Fair cost allocations under conflicts — a game-theoretic point of view —

Yoshio Okamoto*

Institute of Theoretical Computer Science, Department of Computer Science, ETH Zürich, CH-8092, Zürich, Switzerland. okamotoy@inf.ethz.ch

Abstract. We study the cost allocation problem when the players are involved in a conflict situation. More formally, we consider a minimum coloring game, introduced by Deng, Ibaraki & Nagamochi, and provide algorithms for the core, the τ-value, the nucleolus and the Shapley value on some classes of graphs. The investigation gives several insights to the relationship of algorithm theory with cooperative games.

1 Introduction

In cooperative works (projects, or whatever people do in cooperation) we can find two algorithmic problems. First, when people work together, they wish to minimize the cost. This is a focus of optimization theory. Secondly they wish to allocate the minimized total cost in a fair manner. This is a focus of cooperative game theory.

This paper studies fair cost allocations of conflict situations from the viewpoint of cooperative game theory. As a simple model of a conflict, we will use a conflict graph. In a conflict graph, each vertex represents each agent involved in the situation. Two vertices are adjacent through an edge if the corresponding agents are in conflict under the situation. We model the total cost of the conflict simply as it is proportional to the chromatic number of the conflict graph. So we want to divide the chromatic number of the graph and allocate it to each agent. This kind of cooperative games was first introduced by Deng, Ibaraki & Nagamochi [2] by the name of a minimum coloring game. Their paper [2] and a subsequent paper by Deng, Ibaraki, Nagamochi & Zang [3] included some investigations of the core (a sort of fair allocations) of a minimum coloring game.

In this paper, we will study minimum coloring games more thoroughly. Among many kinds of concepts on fair allocations in cooperative game theory, we consider the core (introduced by Gillies [9]), the τ-value (by Tijs [25]), the nucleolus (by Schmeidler [21]) and the Shapley value (by Shapley [23]). For general graphs, it turns out hard to compute any of these fair cost allocations (Proposition 1). Therefore, we will concentrate on a specific class of graphs,

* Supported by the Berlin-Zürich Joint Graduate Program "Combinatorics, Geometry, and Computation" (CGC), financed by ETH Zürich and the German Science Foundation (DFG).

namely perfect graphs. This makes sense since perfect graphs give rise to totally balanced minimum coloring games [3]. For perfect graphs, we will show the following theorems.

- The core of the minimum coloring game on a perfect graph is characterized as the convex hull of the characteristic vectors of the maximum cliques of the graph (Theorem 2). This is a generalization of a result by Deng, Ibaraki & Nagamochi [2] for a bipartite graph, and furthermore it implies that we can compute a core allocation in polynomial time and that we can decide whether a given vector belongs to the core or not in polynomial time for perfect graphs (Corollary 3).
- The τ-value for a perfect graph can be computed in polynomial time. This is a consequence of Theorem 4 and a polynomial-time algorithm for computing the chromatic number of a perfect graph [11]. So far, we only knew that this could be done for complete multipartite graphs (since complete multipartite graphs give rise to submodular minimum coloring games [20] and the τ-value of a submodular cost game can be computed in polynomial time by the application of a polynomial-time algorithm for the submodular function minimization [11]).
- The nucleolus is the barycenter of the characteristic vectors of the maximum cliques for some classes of perfect graphs (Theorem 6). This class includes the complete multipartite graphs and the chordal graphs, which appear in some application contexts like scheduling [10] (Proposition 7). As a consequence, for chordal graphs, we obtain a polynomial-time algorithm for computing the nucleoli. We can see that this is much simplified for forests. In addition, for a complete multipartite graph, we can see that the Shapley value and the nucleolus coincide (Corollary 8).
- The Shapley value can be computed efficiently for the minimum coloring game on a forest (Sect. 3.5). However, the same technique cannot be applied to bipartite graphs in general (Proposition 10).

Perhaps, it was Megiddo [17] who first noticed the computational issue on cooperative game theory. Since then, there have been many studies on the computational complexity and algorithms for the core and the nucleolus and a little about the Shapley value (here we omit the vast references). Especially, cooperative games arising from combinatorial optimization problems fit very much into the framework of algorithm theory. These kinds of cooperative games are called "combinatorial optimization games" and some of these cooperative games and their properties are discussed in Curiel's book [1]. However, none of them dealt with minimum coloring games in spite of the importance as a simple model of conflicts, except for a small amount of papers [2, 3, 20]. Since these works [2, 3, 20] only studied the core, this paper provides the first treatment of solution concepts of minimum coloring games other than cores, and will shed more light on relationship of algorithm theory with cooperative games.

