RAIRO-Oper. Res. 42 (2008) 299–314
DOI: 10.1051/ro:2008016

RAIRO Operations Research
www.rairo-ro.org

FAST COMPUTATION OF THE LEASTCORE AND PRENUCLEOLUS OF COOPERATIVE GAMES [*]

Joseph Frédéric Bonnans[1] and Matthieu André[2]

Abstract. The computation of leastcore and prenucleolus is an efficient way of allocating a common resource among n players. It has, however, the drawback being a linear programming problem with $2^n - 2$ constraints. In this paper we show how, in the case of convex production games, generate constraints by solving small size linear programming problems, with both continuous and integer variables. The approach is extended to games with symmetries (identical players), and to games with partially continuous coalitions. We also study the computation of prenucleolus, and display encouraging numerical results.

Résumé. Le calcul du leastcore et du prénucléole est une manière efficace d'allouer une ressource entre n joueurs. L'inconvénient est qu'il suppose la résolution d'un programme linéaire avec $2^n - 2$ contraintes. Dans cet article nous montrons comment, dans le cas de jeux de production convexes, générer des contraintes en résolvant des programmes linéaires mixtes de petite taille. L'approche est étendue aux jeux avec symétries (joueurs identiques) et aux jeux avec coalitions partiellement continues. Nous étudions aussi le calcul du prénucléole, et donnons des résultats numériques prometteurs.

Keywords. Cooperative games, coalitions, constraint generation, decomposition, convex production games, symmetric games, aggregate players, nucleolus.

Mathematics Subject Classification. 91A12, 90C05, 90C11, 91B32.

Received November 7, 2006. Accepted September 5, 2007.

[*] *This study was performed when the second author was with Gaz de France.*

[1] Inria-Futurs and Centre de Mathématiques Appliquées, École Polytechnique, 91128 Palaiseau, France; `Frederic.Bonnans@inria.fr`

[2] Direction Optimisation Amont-Aval Trading, EDF, 1 Place Pleyel, 93282 Saint-Denis Cedex, France; `matthieu.andre@edfgdf.fr`

1. Introduction

In this paper we study the class of cooperative games, in which costs must be allocated in a fair way among n players. This is an important subject for utility networks, or more generally for all companies subject to regulation rules. The analysis takes into account the possibility of coalitions among players. Note that a new feature of our model is that not all coalitions are allowed, which makes sense in real-world applications. We refer to Boyer *et al.* [1] for an overview of cost allocation using the cooperative game theory.

Denote by $\mathcal{S} := \{1, \cdots, n\}$ the set of players, which are to be interpreted as customers, $\mathcal{P}(\mathcal{S})$ the set of coalitions (all possible subsets of $\mathcal{S}$), $\mathcal{P}_1(\mathcal{S}) := \mathcal{P}(\mathcal{S})\backslash\{\emptyset, \mathcal{S}\}$ the set of non trivial coalitions, and $\mathcal{P}_2(\mathcal{S}) \subset \mathcal{P}_1(\mathcal{S})$ the set of possible coalitions.

Given $x \in I\!R^n$ and $S \in \mathcal{P}(\mathcal{S})$, we denote $x(S) := \sum_{i\in S} x_i$. In the sequel x_i will be interpreted as the amount paid by player i, for $i = 1, \dots, n$, so that $x(S)$ means the amount paid by coalition S. With each coalition $S \in \mathcal{P}(\mathcal{S})$ is associated a real valued cost $c(S)$. We say that $x \in I\!R^n$ is an *allocation* if $x(\mathcal{S}) = c(\mathcal{S})$, and in this case we call $x(S)$ the allocation of coalition S, for all $S \in \mathcal{P}(\mathcal{S})$. The *excess* of a coalition S is the amount

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

This is the difference between what the coalition would have to pay if it ignored the other players, and the amount it has to pay using allocation x. A negative excess for coalition S means that it would be advantageous for S to run its own business. In order to avoid that, a possible way of allocating costs consists in maximizing the minimal excess. This amounts in solving the following optimization problem:

$$\max_{\substack{\varepsilon\in I\!R\\ x\in X}} \varepsilon; \quad x(\mathcal{S}) = c(\mathcal{S}); \quad \varepsilon + x(S) \le c(S), \quad \text{for all} \quad S \in \mathcal{P}_2(\mathcal{S}). \qquad (LP)$$

Here we take into account the set of possible coalitions $\mathcal{P}_2(\mathcal{S})$ which is a subset of $\mathcal{P}_1(\mathcal{S})$, as well as the set $X \subset I\!R^n$ restricting the choice of allocations, as typically will happen in practical situations. We say that the allocation x is *feasible* if (in addition to the relation $x(\mathcal{S}) = c(\mathcal{S})$) it belongs to X. We call the set of solutions of (LP) the *leastcore*. The *core* is the set of feasible allocations for which every excess of coalitions in $\mathcal{P}_2(\mathcal{S})$ is nonnegative. If the core is non empty, then it contains the leastcore. These two definitions of core and leastcore generalize the usual ones for which $X = I\!R^n$ and $\mathcal{P}_2(\mathcal{S}) = \mathcal{P}_1(\mathcal{S})$, see Shapley [13] and the historical references in [9]. If the leastcore is not reduced to one point, one can minimize among its solutions the minimal excess (of coalitions whose excess is not binded). By induction one obtain the prenucleolus, a concept due to Schmeidler [12] (see also the axiomatization in Maschler *et al.* [8]).

In the sequel we assume that X is a polyhedron. Then (LP) is a linear program with $n+1$ variables and as many as $2^n - 2$ explicit constraints (if $\mathcal{P}_2(\mathcal{S}) = \mathcal{P}_1(\mathcal{S})$), in addition to the "implicit" constraint $x \in X$.

