

---

Faculty of Mathematical Sciences



University of Twente  
The Netherlands

---

P.O. Box 217  
7500 AE Enschede  
The Netherlands  
Phone: +31-53-4893400  
Fax: +31-53-4893114  
Email: [memo@math.utwente.nl](mailto:memo@math.utwente.nl)  
[www.math.utwente.nl/publications](http://www.math.utwente.nl/publications)

---

MEMORANDUM No. 1663

Computing an element in the  
lexicographic kernel of a game

U. FAIGLE<sup>1</sup>, W. KERN AND J. KUIPERS<sup>2</sup>

NOVEMBER, 2002

ISSN 0169-2690

---

<sup>1</sup>Zentrum für Angewandte Informatik Köln, Universität zu Köln, Weyertal 80, D-50931 Köln

<sup>2</sup>Department of Mathematics, Maastricht University, PO Box 616, NL-6200 MD Maastricht

# Computing an element in the lexicographic kernel of a game

Ulrich Faigle

Zentrum für Angewandte Informatik  
Universität zu Köln, Weyertal 80, D-50931 Köln

Walter Kern

Faculty of Electrical Engineering,  
Mathematics and Computer Science  
University of Twente, P.O.Box 217, NL-7500 AE Enschede

Jeroen Kuipers

Department of Mathematics  
Maastricht University  
P.O. Box 616, NL-6200 MD Maastricht

## Abstract

The lexicographic kernel of a game lexicographically maximizes the surpluses  $s_{ij}$  (rather than the excesses as would the nucleolus). We show that an element in the lexicographic kernel can be computed efficiently, provided we can efficiently compute the surpluses  $s_{ij}(x)$  corresponding to a given allocation  $x$ . This approach improves the results in [2] 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 *lexicographic kernel* is a solution concept for cooperative games, introduced by Kalai (*cf.* [9]). The basic idea is to lexicographically maximize the vector of surpluses  $s_{ij}$  (rather than the vector of excesses as in the definition of the nucleolus). So this solution concept tries to combine characteristic features of both the nucleolus and the kernel.

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” ([9]) in the sense that it is completely determined by the (least) core as a subset of the euclidian space. M. Yarom ([9]) investigates continuity aspects of this solution concept and presents a bargaining procedure converging to the lexicographic kernel.

The purpose of our work 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 [2] for finding an element in the kernel. Under weak assumptions, concerning the efficient computability of the surpluses, 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 [2]: As the lexicographic kernel is a subset of the kernel, we are now able to completely eliminate Maschler's *transfer steps* (*cf.* [8]) for the computation of a kernel element.

## 2 Basic definitions

We consider (*cooperative*) games  $(N, c)$ , where  $N = \{1, \dots, n\}$  is the set of players and  $c : 2^N \rightarrow \mathbb{R}$  a *cost function*, assigning a cost  $c(S)$  to every *coalition*  $S \subseteq N$ . We assume throughout that  $c(\emptyset) = 0$ . An *allocation* is a vector  $x \in \mathbb{R}^n$  satisfying  $x(N) = c(N)$ , where we use the standard shorthand notation

$$x(S) = \sum_{i \in S} x_i .$$

We let  $X \subseteq \mathbb{R}^n$  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  $\varepsilon$ -*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  $\varepsilon$  is the unique maximum number for which the  $\varepsilon$ -core is nonempty, we obtain the so-called *least core*.

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

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

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.

Replacing the excesses by the surpluses in the last definition, we obtain the so-called *lexicographic pre-kernel*  $\mathcal{K}_{\text{lex}}(c)$ . Thus  $\mathcal{K}_{\text{lex}}(c)$  is the set of allocations  $x \in X$  that lexicographically maximize the  $n(n - 1)$ -dimensional vector obtained by arranging the surpluses  $s_{ij}(x)$  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 in advance. We therefore work with the above “pre-solution concepts”. It is straightforward to modify the algorithm we present in Section 4 so that it computes elements in the lexicographic kernel.

It is well-known that the (pre-)nucleolus is contained in the intersection of the (pre-)kernel with the least core. A similar relation holds for the lexicographic (pre-)kernel.

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

*Proof:* The inclusion  $\mathcal{K}_{\text{lex}}(c) \subseteq \text{least core}(c)$ , 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 surpluses non-decreasingly:

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

If  $x \notin \mathcal{K}(c)$ , 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  $\alpha > 0$ . In this case, we could execute a *transfer* of size  $\alpha$  and pass from  $x$  to the allocation

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

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

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

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

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

$$s_{ji}(x) \leq c(\bar{S}) - x(\bar{S}) < s_{\ell m}(x) + \alpha \leq s_{ij}(x) + \alpha,$$

which contradicts the equality  $s_{ji}(x) = s_{ij}(x) + 2\alpha$ .  $\square$

We end this section by mentioning that the lexicographic (pre-)kernel does not necessarily contain the (pre-)nucleolus, *cf.* [5],[9].