The paper is organized as follows. In the next section, we will introduce some graph-theoretic and game-theoretic concepts which we need in the rest of the paper. Sect. 3 is the main part of the paper. Sect. 4 concludes the paper with

some open problems. Note that, because of the page limitation, all of the proofs will be omitted in this version. (See the full-paper version for the proofs.)

2 Preliminaries

Throughout the paper, for a vector $\boldsymbol{x} \in \mathbb{R}^N$ and $S \subseteq N$, we write $\boldsymbol{x}(S) := \sum\{x_i : i \in S\}$. When $S = \emptyset$, set $\boldsymbol{x}(S) := 0$. For a subset $S \subseteq N$ of a finite set N, the *characteristic vector* of S is a vector $\mathbb{1}_S \in \{0,1\}^N$ defined as $(\mathbb{1}_S)_i = 1$ if $i \in S$ and $(\mathbb{1}_S)_i = 0$ if not.

A *graph* G is a pair $G = (V, E)$ of a finite set V and a set $E \subseteq \binom{V}{2}$ of 2-element subsets of V. (So, our graph is always simple and finite, and V can be empty.) We assume the familiarity of basic terminology from graph theory. Here we just introduce a perfect graph. A graph is *perfect* if for each of the induced subgraphs the chromatic number is equal to the size of a maximum clique. It is known that bipartite graphs, complete multipartite graphs and chordal graphs are perfect. (A graph is *chordal* if each of the induced cycles has length three.)

A *cooperative game* (or simply a *game*) is a pair (N, γ) of a nonempty finite set N and a function $\gamma : 2^N \to \mathbb{R}$ satisfying $\gamma(\emptyset) = 0$. Often, an element of N is called a *player* of the game, and γ is called the *characteristic function* of the game. Furthermore, each subset $S \subseteq N$ is called a *coalition*. Literally, for $S \subseteq N$ the value $\gamma(S)$ is interpreted as the total profit (or the total cost) for the players in S when they work in cooperation. In particular, $\gamma(N)$ represents the total profit (or cost) for the whole players when they all agree with working together. When γ represents a profit, we call the game a *profit game*. On the other hand, when γ represents a cost, we call the game a *cost game*.[1] In this paper, we will mainly consider a certain class of cost games.

One of the aims of cooperative game theory is to provide a concept of "fairness," namely, how to allocate the total cost (or profit) $\gamma(N)$ to each player in a "fair" manner when we take all the $\gamma(S)$'s into account. In the rest of this section, we will describe some allocation rules which are considered fair in cooperative game theory. Formally, a cost allocation is defined as a preimputation in the terminology of cooperative game theory. A *preimputation* of a cost game (N, γ) is a vector $\boldsymbol{x} \in \mathbb{R}^N$ satisfying $\boldsymbol{x}(N) = \gamma(N)$. Each component x_i expresses how much the player $i \in N$ should owe according to the cost allocation $\boldsymbol{x}$.

Now we will define some fair allocations, namely a core allocation, the τ-value, the nucleolus and the Shapley value. See Driessen's book [4] for details (why they are seen fair) and other kinds of fair allocation concepts from cooperative game theory. Let (N, γ) be a cost game. A vector $\boldsymbol{x} \in \mathbb{R}^N$ is called a *core allocation* if $\boldsymbol{x}$ satisfies the following conditions: $\boldsymbol{x}(N) = \gamma(N)$ and $\boldsymbol{x}(S) \leq \gamma(S)$ for all $S \subseteq N$. The set of all core allocations of a cost game (N, γ) is called the *core* of (N, γ). The core was introduced by Gillies [9]. Note that a core might be empty. Therefore, a cost game with a nonempty core is especially interesting, and such a cost game is called *balanced*. Moreover, we call a cost game *totally balanced* if each

[1] So the terms "profit game" and "cost game" are not mathematically determined. They are just determined by the interpretation of a game.

of the subgames has a nonempty core. (Here, a *subgame* of a cost game (N, γ) is a cost game $(T, \gamma^{(T)})$ for some nonempty $T \subseteq N$ defined as $\gamma^{(T)}(S) = \gamma(S)$ for each $S \subseteq T$.) Naturally, a totally balanced game is also balanced. A special subclass of the totally balanced games consists of submodular games (Shapley [22]), where a cost game (N, γ) is called *submodular* (or *concave*) if it satisfies the following condition: $\gamma(S) + \gamma(T) \geq \gamma(S \cup T) + \gamma(S \cap T)$ for all $S, T \subseteq N$. Therefore, we have a chain of implications: submodularity $\Rightarrow$ total balancedness $\Rightarrow$ balancedness. These implications are fundamental in cooperative game theory.