Since the computation of the leastcore and prenucleolus needs to solve a linear program with possibly $2^n - 2$ contraints but only $n + 1$ variables, generation of constraints is a natural approach. We show in this paper that, if the cost function has a certain convexity property (the structure of convex production game), then there exists a fast procedure for constraint generation. Then we show that symmetric games have symmetric solutions, and relate some continuous relaxations to the desaggregation of classes of small players. We show that (in the case of convex production games) one can extend the constraint generation to the case of symmetric games or continuous relaxations.

Our hypothesis on the cost function is a generalization of linear production games discussed in Owen [10]. Generation of constraints in this case was already studied in Hallefjord, Helming and Jörnstein [6]. However, for generating constraints they solve a (mixed integer continuous) problem in which the data of the linear production game are involved, and that may be expensive (see details in remark 4.1). By contrast, we solve a problem of much smaller size, and extend the approach, as was already said, to the search of symmetric solutions and to the case of continuous relaxations.

Let us mention also two references related to the subject of this paper, that do not assume that cost functions result from a linear production game. Fromen [2] gave a method for reducing the number of linear programs to be solved in order to compute the prenucleolus; still, these linear programs remain of large size. Preux *et al.* [11] gave a theoretical result about a column generation method for computing the prenucleolus. They give no numerical results, and it is not easy to figure out if their approach can be effective in practical situations.

The paper is organized as follows. Section 2 recalls the principle of constraint generation. The variant based on lower estimates of the cost function is presented in Section 3. How such lower estimates are obtained in the case of convex production games is the subject of Section 4. Sections 5 and 6 deal with the extension to problems with symmetries and to partial continuous relaxation, respectively. Section 7 analyzes how to compute the prenucleolus and Section 8 adapts to this problem the idea of constraint generation. Finally Section 9 presents some numerical experiments.

Given an optimization problem say (P), by $F(P)$ and $S(P)$ we denote its set of feasible points and set of solutions, respectively.

2. Constraints generation

Relaxing constraints of (LP) for some of the coalitions amount to solve the following problem:

$$\max_{x\in X,\varepsilon} \varepsilon; \quad x(\mathcal{S}) = c(\mathcal{S}); \quad \varepsilon + x(S) \le c(S), \quad \text{for all} \ \ S \in E, \qquad (LP_E)$$

where E is a subset of $\mathcal{P}_2(\mathcal{S})$. Let us formulate an algorithm for solving (LP), based on generation of constraints:

```
Algorithm GENERATION
Data: $k := 0$, $E_0 \subset \mathcal{P}_2(\mathcal{S})$.
    LOOP
      • Compute $x^k$, solution of $(LP_{E_k})$.
      • Find $S_k \in \mathcal{P}_2(\mathcal{S})$ such that
            $c(S_k) - x^k(S_k) \leq c(S) - x^k(S)$,  for all $S \in \mathcal{P}_2(\mathcal{S})$.
      • If $c(S_k) - x^k(S_k) = \mathrm{val}(LP_{E_k})$, stop.
      • $E_{k+1} := E_k \cup \{S_k\}$; $k := k+1$.
    END LOOP
End
```

At each iteration the most violated constraint of (LP) is added. Since the number of constraints of (LP) is finite, the algorithm terminates. In addition we have the estimate

$$c(S_k) - x^k(S_k) \leq \mathrm{val}(LP) \leq \mathrm{val}(LP_{E_k}). \tag{2.1}$$

The first inequality expresses the fact that $(x^k, c(S_k) - x^k(S_k)) \in F(LP)$, while the second is a consequence of the relaxation of constraints. Since E_k is increasing with k, $\mathrm{val}(LP_{E_k})$ is nonincreasing. Relation (2.1) may be used for designing a stopping criterion.

This approach, however, will not be effective unless we have a fast way of finding the most violated constraint of (LP). A first step consists in using, instead of the cost function itself, a lower estimate. This is the subject of the next section.

3. Lower estimates of the cost function

Assume that we have at our disposal a *lower estimate* of $c(\cdot)$ over $\mathcal{P}_2(\mathcal{S})$, *i.e.*, a function $\Psi_k : \mathcal{P}_2(\mathcal{S}) \to I\!R$ such that

$$\Psi_k(S) \leq c(S), \quad \text{for all} \quad S \in \mathcal{P}_2(\mathcal{S}). \tag{3.2}$$

Then instead of searching the minimum over non trivial coalitions of $c(S) - x(S)$, we may search for the minimum in S of $\Psi_k(S) - x(S)$ over $\mathcal{P}_2(\mathcal{S})$. We obtain the following algorithm (we write LB for lower bound) :

```
Algorithm GENERATION-LB
Data: $k := 0$, $E_0 \subset \mathcal{P}_2(\mathcal{S})$.
    LOOP
      • Compute $x^k$, solution of $(LP_{E_k})$.
      • Find $S_k \in \mathcal{P}_2(\mathcal{S})$ such that
            $\Psi_k(S_k) - x^k(S_k) \leq \Psi_k(S) - x^k(S)$,  for all $S \in \mathcal{P}_2(\mathcal{S})$.
      • If $c(S_k) - x^k(S_k) = \mathrm{val}(LP_{E_k})$, stop.
```

```
        • $E_{k+1} := E_k \cup \{S_k\}$; $k := k+1$.
    END LOOP
END
```

We have the estimate, similar to (2.1):

$$\Psi_k(S_k) - x^k(S_k) \le \mathrm{val}(LP) \le \mathrm{val}(LP_{E_k}). \tag{3.3}$$

This approach via lower estimates of the cost function is of interest if the two following conditions are satisfied: (i) finding the coalition minimizing $S \mapsto (\Psi_k(S) - x^k(S))$ over $\mathcal{P}_2(\mathcal{S})$ is cheap, and (ii) Ψ_k is as close as possible to $c(S)$. A natural assumption is that the lower estimate Ψ_k is *exact* over E_k, in the sense that

$$\Psi_k(S) = c(S), \quad \text{for all} \;\; S \in E_k. \tag{3.4}$$

Then the above algorithm terminates, as the following Lemma shows.

