

Suppose that $x \notin \mathcal{K}(c)$ holds. So there exists a smallest index k such that the pair $(i, j) = (i_k, j_k)$ satisfies $s_{ij}(x) < s_{ji}(x)$, say, $s_{ij}(x) = s_{ji}(x) - 2\alpha$ for some $\alpha > 0$. We now execute a *transfer* of size α and pass from x to the allocation

$$\overline{x} = x + \alpha e_j - \alpha e_i$$

(with e_i and e_j being the i-th resp. j-th unit vector in $\mathbb{R}^n$). We claim that the α-transfer yields a lexicographically larger vector of surplusses, contradicting our assumption that $x \in \mathcal{K}_{\text{lex}}(c)$.

Indeed, the transfer yields $s_{ij}(\overline{x}) = s_{ij}(x) + \alpha > s_{ij}(x)$. Suppose that nevertheless the claim is false and there exist players ℓ, m, however, such that

$$s_{\ell m}(\overline{x}) \leq s_{ij}(x) \qquad \text{and} \qquad s_{\ell m}(\overline{x}) < s_{\ell m}(x) \ .$$

Letting $\overline{S}$ (with $\ell \in \overline{S}$ and $m \notin \overline{S}$) be such that $s_{\ell m}(\overline{x}) = c(\overline{S}) - \overline{x}(\overline{S})$, we then must have $j \in \overline{S}$ and $i \notin \overline{S}$ and, therefore,

$$s_{ji}(\overline{x}) \leq s_{\ell m}(\overline{x}) \leq s_{ij}(x) = s_{ji}(x) - 2\alpha = s_{ji}(\overline{x}) - \alpha \ ,$$

which is a contradiction. □

3 Our computational model

In principle, a game (N, c) can be described by a complete list of all 2^n cost values $c(S)$, $S \subseteq N$. Relative to this notion of *input size* most computational game-theoretic problems are trivially easy (efficiently solvable). However, measuring the input size this way is often not adequate. For example, in the case of a minimum spanning tree game we are *not* given such a list of 2^n cost values, but rather a weighted graph on $n + 1$ vertices, from which we can easily infer the cost $c(S)$ for any given coalition $S \subseteq N$.

For this reason, a more adequate (and more interesting) model is used (cf. Faigle et al. 2001 for more details and additional motivation, more general background on computational complexity can be found, e.g. in Faigle (2002) or Schrijver (1986). We consider a fixed class $\mathcal{C}$ of games. Each game $(N, c) \in \mathcal{C}$ has a *compact description* in terms of

- The finite set $N = \{1, \ldots, n\}$ of players
- An upper bound $\langle c \rangle$ on the maximum size of a cost value, i.e. $\langle c \rangle \geq \max_{S \subseteq N} \langle c(S) \rangle$.
- An algorithm ("oracle") which, on input $S \subseteq N$, computes the corresponding cost $c(S)$.

(Here, we assume that all costs $c(S)$ are rational numbers. The size $\langle r \rangle$ of a rational number $r = p/q$ is the number of bits necessary to represent p and q in binary.)

We consider *algorithms for the class* $\mathcal{C}$. The input for such an algorithm A is a game $(N, c) \in \mathcal{C}$, presented *via* the player set N, the upper bound $\langle c \rangle$ and access to the oracle for computing the cost values. There may also be additional input such as, e.g., an allocation x of size (encoding length) $\langle x \rangle$. The *running time* of A is measured in terms of the number of elementary (bit) operations plus calls to the

oracle for computing certain c-values. Correspondingly, we say that A is *efficient*, if the number of elementary operations and oracle calls is polynomially bounded in n, $\langle c\rangle$ and $\langle x\rangle$. (See Faigle et al. 2001 for a concrete example.)

For certain classes of games (e.g. minimum spanning tree games, cf. Faigle et al.1997), computing $e_{\min}(x)$ or $s_{ij}(x)$ for a given allocation x is NP-hard. For such classes of games we can hardly expect to be able to compute the nucleolus or elements in the (lexicographic) kernel efficiently. We therefore make the assumption (CCM) below on the computational complexity of minimal surplus computation relative to the class $\mathcal{C}$ of games:

There exists an efficient algorithm A which, on input $(N, c) \in \mathcal{C}$ and allocation $x \in X$, efficiently computes the number $e_{\min}(x)$. (CCM)

As shown in Faigle et al. (2001), this assumption is tantamount to the efficient computability of the surplusses $s_{ij}(x)$. Furthermore, not only the surplusses $s_{ij}(x)$ can be computed efficiently, but we can also identify in polynomial time a coalition $S \subseteq N$ containing i, but not j, with $c(S) - x(S) = s_{ij}(x)$.