Here we introduce the τ-value due to Tijs [25]. Let (N, γ) be a cost game. Define two vectors $\underline{\boldsymbol{m}}, \overline{\boldsymbol{m}} \in \mathbb{R}^N$ by $\underline{m}_i := \gamma(N) - \gamma(N \setminus \{i\})$ and $\overline{m}_i := \min\{\gamma(S) - \underline{\boldsymbol{m}}(S \setminus \{i\}) : i \in S \subseteq N\}$ for each $i \in N$. It is known [25] that, if a cost game (N, γ) is balanced, it holds that $\underline{m}_i \leq \overline{m}_i$ for each $i \in N$ and $\underline{\boldsymbol{m}}(N) \leq \gamma(N) \leq \overline{\boldsymbol{m}}(N)$. Then, the *$\tau$-value* of a balanced cost game (N, γ) is defined as a vector $\boldsymbol{x} \in \mathbb{R}^N$ uniquely represented by $\boldsymbol{x} := \lambda \underline{\boldsymbol{m}} + (1 - \lambda)\overline{\boldsymbol{m}}$ where we choose $\lambda \in [0, 1]$ so that $\boldsymbol{x}(N) = \gamma(N)$ is satisfied. From the definition we can see that the τ-value can be computed in polynomial time for submodular games by an algorithm for the submodular function minimization [11].

Next, we will define the nucleolus, which was introduced by Schmeidler [21]. It is known that the nucleolus always belongs to the core if it is not empty. For our purpose, it is better to use an algorithmic definition of the nucleolus due to Peleg (referred by Kopelowitz [12]) than the original one by Schmeidler [21]. In this scheme, we successively solve a series of linear programming problems (P_1), (P_2), and so on. For a cost game (N, γ), the i-th problem (P_i) is described as follows: find $(\boldsymbol{x}, \epsilon) \in \mathbb{R}^N \times \mathbb{R}$ to

$$
\begin{array}{lll}
(\mathsf{P}_\mathsf{i}): \text{ maximize} & \epsilon & \\
\text{subject to} & \boldsymbol{x}(N) = \gamma(N) & \\
& \boldsymbol{x}(S) = \gamma(S) - \epsilon_l & \text{for all } S \in \mathcal{C}_l \\
& & \text{for all } l \in \{1, \ldots, i-1\} \\
& \boldsymbol{x}(S) \leq \gamma(S) - \epsilon & \text{for all } S \in \mathcal{C}_0 \setminus \bigcup_{l=1}^{i-1} \mathcal{C}_l,
\end{array}
$$

where $\mathcal{C}_0 = 2^N \setminus \{N, \emptyset\}$, ϵ_j is the optimal value of (P_j), and $\mathcal{C}_j = \{S \in \mathcal{C}_0 \setminus \bigcup_{l=1}^{j-1} \mathcal{C}_l : \boldsymbol{x}(S) = \gamma(S) - \epsilon_j \text{ for all optimal solutions } (\boldsymbol{x}, \epsilon_j) \text{ of } (\mathsf{P}_\mathsf{j})\}$ for every $j = 1, \ldots, i-1$. It is known that finally, say, at the t-th step, the problem has a unique optimal solution $(\boldsymbol{x}^*, \epsilon_t)$. (This was essentially shown by Maschler, Peleg & Shapley [15].) Then the vector $\boldsymbol{x}^*$ is called the *nucleolus* of the game. Notice that the procedure above is not a polynomial-time algorithm. Indeed it is NP-hard to compute a nucleolus for a totally balanced game [6]. However, the nucleolus can be computed in polynomial time for a submodular game [7, 13].

The Shapley value is another cost allocation we will study. The *Shapley value* of a game (N, γ) is a vector $\boldsymbol{x} \in \mathbb{R}^N$ which is defined as

$$
x_i := \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} (\gamma(S \cup \{i\}) - \gamma(S)).
$$

The Shapley value was first introduced by Shapley [23]. By the definition, the Shapley value always uniquely exists even if the game is not balanced. However, the Shapley value may not be a core allocation. On an algorithmic side, there exists an exponential lower bound for the computation of the Shapley value [5].