Lemma 3.1. *Assume that the exactness hypothesis (3.4) holds. Then the algorithm stops after a finite number of iterations, and its output is an allocation of the leastcore.*

Proof. If the algorithm does not terminate, since the set $\mathcal{P}_2(\mathcal{S})$ is finite, we have that $S_k \in E_k$ for some k. By the exactness hypothesis (3.4), $\Psi_k(S_k) = c(S_k)$, in contradiction with the stopping criterium at iteration k. $\square$

It remains to identify in which situations we are able to construct lower estimates of the cost function, such that the problem of finding S_k is tractable. We will see in the next section that these properties are, in a certain sense, satisfied if the cost function has a property of inner convexity.

4. Convex production games

Assume that each player i is a customer for whom must be provided an amount $b^i \in I\!R^p$ of certain goods. The amount needed by coalition S is $b(S) := \sum_{i \in S} b^i$. Assume also that the cost of providing b is the value function of an optimization program of the following type:

$$\operatorname*{Min}_z f(z); \quad Az = b, \quad z \ge 0, \tag{P_b}$$

where $f : I\!R^m \to I\!R$ is a convex function. Then we speak of a convex production game; when f is linear this reduces to the linear production game setting (see Owen [10]). The *Lagrangian function* of problem (P_b) is (denoting by "$\cdot$" the scalar product)

$$L(z, \lambda, s, b) := f(z) + \lambda \cdot (Az - b) - s \cdot z,$$

with here $\lambda \in I\!R^p$ and $s \in I\!R^m_+$. Denote by $v(b)$ the value of (P_b), *i.e.*, $v(b) = \text{val}(P_b)$. Let $\bar{z}$ be solution of $(P_{\bar{b}})$, and let $(\bar{\lambda}, \bar{s})$ be an associated Lagrange multiplier, that is,

$$L(\bar{z}, \bar{\lambda}, \bar{s}, \bar{b}) = \min_z L(z, \bar{\lambda}, \bar{s}, \bar{b}); \quad \bar{s} \geq 0; \quad \bar{s} \cdot \bar{z} = 0. \tag{4.5}$$

Then for any $z \in F(P_b)$, we have that

$$f(z) \geq L(z, \bar{\lambda}, \bar{s}, b) = L(z, \bar{\lambda}, \bar{s}, \bar{b}) - \bar{\lambda} \cdot (b - \bar{b}) \geq L(\bar{z}, \bar{\lambda}, \bar{s}, \bar{b}) - \bar{\lambda} \cdot (b - \bar{b}).$$

Minimizing over $z \in F(P)$, and using $L(\bar{z}, \bar{\lambda}, \bar{s}, \bar{b}) = v(\bar{b})$, obtain

$$v(b) \geq v(\bar{b}) - \bar{\lambda} \cdot (b - \bar{b}). \tag{4.6}$$

Therefore, solving $(P_{\bar{b}})$ provides an affine minorant of $v(\cdot)$, exact (equal to $v(\cdot)$) at $\bar{b}$. Consequently, for all $S \in \mathcal{P}_1(\mathcal{S})$,

$$c(S) = v(b(S)) \geq v(b(S_k)) - \lambda_k \cdot (b(S) - b(S_k)), \tag{4.7}$$

where λ_k is a Lagrange multiplier computed when solving $(P_{b(S_k)})$. Coming back to the definition of the cost function $c(\cdot)$, we get that

$$c(S) \geq c(S_k) - \lambda_k \cdot (b(S) - b(S_k)). \tag{4.8}$$

Set $\hat{c}_k := c(S_k) + \lambda_k \cdot b(S_k)$. At iteration K, we obtain a lower estimate $\Psi_K(S)$ of $c(S)$, where

$$\Psi_K(S) := \max_{0 \leq k \leq K} \{\hat{c}_k - \lambda_k \cdot b(S)\}. \tag{4.9}$$

Given $S \in \mathcal{P}_2(\mathcal{S})$, denote by $\mathbf{1}_S$ its characteristic vector

$$(\mathbf{1}_S)_i = 1 \text{ if } i \in S; \; 0 \text{ otherwise.} \tag{4.10}$$

We identify $y \in \mathcal{P}_2(\mathcal{S})$ with the characteristic vector of an element of $\mathcal{P}_2(\mathcal{S})$. It follows that minimizing $\Psi_K(S) - x(S)$ over $\mathcal{P}_2(\mathcal{S})$ (in order to generate the next constraint) means solving a optimization problem with n variables in $\{0,1\}$ and a continuous variable

$$\underset{\substack{w \in I\!R \\ y \in \mathcal{P}_2(\mathcal{S})}}{\text{Min}} \quad w - \sum_{i=1}^n y_i x_i; \quad \hat{c}_k - \lambda_k \cdot \left(\sum_{i=1}^n y_i b^i \right) \leq w, \quad \text{for all } k \leq K. \tag{L_K}$$

If the constraint $y \in \mathcal{P}_2(\mathcal{S})$ are expressed as linear constraints over $\{0,1\}^n$, then (L_K) is in the format of a mixed linear program (with $\{0,1\}$ and continuous variables). This holds in particular in the "standard case", when $\mathcal{P}_2(\mathcal{S}) = \mathcal{P}_1(\mathcal{S})$, since

$$\mathcal{P}_1(\mathcal{S}) = \left\{ y \in \{0,1\}^n; \; 1 \leq \sum_{i=1}^n y_i \leq n-1 \right\}. \tag{4.11}$$

In addition, two successive problems of type (L_K) differ only by the addition of one cut. This structure may help for a fast resolution of (L_K) (this is the classical situation in branch and cut algorithms, where hot start options allow to speed up computations).