### 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.* [2] for more details and additional motivation). 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  $N = \{1, \dots, n\}$  of players
- An upper bound  $\langle c \rangle$  on the maximum size of a cost value, *i.e.*,  $\langle c \rangle \geq \max_{S \subseteq N} c(S)$ .
- 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 [2] for a concrete example.)

For certain classes of games (*e.g.*, minimum spanning tree games, *cf.* [1]), 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 compute the nucleolus or elements in the (lexicographic) kernel efficiently. We therefore assume that our class  $\mathcal{C}$  of games satisfies

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

As shown in [2], this assumption is tantamount to the efficient computability of the surpluses  $s_{ij}(x)$ . Furthermore, not only the surpluses  $s_{ij}(x)$  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.* [8]). Hence (CCM) holds, for example, for any class of convex games. A concrete example is provided, *e.g.*, by Mediggo’s [6] 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.* [4]).

## 4 The lexicographic pre-kernel

We consider a fixed class  $\mathcal{C}$  of games satisfying (CCM). Assume  $(N, c) \in \mathcal{C}$  and let  $I$  denote the set of pairs  $(i, j)$  of players  $i \neq j$ . We then consider the problem

$$(1) \quad \varepsilon_1 := \max_{\substack{s_{ij}(x) \geq \varepsilon, \quad (i, j) \in I \\ x \in X}} \varepsilon$$

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  $(x, \varepsilon)$  forms a polyhedron  $P \subseteq \mathbb{R}^{n+1}$ .

Given a vector  $(\bar{x}, \bar{\varepsilon}) \in \mathbb{R}^{n+1}$ , (CCM) allows us to check efficiently whether  $(\bar{x}, \bar{\varepsilon}) \in P$ . Moreover, in case  $(\bar{x}, \bar{\varepsilon}) \notin P$ , we can efficiently determine a corresponding *violated inequality*, i.e., a linear inequality from the constraints in (1) that is violated by  $(\bar{x}, \bar{\varepsilon})$ .

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

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

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

This observation, together with standard results on the ellipsoid method (cf. also [2]), yields an efficient algorithm for solving (1). Note that (1) is feasible and bounded, so optimal solutions exist. (The corresponding optimal  $\varepsilon_1$  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  $(x, \varepsilon_1)$  is an optimal solution of (1). This is straightforward: For each  $(i_1, j_1) \in I$ , we solve

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

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

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

we obtain an allocation  $\bar{x} \in X$ . Due to the concavity of the  $s_{ij}$ , this vector  $\bar{x}$  solves (1) with  $s_{ij}(\bar{x}) \geq \varepsilon_1$  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  $(x, \varepsilon_1)$  of (1).

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

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

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

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

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

There is one problem left. To prove efficiency of our algorithm, we have to analyze the size of the numbers  $\varepsilon_1, \dots, \varepsilon_r$  that we compute iteratively. But this is easy by using the following *a posteriori* argument. Relative to the partition  $I = I_1 \cup \dots \cup I_r$  that we have constructed, the values  $\varepsilon_1, \dots, \varepsilon_r$  are uniquely determined by the solution of the following lexicographic maximization problem

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

with  $n + r$  variables  $x_1, \dots, x_n, \varepsilon_1, \dots, \varepsilon_r$ . 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  $\varepsilon_1, \dots, \varepsilon_r$  are polynomially bounded in  $n$  and  $\langle c \rangle$ , as required.

Summarizing, we have proved

**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  $\mathcal{C}$ ).*  $\square$

## References

- [1] U. Faigle, S.P. Fekete, W. Hochstättler and W. Kern: *On the complexity of testing core membership for min cost spanning tree games*, Int. Journal of Game Theory 26, 361–366 (1997).
- [2] U. Faigle, W. Kern, and J. Kuipers: *On the computation of the nucleolus of a cooperative game*, Int. Journal of Game Theory 30(1), 79–98, (2001).
- [3] M. Justman: *Iterative processes with “nucleolar” restrictions*, Int. Journal of Game Theory 6, 189–212, (1977).
- [4] W. Kern, D. Paulusma: *Matching games: The least core and the nucleolus*, Math. of Operations Research, to appear.
- [5] M. Maschler, B. Peleg: *Stable sets and stable points of set valued dynamic systems with applications to game theory*, SIAM Journal of Control and Optimization 14, 985–995 (1976).
- [6] N. Megiddo: *Computational complexity of the game theory approach to cost allocation for a tree*, Mathematics of Operations Research 3, 189–196 (1987).
- [7] A. Schrijver: *A combinatorial algorithm for minimizing submodular functions in strongly polynomial time*, Journal of Combinatorial Theory B80, 346–355 (2000).
- [8] R.E. Stearns: *Convergent transfer schemes for  $n$ -person games*, Trans. Amer. Math. Soc. 134, 449–459 (1968).
- [9] M. Yarom: *The lexicographic kernel of a cooperative game*, Math. of Operations Research 6(1), 88–100 (1981).