3 Minimum coloring games

As we saw in Sect. 1, the minimum coloring problem is a simple model of conflicts. The corresponding cost game is called a minimum coloring game. In Sect. 3.1 we will define a minimum coloring game, and subsequently we will investigate the core, the τ-value, the nucleolus and the Shapley value.

3.1 Definition and a hardness result

Let $G = (V, E)$ be a graph. The *minimum coloring game* on G is a cost game (V, χ_G) where $\chi_G : 2^V \to \mathbb{R}$ is defined as $\chi_G(S) := \chi(G[S])$ for all $S \subseteq V$. Remember that $G[S]$ represents the subgraph of G induced by S, and $\chi(G)$ is the chromatic number of G.

Deng, Nagamochi & Ibaraki [2] proved that it is NP-complete to decide whether the minimum coloring game on a given graph is balanced. Later, Deng, Ibaraki, Nagamochi & Zang [3] showed that the minimum coloring game on a graph G is totally balanced if and only if G is perfect. So the decision problem on the total balancedness of a minimum coloring game is as hard as recognizing perfect graphs. Furthermore, Okamoto [20] showed that the minimum coloring game on a graph G is submodular if and only if G is complete multipartite. So we can decide whether a given graph yields a submodular minimum coloring game in polynomial time.

In this paper, we are interested in the computation of a core allocation, the τ-value, the nucleolus and the Shapley value of a minimum coloring game. The following observation is important although it is not difficult to show.

Proposition 1. *It is NP-hard to compute a preimputation of the minimum coloring game on a graph given from a class for which the computation of the chromatic number is NP-hard.*

Proposition 1 suggests that, in order to obtain a polynomial-time algorithm to compute a certain preimputation of a minimum coloring game, it would be better to concentrate on a class of graphs for which the chromatic number can be computed in polynomial time. Perfect graphs form such a class. From now on, we will concentrate on perfect graphs, and in Sect. 3.2–3.5, we will investigate the core, the τ-value, the nucleolus and the Shapley value respectively.

3.2 The core of a minimum coloring game

As described above, the minimum coloring game on a perfect graph is totally balanced [3], which implies that the core is nonempty. Now we characterize the core for a perfect graph in terms of its extreme points.

Theorem 2. *The core of the minimum coloring game on a perfect graph is the convex hull of the characteristic vectors of the maximum cliques of the graph.*

Notice that Theorem 2 generalizes a theorem by Deng, Ibaraki & Nagamochi [2] on bipartite graphs to all perfect graphs.

Making use of a result by Grötschel, Lovász & Schrijver [11] with Theorem 2, we will conclude the following algorithmic consequences.

Corollary 3. *The following problems can be solved in polynomial time. (1) The problem to compute a core allocation of the minimum coloring game on a perfect graph. (2) The problem to decide whether a given vector belongs to the core of the minimum coloring game on a perfect graph.*

3.3 The τ-value of a minimum coloring game

As we observed before, the τ-value can be computed in polynomial time for a submodular game by an algorithm for the submodular function minimization. However, for minimum coloring games, this only works for complete multipartite graphs. In this subsection, we will describe how to compute the τ-values in polynomial time for perfect graphs. Remark that the τ-value for a perfect graph is well-defined since the minimum coloring game on a perfect graph is totally balanced [3]. The next theorem is crucial.

Theorem 4. *Let $G = (V, E)$ be a perfect graph. Consider the minimum coloring game (V, χ_G) on G. If we denote all the maximum cliques of G by $K_1, ..., K_k$ and $K = K_1 \cap \cdots \cap K_k$, then we have that, for each $v \in V$, $\underline{m}_v = 1$ if $v \in K$ and $\underline{m}_v = 0$ otherwise; $\overline{m}_v = 1$ if $v \in K$ or $K \subseteq N_G(v)$ and $\overline{m}_v = 0$ otherwise.*