Remark 4.1. The method presented here is related to the column generation method by Hallefjord *et al.* [6]. The essential difference is that [6] uses the true value of the cost function for generating the new column, *i.e.*, the problem to be solved at each iteration is (when, as in [6], we choose $\mathcal{P}_2(\mathcal{S}) = \mathcal{P}_1(\mathcal{S})$):

$$\underset{\substack{z\geq 0\\ y\in\{0,1\}^n}}{\text{Min}} \quad f(z) - \sum_{i=1}^{n} y_i x_i; \quad Az = \sum_{i=1}^{n} b_i y_i; \quad 1 \leq \sum_{i=1}^{n} y_i \leq n-1. \qquad (L'_K)$$

If the dimension of z is large, or if f has a complex expression, this may be much more expensive, especially during the first iterations where solving (L_K) is quite cheap.

4.1. Linear production games

For the purpose of comparison to our previous analysis, we recall in this section a result of Owen [10] (generalized by Granot [5]), as well as its proof, since the latter is short, for proving the existence and providing a fast computation of an allocation in the core for specific linear production games of the following form:

$$\underset{z}{\text{Min}}\, c^\top z; \quad Az \geq b, \quad z \geq 0. \qquad (P'_b)$$

We denote the *Lagrangian function* of problem (P'_b) by

$$L(z,\lambda,b) := c^\top z - \lambda^\top (Az - b)$$

where $\lambda \in \mathbb{R}^p_+$. The dual problem is

$$\sup_{\lambda\geq 0} b^\top \lambda; \quad c - A^\top \lambda \geq 0.$$

Theorem 4.2 (Owen [10])**.** *Assume that $\bar{b} := b(\mathcal{S})$ is such that problem $(P'_{\bar{b}})$ has a finite value. Then the core is non empty, and any dual optimal solution $\bar{\lambda}$ is such that $\bar{x} : S \mapsto b(S)^\top \bar{\lambda}$ is an allocation with non negative excess, for all possible coalitions.*

Proof. The duality theory for linear programs implies $c(\mathcal{S}) = \bar{b}^\top \bar{\lambda}$, so that $\bar{x}$ is an allocation. Given any coalition S, since the feasible set of the dual does not depend on b, we have that $\bar{\lambda}$ is feasible for the corresponding dual problem $\text{Max}_{\lambda\geq 0}\{b(S)^\top \lambda;\ c - A^\top \lambda \geq 0\}$. By duality theory again, we have then that $c(S) \geq b(S)^\top \bar{\lambda} = x(S)$, *i.e.*, $c(S) - x(S) \geq 0$, proving that $\bar{x}$ is an allocation in the core. $\square$

For linear production games in the format $(P'_{\bar b})$, Owen's theorem 4.2 proves non emptyness of the core and provides a cheap way of computing a corresponding allocation. Yet it does provide neither a set of active constraints nor an allocation in the leastcore. Even if the core happens to be equal to the leastcore (but that is not known *a-priori*), a constraint generation procedure is still useful for computing the prenucleolus; see Section 7.

Note also that if we add to the formulation of $(P'_{\bar b})$ at least one linear constraint with constant right-hand-side, then the proof is not valid.

5. Symmetries and disaggregation

This section deals with the case when each (aggregated) player i is in fact the aggregation of n_i *identical* "elementary players". In other words, coalitions with only some of the n_i elementary players are possible, and their cost is a function of the number of elementary players in each class of aggregated players. The disagregated formulation has therefore $\hat n := \sum_i n_i$ players. We denote the cost function of the disaggregated formulation again by $c(\cdot)$; this function depends only on the fraction of players of each class belonging to the coalition (and not on the elementary players themself). Let us show that the disagregated problem has a symmetric solution:

Lemma 5.1. *We assume that the set $\mathcal{P}_2(\mathcal{S})$ of feasible coalitions of the disagregated problem is symmetric (i.e., invariant by permutation between players of the same class). Then the disagregated problem has a symmetric solution, i.e., one for which the allocation is identical for all elementary players of the same class.*

Proof. The problem of computing the leastcore being convex, its solution set is convex. Taking the average value of a particular solution and of all those obtained by permutation between players of the same class, we obtain a symmetric solution. □

Since these symmetric solutions are socially fair (they allocate the same amount to identical players) it is of interest to obtain them through a dedicated formulation. Denote by $y_i \in \{0, 1/n_i, 2/n_i, \dots, 1\}$ the fraction of elementary players of class i in coalition $y \in \mathbb{R}^n$; the set of feasible coalitions of the disagregated problem (again assumed to be polyhedral) is still denoted $X \subset \mathbb{R}^n$. Then the resource allocated to coalition y, where $y_i \in \{0, 1/n_i, 2/n_i, \dots, 1\}$ is still of the form $x(y) := \sum_i x_i y_i$. The computation of symmetric elements of the leastcore may be written as

$$\max_{\substack{\varepsilon\in\mathbb{R}\\ x\in X}} \varepsilon; \quad x(\mathcal{S}) = c(\mathcal{S}); \quad \varepsilon + x(y) \le c(y), \qquad \text{for all } \; y_i \in \{0, 1/n_i, 2/n_i, \dots, 1\}, \quad i = 1, \dots, n. \tag{$DNLP$}$$

Problem $(DNLP)$ has $(\Pi_i(1+n_i) - 2)$ constraints, which is much less than the $\left(2^{\sum_i n_i} - 2\right)$ constraints of the naive formulation of the disagregated problem. If a lower bound Ψ of $c(\cdot)$ is available (in particular in the case of a convex production game) then the constraint generation problem may be written as

$$\operatorname*{Min}_{y} \Psi_K(y) - x(y); \quad y_i \in \{0, 1/n_i, 2/n_i, \dots, 1\}, \quad i = 1, \dots, n. \qquad (DNL)$$

This is an integer (with in addition a continuous variable) linear programming problem. For convex production games we can use the minorant discussed in the previous section, and have therefore an effective procedure.