Computing $e_{\min}(x)$ can be done efficiently, for example, when c (and hence $c-x$) is submodular (cf. Schrijver 2000). Hence (CCM) holds, for example, for any class of convex games. A concrete example is provided, e.g. by Mediggo's (1987) tree games. There are, however, also non-convex games that satisfy (CCM). An interesting case is, e.g. the class of (non-bipartite) matching games (cf. Kurn and Paulusma 2003).

4 The lexicographic pre-kernel

We consider a fixed class $\mathcal{C}$ of games. The purpose of this section is a proof of our main result:

Theorem 4.1 *If $\mathcal{C}$ satisfies (CCM), the problem of computing an element in the lexicographic pre-kernel is efficiently solvable (for games in $\mathcal{C}$).*

To establish the validity of Theorem 4.1, let $(N, c) \in \mathcal{C}$ be an arbitrary instance of the computational problem. Denoting by I the set of pairs (i, j) of players $i \neq j$, consider the optimization problem

$$\begin{aligned} \varepsilon_1 := \max\ & \varepsilon \\ & s_{ij}(x) \geq \varepsilon, \quad (i, j) \in I \\ & x \in X. \end{aligned} \tag{1}$$

Observe that a constraint $s_{ij}(x) \geq \varepsilon$ actually corresponds to 2^{n-2} linear constraints of the form

$$c(S) - x(S) \geq \varepsilon.$$

So (1) is a linear program and its set of feasible solutions (x, ε) forms a polyhedron $P \subseteq \mathbb{R}^{n+1}$.

Given a vector $(\overline{x}, \overline{\varepsilon}) \in \mathbb{R}^{n+1}$, (CCM) allows us to check efficiently whether $(\overline{x}, \overline{\varepsilon}) \in P$ holds true. Moreover, in case $(\overline{x}, \overline{\varepsilon}) \notin P$, we can efficiently determine

a corresponding *violated inequality*, i.e. a linear inequality from the constraints in (1) that is violated by $(\overline{x}, \overline{\varepsilon})$.

Indeed, assume that, say, $s_{ij}(\overline{x}) < \overline{\varepsilon}$ holds for some $(i, j) \in I$. As pointed out at the end of Sect. 3, (CCM) also allows us to compute efficiently a corresponding coalition $S \subseteq N$ with $i \in S$, $j \notin S$ and $e(S, \overline{x}) = s_{ij}(\overline{x})$. Then

$$c(S) - x(S) \geq \varepsilon$$

is one of the constraints in (1) that is violated by $(\overline{x}, \overline{\varepsilon})$.

This observation, together with standard results on the ellipsoid method (cf. also Faigle et al. 2001), yields an efficient algorithm for solving (1). Note that (1) is feasible and bounded, so optimal solutions exist. (The corresponding optimal ε_1 defines the least core.)

Our next step is to identify the set $I_1 \subseteq I$ of pairs (i, j) for which the constraint $s_{ij}(x) \geq \varepsilon_1$ is necessarily tight whenever (x, ε_1) is an optimal solution of (1). This is straightforward: For each $(i_1, j_1) \in I$, we solve

$$\begin{aligned} \varepsilon_{i_1 j_1} := \quad & \max \varepsilon \\ & s_{i_1 j_1}(x) \geq \varepsilon \\ & s_{ij}(x) \geq \varepsilon_1 \quad (i, j) \in I \backslash \{(i_1, j_1)\} \\ & x \in X. \end{aligned} \tag{2}$$

and include (i_1, j_1) into I_1 if and only if $\varepsilon_{i_1, j_1} = \varepsilon_1$. (Note that $\varepsilon_{i_1, j_1} \geq \varepsilon_1$ holds in general.)