Based on Theorem 4, we are able to establish an algorithm to compute the τ-value of the minimum coloring game on every perfect graph. First we will compute $\underline{\boldsymbol{m}}$. To do that, we will just compute $\chi_G(V)$ and $\chi_G(V \setminus \{v\})$ and conclude that $\underline{m}_v = \chi_G(V) - \chi_G(V \setminus \{v\})$ for all $v \in V$. Since the chromatic number of a perfect graph can be computed in polynomial time [11], $\underline{\boldsymbol{m}}$ can also be computed in polynomial time. From $\underline{\boldsymbol{m}}$, we can see what K is, by using Theorem 4, in linear time. Namely, $K = \{v \in V : \underline{m}_v = 1\}$. After we know K, we can immediately compute $\overline{\boldsymbol{m}}$ again with help of Theorem 4. Finally, we can also compute the appropriate λ in a straightforward manner because we have already known $\chi_G(V)$. Thus we have obtained the τ-value and this algorithm runs in polynomial time.

Note that Theorem 4 enables us to compute the τ-value for a complete multipartite graph without using an algorithm for the submodular function minimization. (Actually, we can easily find the explicit formula of the τ-value for such a graph from Theorem 4.)

3.4 The nucleolus of a minimum coloring game

Now we will investigate the nucleolus. Unfortunately, we are not aware of an efficient algorithm for general perfect graphs. So we restrict to a subclass of the perfect graphs. To do that, we will introduce some terms.

Let $G = (V, E)$ be a perfect graph, $\mathcal{I}_G$ be the family of nonempty independent sets of G, and $K_1, K_2, \ldots, K_k$ be the maximum cliques of G. For every nonempty independent set $I \in \mathcal{I}_G$, define $g(I) := |\{i \in \{1, \ldots, k\} : |I \cap K_i| = 1\}|$. Then consider the set $\Gamma := \{g(I) : I \in \mathcal{I}_G\}$. (Note that Γ is not a multiset.) Let us enumerate the elements of Γ in decreasing order as: $g_1 > g_2 > \cdots > g_{|\Gamma|}$. The following lemma due to Lovász [14] tells us $g_1 = k$.

Lemma 5 ([14]). *For a perfect graph G, there exists an independent set of G which has a nonempty intersection with each of the maximum cliques of G.*

On the other hand, we have less control for g_2 in general. This motivates the definition of O-goodness. A perfect graph G is called *O-good* if $|\Gamma| = 1$ or G satisfies the following condition: for each $j \in \{1, \ldots, k\}$ there exists an independent set $I^{(j)} \in \mathcal{I}_G$ such that $|I^{(j)} \cap K_j| = 0$ and $|\{i \in \{1, \ldots, k\} \setminus \{j\} : |I^{(j)} \cap K_i| = 1\}| = g_2$.

Our theorem on the nucleolus is as follows.

Theorem 6. *The nucleolus of the minimum coloring game on an O-good perfect graph is the barycenter of the characteristic vectors of the maximum cliques of the graph.*

Now we will observe what kind of perfect graphs are O-good.

Proposition 7. *1. A perfect graph with only one maximum clique is O-good. 2. A forest is O-good. 3. A complete multipartite graph is O-good. 4. A chordal graph is O-good.*

Note that there exists a bipartite graph which is not O-good and for which the nucleolus is not given as in Theorem 6.

Now we will describe how to compute the nucleolus for classes of O-good perfect graphs in Proposition 7. First, let us consider a forest, which is O-good by Proposition 7.2. In a forest $G = (V, E)$, the maximum cliques are the edges of G. Let $d_G(v)$ be the degree of $v \in V$ in G. Then, Theorem 6 concludes that the v-th component of the nucleolus for the forest G is $d_G(v)/|E|$. So the nucleolus for a forest can be easily computed.

Next, consider a complete multipartite graph, which is O-good by Proposition 7.3. Let $G = (V, E)$ be a complete r-partite graph in which V is partitioned into $V_1, \ldots, V_r$ and $E = \{\{u, v\} : u \in V_i,\ v \in V_j,\ i, j \in \{1, \ldots, r\},\ i \neq j\}$. Then a maximum clique K is a vertex subset which satisfies that $|V_i \cap K| = 1$ for all $i \in \{1, \ldots, r\}$. So, with help of Theorem 6, the v-th component of the nucleolus for the complete r-partite graph G can be computed as $1/|V_i|$ if $v \in V_i$. Actually in a past work, the author investigated the Shapley value for a complete multipartite graph [20], and this is exactly the same as the expression above. Thus we immediately obtain the following corollary.