6. Partial continuous relaxation

If a player $i \in \{1, \dots, n\}$ is the aggregation of a very large number of players, then we might think of approximating the disagregated problem by allowing y_i to be in $[0,1]$. This process will be called the *continuous relaxation* of class i.

Let $\{1, \dots, n_1\}$ be the set of players for which a continuous relaxation is performed. A *relaxed coalition* is an element of $\mathcal{P}^R(\mathcal{S}) := [0,1]^{n_1} \times \{0,1\}^{n_2}$, where $n_2 := n - n_1$. The set of nontrivial coalitions is $\mathcal{P}_1^R(\mathcal{S}) := \mathcal{P}^R(\mathcal{S}) \setminus \{\emptyset, \mathcal{S}\}$. The (symmetric) allocation for a relaxed coalition $y \in \mathcal{P}^R(\mathcal{S})$ is $x(y) = x \cdot y$. Given $\mathcal{P}_2^R(\mathcal{S}) \subset \mathcal{P}_1^R(\mathcal{S})$, the leastcore is now a solution of the following problem:

$$\max_{\substack{\varepsilon \in I\!R \\ x \in X}} \varepsilon; \quad x(\mathcal{S}) = c(\mathcal{S}); \quad \varepsilon + x(y) \le c(y), \quad \text{for all } \; y \in \mathcal{P}_2^R(\mathcal{S}). \qquad (RNLP)$$

Since this problem has infinitely many constraints, the idea of generation of constraints appears to be quite natural. Given an allocation $x \in I\!R^n$, the problem of finding the most active constraint may be formulated as follows: find $y \in \mathcal{S}$ such that

$$c(y) - x(y) \le c(y') - x(y'), \quad \text{for all } \; y' \in \mathcal{P}_2^R(\mathcal{S}). \qquad (6.12)$$

This is not an easy problem. However, in the case of a convex production game, with a relaxed coalition is associated the amount of goods $b(y) := \sum_{i=1}^n y_i b_i$, the associated cost is $c(y) := v(b(y))$, and the computation of v and its subdifferential at points $b_k := b(y_k)$, $k \le K$, provides a convex lower bound that we denote again Ψ_K. The problem of generating a constraint reads then as

$$\operatorname*{Min}_{y \in \mathcal{P}^R(\mathcal{S})_2} \Psi_K(y) - x(y). \qquad (RNL)$$

It may be interpreted as a partially (for the first n_1 variables) continuous relaxation of problem (L_K). Therefore again in the case of convex production games we have an effective constraint generation procedure. This constraint will not of course find the solution after a finite number of iterations but rather converge to the optimal value, following standard results on cutting planes methods for minimizing a convex function due to Kelley [7].

7. Computation of the prenucleolus

This section is devoted to a detailed discussion of the number of steps needed for computing the prenucleolus. After the presentation of the method we introduce the notion of essential steps. Our main result (proposition 7.5) is that the computation needs at most n essential steps.

In this section we assume for the sake of simplicity that $X = I\!R^n$. Let η be a Lagrange multiplier associated with the inequality constraints of the linear program (LP) (η has dimension $2^n - 2$). Denote

$$I_1 := \{S \in \mathcal{P}_1(\mathcal{S}); \eta_S > 0\}; \quad J_1 := \mathcal{P}_1(\mathcal{S}) \setminus I_1. \tag{7.13}$$

From the duality theory of linear programs, we know that $(x, \varepsilon_1) \in S(LP)$ iff it is feasible and satisfies complementarity relations with a Lagrange multiplier:

$$\begin{array}{ll} x(\mathcal{S}) = c(\mathcal{S}); & \varepsilon_1 + x(S) = c(S),\ S \in I_1; \\ & \varepsilon_1 + x(S) \le c(S),\ S \in J_1. \end{array} \tag{7.14}$$

Let $v_1 := \mathrm{val}(LP)$. Relation (7.14) implies in particular $\varepsilon_1 = v_1$. It is important, however, to keep ε_1 as a variable in practical computations in order to avoid instabilities (see Rem. 7.1 below).

One may try to maximise, over solutions of (LP), the minimal excess for coalitions in J_1; this means solving the following problem:

$$\max_{x,\varepsilon_1,\varepsilon_2} \varepsilon_2; \quad \begin{array}{ll} x(\mathcal{S}) = c(\mathcal{S}); & \varepsilon_1 + x(S) = c(S),\ S \in I_1; \\ & \varepsilon_2 + x(S) \le c(S),\ S \in J_1. \end{array} \tag{LP_1}$$

We did not repeat the last constraints of (7.14), which are automatically satisfied at any solution of (LP_1).

Remark 7.1. Problem (LP_1) is feasible, and also qualified in the sense that there exists a feasible point for which all inequality constraints are strictly satisfied: take (x, ε_1) solution of (LP), and $\varepsilon_2 < \varepsilon_1$. In order to obtain a numerically stable formulation, however, one has to eliminate possibly redundant equality constraints in (LP_1).

Once (LP_1) is solved, we may continue solving a sequence of problems of the form

$$\max_{x,\varepsilon_1,\ldots,\varepsilon_{k+1}} \varepsilon_{k+1}; \quad x(\mathcal{S}) = c(\mathcal{S}); \quad \begin{array}{rcll} \varepsilon_1 + x(S) & = & c(S), & S \in I_1; \\ & \vdots & & \\ \varepsilon_k + x(S) & = & c(S), & S \in I_k; \\ \varepsilon_{k+1} + x(S) & \le & c(S), & S \in J_k. \end{array} \tag{LP_k}$$

Sets I_k and J_k are defined inductively for $k \ge 2$:

$$I_k := \{S \in J_{k-1};\ \eta_S^k > 0\}; \quad J_k := J_{k-1} \setminus I_k,$$

where η^k is a Lagrange multiplier associated with the inequality constraints of program (LP_{k-1}). The sequence stops when I_k is empty.