By definition, each $(i_1, j_1) \in I$ thus admits a corresponding $x = x(i_1, j_1)$ such that $s_{ij}(x) \geq \varepsilon_1$, for all (i, j) and $s_{i_1, j_1}(x) = \varepsilon_1$ if and only if $(i_1, j_1) \in I_1$. Taking the average

$$\overline{x} = \frac{1}{|I|} \sum_{(i,j) \in I} x(i, j),$$

we obtain an allocation $\overline{x} \in X$. Due to the concavity of the s_{ij}, this vector $\overline{x}$ solves (1) with $s_{ij}(\overline{x}) \geq \varepsilon_1$ being tight exactly when $(i, j) \in I_1$. So we conclude that indeed I_1 is the set of pairs for which the constraint $s_{ij}(x) \geq \varepsilon$ in (1) is necessarily tight at an optimum solution (x, ε_1) of (1).

Having computed $I_1 \subseteq I$, we proceed to solve

$$\begin{aligned} \varepsilon_2 := \ & \max \varepsilon \\ & s_{ij}(x) \geq \varepsilon_1 \quad (i, j) \in I_1 \\ & s_{ij}(x) \geq \varepsilon \quad (i, j) \in I \backslash I_1 \\ & x \in X. \end{aligned} \tag{3}$$

with optimum $\varepsilon_2 > \varepsilon_1$ and determine a corresponding set I_2 in a similar way. After at most $r = O(n^2)$ iterations, we end with a complete description of the lexicographic pre-kernel

$$\mathcal{K}_{\text{lex}} = \{x \in X \mid s_{ij}(x) \geq \varepsilon_k,\ (i, j) \in I_k,\ k = 1, \ldots, r\}$$

and some $\overline{x} \in \mathcal{K}_{\text{lex}}$ (obtained while computing ε_r and I_r).

There is one problem left. To prove efficiency of our algorithm, we have to analyze the size of the numbers $\varepsilon_1, \ldots, \varepsilon_r$ that we compute iteratively. But this

is easy by using the following a posteriori argument. Relative to the partition $I = I_1 \cup \cdots \cup I_r$ that we have constructed, the values $\varepsilon_1, \ldots, \varepsilon_r$ are uniquely determined by the solution of the following lexicographic maximization problem

$$\begin{array}{lll} \text{lex}-\max & (\varepsilon_1, \ldots, \varepsilon_r) & \\ \text{s.t.} & s_{ij}(x) \geq \varepsilon_k, & (i, j) \in I_k,\ k = 1, \ldots, r \\ & x \in X & \end{array}$$

with $n + r$ variables $x_1, \ldots, x_n, \varepsilon_1, \ldots, \varepsilon_r$. The optimum is attained at a vertex of the feasible set P. Such a vertex has components polynomially bounded in the dimension $n + r = O(n^2)$ and the maximum size of a coefficient in the system of inequalities describing P. Hence, in particular, the optimum values $\varepsilon_1, \ldots, \varepsilon_r$ are polynomially bounded in n and $\langle c \rangle$, as required.

So Theorem 4.1 is proved.

References

1. Davis M, Maschler M (1965) The kernel of a cooperative game. Naval Logistics Quaterly 12 (1965), 223–259
2. Faigle U, Fekete SP, Hochstättler W, Kern W (1997) On the complexity of testing core membership for min cost spanning tree games. Int J Game Theory 26:361–366
3. Faigle U, Kern W, Kuipers J (2001) On the computation of the nucleolus of a cooperative game. Int J Game Theory 30:79–98
4. Faigle U, Kern W, Still G (2002) Algorithmic principles of mathematical programming. Kluwer, Dordrecht
5. Justman M(1977) Iterative processes with "nucleolar" restrictions. Int J Game Theory 6: 189–212
6. Kern W, Paulusma D (2003) Matching games: the least core and the nucleolus. Math Oper Res 28:294–308
7. Maschler M, Peleg B (1975) Stable sets and stable points of set valued dynamic systems with applications to game theory. SIAM J Control Optim 14:985–995
8. Megiddo N (1987) Computational complexity of the game theory approach to cost allocation for a tree. Math Oper Res 3:89–196
9. Schmeidler D (1969) The nucleolus of a characteristic function game. SIAM J Appl Math 17:1163–1170
10. Schrijver A (1986) Theory of linear and integer programming. Wiley, Newark
11. Schrijver A (2006) A combinatorial algorithm for minimizing submodular functions in strongly polynomial time. J Comb Theory B80:346–355
12. Stearns RE (1968) Convergent transfer schemes for n-person games. Trans Am Math Soc 134:449–459
13. Yarom M (1981) The lexicographic kernel of a cooperative game. Math Oper Res 6:88–100