Corollary 8. *For the minimum coloring game on a complete multipartite graph, the Shapley value and the nucleolus coincide.*

Notice that Corollary 8 does not hold for a general submodular game.

For a chordal graph, which is O-good by Proposition 7.4, we are not aware of a closed formula for the nucleolus. However, we can still compute the nucleolus in polynomial time. The strategy is to enumerate all the maximum cliques. A polynomial-time algorithm to enumerate all the maximal cliques of a chordal graph is known (and described in Golumbic's book [10]), which is based on a proposition by Fulkerson & Gross [8]. (Note that the number of the maximal cliques of a chordal graph is bounded by the number of the vertices from above [8].) Therefore, from this enumeration, we are able to obtain the list of all maximum cliques of the chordal graph. In this way, we can compute the nucleolus from this list with Theorem 6.

3.5 The Shapley value of a minimum coloring game

In the former work [20], the author provided the formula of the Shapley value for a complete multipartite graph (see also Corollary 8). Here we will give a polynomial-time algorithm to compute the Shapley value of a minimum coloring game on a forest. Note that a forest is bipartite. On the Shapley value for a bipartite graph, we have the following lemma.

Lemma 9. *For a bipartite graph $G = (V, E)$, the v-th component of the Shapley value $\boldsymbol{\phi}$ of the minimum coloring game on G can be written as*

$$\phi_v = \frac{1}{n} + \sum_{k=1}^{n-1} a_v^k(G) \frac{k!(n-k-1)!}{n!}$$

where $n = |V|$ and $a_v^k(G)$ is the number of independent sets I of G such that $v \notin I$, $|I| = k$ and $I \cup \{v\}$ is not independent.

Therefore, if we have a polynomial-time algorithm to compute $a_v^k(G)$ for every $k \in \{1, \ldots, n-1\}$, we are able to obtain the Shapley value in polynomial time. However we are not aware of such a procedure for a general bipartite graph. Actually, we have a hardness result again. (This is just a simple application of the fact that counting the number of independent sets in a bipartite graph is #P-complete [19].)

Proposition 10. *For bipartite graphs G with n vertices and $k \in \{1, \ldots, n-1\}$, it is #P-complete to compute the value $a_v^k(G)$.*

Therefore, we will consider a special case in which G is a forest. As a result, we give a polynomial-time algorithm to compute $a_v^k(G)$ for every $k \in \{1, \ldots, n-1\}$ when G is a forest with n vertices.

Because of the lack of pages, we are not able to give a description of the algorithm here. The idea is as follows. First, we observe that, when G is a forest, we can reduce the computation of the $a_v^k(G)$ to the computation of the number of independent sets of given sizes in G. Then we establish an algorithm to compute these numbers with the dynamic programming technique. Thus, we can obtain a polynomial-time algorithm to compute the Shapley value for a forest.

4 Conclusion

We have investigated some fair cost allocations (the core, the τ-value, the nucleolus, the Shapley value) under conflict situations by modeling them as minimum coloring games. Our investigation on the core and the nucleolus suggests that results from polyhedral combinatorics are really useful in the study of cooperative games arising from combinatorial optimization games. This was indeed true in the work by Deng, Ibaraki & Nagamochi [2]. Furthermore, our approach to the nucleolus gives a viewpoint different from the literature. In the literature, polynomial-time algorithms for nucleoli of special classes of games were based on properties of the classes such as the number of essential coalitions is small (e.g. [24]) or the nucleolus is a unique vector in the intersection of the core and the kernel (e.g. [6]). Furthermore, usually the core is treated as a system of linear inequalities. On the contrary, in our approach we first look at the core as the convex hull of its extreme points (Theorem 2) and represent the nucleolus as a convex combination of them (Theorem 6). This should also be useful for other cooperative games. In addition, our investigation on the Shapley value suggests the use of a recurrence and the dynamic programming technique. Indeed, it is known that for the power indices of a weighted majority game the dynamic programming is quite useful [16]. Since there seemed to exist no application of the dynamic programming to cooperative games other than weighted majority games so far, it should be interesting to investigate more possibility of the dynamic programming technique for the Shapley value.

Finally, we will mention some open questions which this paper raises. First of all, we do not know how to compute the nucleolus and the Shapley value for a general perfect graph or even for a bipartite graph in polynomial time. It could be NP-hard or #P-hard. Another question is the computation of fair cost allocations for the graphs which are not perfect but for which the chromatic number can be computed in polynomial time, like outer-planar graphs. This would be really interesting since the core for an outer-planar graph can be empty.