Denote $v_k := \operatorname{val}(LP_k)$. Since the solution $(x, \varepsilon_1, \ldots, \varepsilon_k)$ of (LP_{k-1}), with $\varepsilon_{k+1} = \varepsilon_k$, is a feasible point of (LP_k), the sequence v_k is nondecreasing. If $v_k = v_{k-1}$ and $I_k \neq \emptyset$, then I_k is a set of constraints that are always active at any solution of (LP_{k-1}), and at the same time $\eta_S^{k-1} = 0$, for all $S \in I_{k-1}$.

By the Goldman-Tucker Theorem [4], we know that a linear program satisfies the hypothesis of strict complementarity. In other words, if a linear program has a finite value, then its dual has at least one solution such that its nonzero components coincide with the set of active inequality constraints at all primal solutions. Such strictly complementarity multipliers coincide with the relative interior of the solution set of the dual program. We denote the dual of (LP_k) by (DLP_k), and the relative interior of the solution set for the dual as $\operatorname{ri} S(DLP_k)$.

Lemma 7.2. *The inequality $v_k > v_{k-1}$ holds iff $\eta^{k-1} \in \operatorname{ri} S(DLP_{k-1})$.*

Proof. If $\eta^{k-1} \notin \operatorname{ri} S(DLP_{k-1})$, this means that strict complementarity with one primal solution does not hold: there exists a coalition $S \in J_{k-1}$ such that $x(S) = c(S) - \varepsilon_k$ for all $(x, \varepsilon_1, \ldots, \varepsilon_k) \in S(LP_{k-1})$. Since any feasible point $(x, \varepsilon_1, \ldots, \varepsilon_{k+1})$ of (LP_k) is such that $(x, \varepsilon_1, \ldots, \varepsilon_k) \in S(LP_{k-1})$, we deduce that $v_k = \varepsilon_{k+1} \leq \varepsilon_k = v_{k-1}$ Since the converse inequality always holds, this proves that $v_k = v_{k-1}$.

If on the contrary $\eta^{k-1} \in \operatorname{ri} S(DLP_{k-1})$, with each $S \in J_{k-1}$ is associated some $(x_S, \varepsilon_1, \ldots, \varepsilon_k) \in S(LP_{k-1})$ (values of ε_i are identical over $S(LP_{k-1})$) such that $x_S(S) < c(S) - \varepsilon_k$. Denote by j_{k-1} the cardinal of J_{k-1}, and set $\hat{x} := (j_{k-1})^{-1} \sum_{S \in J_{k-1}} x_S$. Since $S(LP_{k-1})$ is a convex set, we have that $(\hat{x}_S, \varepsilon_1, \ldots, \varepsilon_k)$ belongs to$S(LP_{k-1})$, and also $\hat{x}_S(S) < c(S) - \varepsilon_k$, for all $S \in J_{k-1}$, which implies that there exists some $\hat{\varepsilon} > \varepsilon_k$ such that $(\hat{x}_S, \varepsilon_1, \ldots, \varepsilon_k, \hat{\varepsilon})$ is a feasible point of J_{k-1}. Therefore $\operatorname{val}(LP_k) > \hat{\varepsilon} > \varepsilon_k$. $\square$

Remark 7.3. An interior-point solver provides a Lagrange multiplier in the relative interior of $S(DLP_{k-1})$, unless a purification procedure is performed. On the contrary, simplicial algorithms compute in general a Lagrange multiplier on the relative boundary of $S(DLP_{k-1})$.

In order to estimate the number of steps in the computation of prenucleolus, we need the following definition.

Definition 7.4. (i) Let $I \subset \mathcal{P}_1(\mathcal{S})$ and $\bar{S} \in \mathcal{P}_1(\mathcal{S})$. We say that $\bar{S}$ depends on I if the value of $x(\bar{S})$ is determined by the values $\{x(S), S \in I\}$. We say that $J \subset \mathcal{P}_1(\mathcal{S})$ depends on I if any $S \in J$ depends on I.
(ii) Step k in the computation of prenucleolus is said to be *essential* if at least one element of I_k does not depend on $\mathcal{S} \cup (\cup\{I_\ell; \ell < k\})$.

If S depends on I, then the linear program

$$\operatorname*{Min}_x x(S); \quad x(S') = 0; \quad S' \in I$$

has value 0; by duality, we deduce easily that S depends on I iff $\mathbf{1}_S$ is a linear combination of vectors $\{\mathbf{1}_{S'}, S' \in I\}$.

Lemma 7.5. *The computation of prenucleolus has at most n essential steps.*

Proof. Each essential step adds a linear constraint, linearly independant from the linear constraints already active. Since $x \in \mathbb{R}^n$, this may occur at most n times. □

Remark 7.6. (i) In order to understand the nature of non essential steps, consider the case when the leastcore has a unique element. Then the computation of the prenucleolus reduces to an ordering of the non saturated constraints following their value at the leastcore. Obviously the number of non essential steps may be extremely large.

(ii) One could check uniqueness of the solution of problem (LP_k) in order to stop the procedure.

(iii) After having solved problem (LP_k), one could try to compute and eliminate all inequality constraints that depend on the equality constraints of (LP_k). However, we do not know any fast algorithm for doing it.

(iv) It is useful, in order to have stable computations, to check the linear independance of the equality constraints of (LP_k). This can be done by *e.g.* a QR type orthogonal factorization, see [3].

8. Generation of constraints and prenucleolus

8.1. General case

The idea of constraint generation can be extended to the computation of prenucleolus. Without entering into all details, let us specify the essential step, which is the formulation of the problem of generating a constraint after solving (LP_k):

$$\begin{array}{l}\text{Compute } \bar{S} \in J_k \text{ such that} \\ c(\bar{S}) - x(\bar{S}) \leq c(S) - x(S), \quad \text{for all } S \in J_k.\end{array} \tag{8.15}$$

Here k, J_k and x are given. If we have only a lower boud Ψ of the cost function $c(\cdot)$, the problem to be solved reads as:

$$\begin{array}{l}\text{Compute } \bar{S} \in J_k \text{ such that} \\ \Psi(\bar{S}) - x(\bar{S}) \leq \Psi(S) - x(S), \quad \text{for all } S \in J_k.\end{array} \tag{8.16}$$

In the case of a convex production game (Sect. 4), the lower bound has an expression of type

$$\Psi(S) := \max_{\ell \in L} \left\{ \hat{c}_\ell - \lambda_\ell \cdot b(S) \right\}, \tag{8.17}$$

where L is a finite set. The problem to be solved has an expression similar to the one of problem (L_K) of Section 4, but with the additional constraint $y \in J_k$. It is useful to rewrite it as a mixed (discrete and continuous variables) linear program in order to be able to solve it effectively. This can be done in the following way. Given z and y in $\{0,1\}^n$, denote

$$a^z(y) := \sum_{\{i;z_i=0\}} y_i + \sum_{\{i;z_i=1\}} (1-y_i). \tag{8.18}$$

This sum on nonnegative amounts is equal to 0 iff $y = z$, and otherwise has value at least 1. The constraint $a^z(y) \geq 1$ is therefore equivalent to the constraint $y \neq z$. Hallefjord *et al.* [6] already used this formulation for the computation of the prenucleolus.

Denote by $\bar{J}_k := \mathcal{P}_1(\mathcal{S}) \setminus J_k$ the set of coalitions for which an equality constraint already holds. The problem of constraint generation for convex production games can be written as

$$\operatorname*{Min}_{\substack{w\in I\!R\\ y\in\{0,1\}^n}} \; w - \sum_{i=1}^n y_i x_i; \quad \begin{cases} \hat{c}_\ell - \lambda_\ell \cdot \left(\displaystyle\sum_{i=1}^n y_i b^i\right) \leq w, \quad \text{for all } \; \ell \in L; \\ 1 \leq \displaystyle\sum_{i=1}^n y_i \leq n-1; \\ a^z(y) \geq 1, \quad \text{for all } z \in \bar{J}_k. \end{cases} \tag{$\bar{L}_{L,k}$}$$

Excluding redundant equality constraints, we always have $|\bar{J}_k| \leq n$. Problem $(\bar{L}_{L,k})$ seems therefore of acceptable complexity.

8.2. Case of symmetric games

In the case of symmetric games, described in Section 5, the approach is similar. The main difference is that since now $n_i y_i$ is an integer in $\{0, n_i\}$ rather than $\{0,1\}$ (see problem $(DNLP)$), the exclusion constraint has a different expression. We need some notations. Let

$$Y := \{y \in I\!R^n; \quad n_i y_i \in \{0,\dots,n_i\}\} \tag{8.19}$$

be the set of such fractions y. Given $z \in Y$, we look for an expression of the exclusion constraint $\{y \in Y; y \neq z\}$. For that we split y as a sum $y = y^\ell + y^u$, where for each component i, $y_i^\ell = \min(y_i, z_i)$ and $y_i^u = \max(y_i - z_i, 0)$. Then $y \in Y$ is such that

$$a^z(y) := \sum_{i=1}^n (z_i - y_i^\ell + y_i^u) \tag{8.20}$$

is nonnegative, and has value 0 iff $y = z$. We obtain the following set of conditions, for $y \in Y$:

$$\begin{cases} y^\ell \in Y; y^u \in Y; \alpha \in \{0,1\}^n; \\ a^z(y) = 0; \quad y = y^\ell + y^u; \\ \alpha_i z_i \leq y_i^\ell \leq z_i; \quad 0 \leq y_i^u \leq \alpha_i(1-z_i). \end{cases} \tag{8.21}$$

Again, excluding redundant equality constraints, we always have no more that n sets of relations of type (8.21). The corresponding problem of generating a constraint, of the type stated below, is therefore again of acceptable complexity:

$$\operatorname*{Min}_{\substack{w\in I\!R\\ y\in Y}} w - \sum_{i=1}^{n} y_i x_i; \quad \begin{cases} \hat{c}_\ell - \lambda_\ell \cdot \left(\displaystyle\sum_{i=1}^{n} y_i b^i\right) \leq w, \quad \text{for all} \quad \ell \in L; \\ 0 \neq \displaystyle\sum_{i=1}^{n} y_i \neq n; \\ (8.21) \text{ holds, for all } z \in \bar{J}_k. \end{cases} \qquad (\bar{L}_{LS,k})$$

9. Numerical experiments

We tested the constraint generation algorithms of the following family of test problems. Consider the problem of allocating costs for a water network connecting n different cities. Each city i requires a quantity $Q_{i,t}$ of water over two different time periods $t \in T$. All these city exploit a unique spring s, with no supply restrictions. For each period t, the supply is denoted $\mathbf{Y}_t$.