Acknowledgements The discussion with Hiroaki Mohri for an article [18] motivated this work. The author thanks him. In addition, the author is grateful to Udo Adamy, Kenji Kashiwabara and Tadashi Sakuma for discussion and/or useful comments and to Emo Welzl for his hospitality.

References

1. I.J. Curiel: Cooperative Game Theory and Applications: Cooperative Games Arising from Combinatorial Optimization Problems. Kluwer Academic Publishers, Dordrecht, 1997.
2. X. Deng, T. Ibaraki and H. Nagamochi: Algorithmic aspects of the core of combinatorial optimization games. Math. Oper. Res. **24** (1999) 751–766.
3. X. Deng, T. Ibaraki, H. Nagamochi and W. Zang: Totally balanced combinatorial optimization games. Math. Program. **87** (2000) 441–452.

4. T. Driessen: Cooperative Games, Solutions and Applications. Kluwer Academic Publishers, Dordrecht, 1988.
5. U. Faigle and W. Kern: The Shapley value for cooperative games under precedence constraints. Internat. J. Game Theory **21** (1992) 249–266.
6. U. Faigle, W. Kern and J. Kuipers: Computing the nucleolus of min-cost spanning tree games is NP-hard. Internat. J. Game Theory **27** (1998) 443–450.
7. U. Faigle, W. Kern and J. Kuipers: On the computation of the nucleolus of a cooperative game. Internat. J. Game Theory **30** (2001) 79–98.
8. D.R. Fulkerson and O.A. Gross: Incidence matrices and interval graphs. Pacific J. Math. **15** (1965) 835–855.
9. D.B. Gillies: Some theorems on n-person games. Ph.D. Thesis, Princeton University, 1953.
10. M.C. Golumbic: Algorithmic Graph Theory and Perfect Graphs. Academic Press, New York, 1980.
11. M. Grötschel, L. Lovász and A. Schrijver: Geometric algorithms and combinatorial optimization. Second edition. Springer-Verlag, Berlin, 1993.
12. A. Kopelowitz: Computation of the kernels of simple games and the nucleolus of n-person games. RM 31, Research Program in Game Theory and Mathematical Economics, The Hebrew University of Jerusalem, 1967.
13. J. Kuipers: A polynomial time algorithm for computing the nucleolus of convex games. Report M 96-12, Maastricht University, 1996.
14. L. Lovász: Normal hypergraphs and the perfect graph conjecture. Discrete Math. **2** (1972) 253–267.
15. M. Maschler, B. Peleg and L.S. Shapley: Geometric properties of the kernel, nucleolus and related solution concepts. Math. Oper. Res. **4** (1979) 303–338.
16. T. Matsui and Y. Matsui: A survey of algorithms for calculating power indices of weighted majority games. J. Oper. Res. Soc. Japan **43** (2000) 71–86.
17. N. Megiddo: Computational complexity and the game theory approach to cost allocation for a tree. Math. Oper. Res. **3** (1978) 189–196.
18. H. Mohri and Y. Okamoto: Discrete optimization and cooperative games (2). (In Japanese.) Comm. Oper. Res. Soc. Japan **48** (2003) 114–120.
19. J.S. Provan and M.O. Ball: The complexity of counting cuts and of computing the probability that a graph is connected. SIAM J. Comput. **12** (1983) 777–788.
20. Y. Okamoto: Submodularity of some classes of the combinatorial optimization games. Math. Methods Oper. Res. **58** (2003), to appear.
21. D. Schmeidler: The nucleolus of a characteristic function game. SIAM J. Appl. Math. **17** (1969) 1163–1170.
22. L.S. Shapley: Cores of convex games. Internat. J. Game Theory **1** (1971) 11–26. Errata is in the same volume, 1972, pp. 199.
23. L.S. Shapley: A value for n-person games. In: H. Kuhn and A.W. Tucker eds., Contributions to the Theory of Games II, Princeton University Press, Princeton, New Jersey, 1953, 307–317.
24. T. Solymosi and T.E.S. Raghavan: An algorithm for finding the nucleolus of assignment games. Internat. J. Game Theory **23** (1994) 119–143.
25. S.H. Tijs: Bounds for the core and the τ-value. In: O. Moeshlin and P. Pallaschke, eds., Game Theory and Mathematical Economics, North Holland, Amsterdam, 1981, 123–132.