A site numbered n (spring or city) has coordinates (x_n, y_n); the cost of building a pipe between two cities n_1, n_2 of capacity $\mathbf{C}_{n_1,n_2}$ is proportional to the (Euclidean) distance d_{n_1,n_2} between the two sites, the unit price being p. The total cost (for a given coalition) includes also a fixed cost IC; the expression of the total cost is therefore

$$f(\mathbf{C}) := IC + p \sum_{n_1 < n_2} d_{n_1,n_2} \mathbf{C}_{n_1,n_2}. \qquad (9.22)$$

The flow $\mathbf{F}_{n_1,n_2,t}$ must satisfy the capacity constraints, as well as Kirchhoff's law at each site n_1 for all times:

$$0 \leq \mathbf{F}_{n_1,n_2,t} \leq \mathbf{C}_{n_1,n_2}, \quad \text{for all} \quad t \in T \ \forall (n_1, n_2), \qquad (9.23)$$

$$\sum_{n_2} \mathbf{F}_{n_1,n_2,t} + Q_{n_1,t} = \sum_{n_2} \mathbf{F}_{n_2,n_1,t} \quad \text{for all} \quad t \in T, \ n_1 \neq s, \qquad (9.24)$$

$$\sum_{n_2} \mathbf{F}_{s,n_2,t} = \sum_{n_2} \mathbf{F}_{n_2,s,t} + \mathcal{V} Y_t, \quad \text{for all} \quad t \in T. \qquad (9.25)$$

The minimum cost design problem for a coalition $\mathcal{S}$ is therefore

$$\operatorname{Min} f(\mathbf{C}), \quad \text{subject to} \quad (9.23)\text{–}(9.25).$$

Note that cooperation is clearly profitable between cities, since (i) the investment cost is shared between cities, and (ii) the two periods structure needs a nontrivial use of available capacities (flows are not necessarily equal to capacities as would be the case for only one time period).

Also note that by increasing the number of steps of time, one would increase the complexity of optimization, without adding a significant new dimension of

TABLE 1. (Average, standard deviation) of computational time (in ds).

Cities, data	Classical	Hallefjord	NCGM
(4, 100)	(17.6, 2.3)	(10.8, 5.9)	(10.2, 5.2)
(5, 100)	(33.2, 2.3)	(14.5, 15.2)	(13.2, 12.8)
(6, 100)	(70.2, 7.5)	(19.7, 21.7)	(18.0, 16.6)
(7, 100)	(178.5, 31.0)	(91.5, 172.3)	(70.4, 121.8)
(8, 100)	(488.3, 174.5)	(177.3, 512.9)	(121.7, 330.6)
(9, 100)	(1365.9, 403.0)	(273.7, 1194.4)	(191.1, 765.9)

TABLE 2. Rounded (average, standard deviation) of used coalitions.

Cities, data	Classical	Hallefjord	NCGM
(4, 100)	(16, 0)	(4, 1)	(4, 1)
(5, 100)	(32, 0)	(6, 2)	(6, 2)
(6, 100)	(64, 0)	(7, 2)	(7, 2)
(7, 100)	(128, 0)	(11, 8)	(11, 9)
(8, 100)	(256, 0)	(12, 11)	(12, 11)
(9, 100)	(512, 0)	(12, 13)	(13, 14)

cooperation. I thus suppose that the structure of the numerical results should be comparable.

The commercial software AIMMS 3.5, and its uniform random number generator, has been used for generating the data of problems. Sites (cities or spring) coordinates, just like pipe capacities, and cities demand $Q_{i,t}$ are real and random, all in the range [–10; 10]. Unit price p is equal to 1/unit of water, and fixed cost IC is equal to 10.

We compared three different methods to compute the prenucleolus: the classical algorithm (no constraint generation), the constraint generation approach in [6], and our new constraint generation method (called NCGM).

Tables 1 and 2 display the results for different problem sizes[1].

According to these results, NCGM seems to be the best among the three methods. Large values of standard deviation also tend to show an important dependency of the performance of constraint generation methods to the data of the problem, whereas classical method is more stable. We also noticed that constraint generation tends to be less efficient when the number of iterations needed to find the prenucleolus increases: it may be explained by the increasing number of cuts of the constraint generation algorithm, which are really time consuming.

Acknowledgements. The authors thank two anonymous refereees for their useful remarks.

[1]Computer configuration: Windows 2000 SP2 - RAM: 1 Go - CPU: 2.4 GHz.

References

[1] M. Boyer, M. Moreau and M. Truchon, Partage des coût et tarification des infrastuctures. Les méthodes de partage de coût. Un survol. Technical Report *RP-18*, Cirano (2002).

[2] B. Fromen, Reducing the number of linear programs needed for solving the nucleolus problem of n-person game theory. *Eur. J. Oper. Res.* **98** (1997) 626–636.

[3] P.E. Gill, W. Murray and M.H. Wright, *Practical optimization.* Academic Press, London (1981).

[4] A.J. Goldman and A.W. Tucker, Polyhedral convex cones. In H.W. Kuhn and A.W. Tucker, editors, *Linear inequalities and related systems*, Princeton, 1956. Princeton University Press 19–40.

[5] D. Granot, A generalized linear production model: a unifying model. *Math. Program.* **34** (1986) 212–222.

[6] A. Hallefjord, R. Helming and K. Jörnstein, Computing the nucleolus when the characteristic function is given implicitly: a constraint generation approach. *Int. J. Game Theor.* **24** (1995) 357–372.

[7] J.E. Kelley, The cutting plane method for solving convex programs. *J. Soc. Indust. Appl. Math.* **8** (1960) 703–712.

[8] M. Maschler, J.A.M. Potters and S.H. Tijs, The general nucleolus and the reduced game property. *Int. J. Game Theor.* **21** (1992) 85–106.

[9] Jörg Oswald, Jean Derks and Hans Peters, Prenucleolus and nucleolus of a cooperative game: characterizations by tight coalitions. In *3rd International Conference on Approximation and Optimization in the Caribbean (Puebla, 1995)*, *Aportaciones Mat. Comun.* vol. **24**, Soc. Mat. Mexicana, México (1998) 197–216.

[10] G. Owen, On the core of linear production games. *Math. Program.* **9** (1975) 358–370.

[11] N. Preux, F. Bendali, J. Mailfert, and A. Quilliot. Cœur et nucléolus des jeux de recouvrement. *RAIRO-Oper. Res.* **34** (2000) 363–383.

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

[13] L.S. Shapley, On balanced sets and cores. *Nav. Res. Logist. Q.* **14** (1967) 453–460.