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

Yoshio Okamoto[†]

Department of Information and Computer Sciences,
Toyohashi University of Technology,
Hibarigaoka 1-1, Tempaku-cho, Toyohashi, Aichi, 441-8580, Japan
E-mail: `okamotoy@ics.tut.ac.jp`

Abstract

Optimization theory resolves problems to minimize total costs when the agents are involved in some conflicts. In this paper, we consider how to allocate the minimized total cost among the agents. To do that, the allocation is required to be fair in a certain sense. We use a game-theoretic point of view, and provide algorithms to compute fair allocations in polynomial time for a certain conflict situation. More specifically, we study a minimum coloring game, introduced by Deng, Ibaraki & Nagamochi [X. Deng, T. Ibaraki and H. Nagamochi. Algorithmic aspects of the core of combinatorial optimization games. Math. Oper. Res. **24** (1999) 751–766], and investigate the core, the nucleolus, the τ-value, and the Shapley value. In particular, we provide the following four results. (1) The characterization of the core for a perfect graph in terms of its extreme points. This leads to polynomial-time algorithms to compute a vector in the core, and to determine whether a given vector belongs to the core. (2) A characterization of the nucleolus for some classes of the graphs, including the complete multipartite graphs and the chordal graphs. This leads to a polynomial-time algorithm to compute the nucleolus for these classes of graphs. (3) A polynomial-time algorithm to compute the τ-value for a perfect graph. (4) A polynomial-time algorithm to compute the Shapley value for a forest. The investigation of this paper gives several insights to the relationship of algorithm theory with cooperative games.
Key Words: Algorithmic game theory; Combinatorial optimization game; Cooperative game; Solution concept; Transferable utility game

1 Introduction

The main concern of optimization theory is to optimize a certain objective under some constraints. This appears to be quite useful, and the optimization paradigm has been applied to most of the disciplines. Among them, some real-world situations require us to solve the following type of problems: minimizing the total cost for units (or agents like people, nations, companies, jobs,

[*]The extended abstract version has appeared in Proceedings of 14th International Symposium on Algorithms and Computation (ISAAC 2003), Lecture Notes in Computer Science **2906** (2003) 686–695.

[†]This work was done while the author was at Institute of Theoretical Computer Science, Department of Computer Science, ETH Zurich, under the support of the Berlin-Zürich Joint Graduate Program "Combinatorics, Geometry, and Computation" (CGC), financed by ETH Zurich and the German Science Foundation (DFG).

processes, requests and so on) under their conflict. Then, optimization theory is able to resolve this type of problems, and the algorithmic studies for optimization provide how to compute an optimal solution. However, we have further problems to solve. The next requirement is to solve the following type of problems: computing an allocation of the minimized total cost to all units so that each unit can be satisfied. Since a usual optimization theory itself does not answer this type of questions, we need another theory.

However, this is exactly the scope of game theory, especially cooperative game theory. Cooperative game theory has been developing results on fair cost allocations, and we are interested in efficient algorithms to compute such fair cost allocations.

This paper studies fair cost allocations over conflict situations from the viewpoint of cooperative game theory. As a simple model of conflict, we use a conflict graph. In a conflict graph, each vertex represents an agent involved in the situation, and two vertices are adjacent through an edge if the corresponding agents are involved 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 (possibly fractionally) and allocate it to each agent. For example, when we model a certain scheduling problem in a conflict graph, each vertex corresponds to a job to be processed, each edge corresponds to two jobs which cannot be processed on the same machine, and each color corresponds to a machine, so the chromatic number is the number of machines we need to finish all jobs and the total cost (in this case each machine has an independent and identical cost for installation) must be allocated to the jobs. This kind of cooperative games was first introduced by Deng, Ibaraki & Nagamochi [7] as a minimum coloring game. Their paper [7] and subsequent papers by Deng, Ibaraki Nagamochi & Zang [8], Okamoto [32] and Bietenhader & Okamoto [1] studied the core (a sort of fair allocations) of a minimum coloring game.

In this paper, we study the minimum coloring games more thoroughly. Among many kinds of concepts on fair allocations in cooperative game theory, we consider the core (introduced by Gillies [17]), the nucleolus (by Schmeidler [34]), the τ-value (by Tijs [39]), and the Shapley value (by Shapley [35]). In general, it is hard to compute any of these fair cost allocations for a minimum coloring game (Proposition 1). Therefore, we concentrate on a special class of graphs, namely perfect graphs. This makes sense since perfect graphs give rise to totally balanced minimum coloring games [8]. For perfect graphs, we 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 in the graph (Theorem 2). This is a generalization of a result by Deng, Ibaraki & Nagamochi [7] 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 5).

- The nucleolus is the barycenter of the characteristic vectors of the maximum cliques for some class of perfect graphs (Theorem 7). This class includes the complete multipartite graphs and the chordal graphs (Proposition 12), which appear in some application contexts like scheduling and computational biology [18]. As a consequence, for a chordal graph, we obtain a polynomial-time algorithm to compute the nucleolus. The computation can be much simpler in forests. In addition, for a complete multipartite graph, we show that the Shapley value and the nucleolus coincide (Corollary 15).

- The τ-value for a perfect graph can be computed in polynomial time. This is a consequence of

Theorem 17 and a polynomial-time algorithm to compute the chromatic number of a perfect graph [19]. Previously, we only knew that this could be done for complete multipartite graphs (since complete multipartite graphs give rise to submodular minimum coloring games [32] and the τ-value of a submodular cost game can be computed in polynomial time by a polynomial-time algorithm for the submodular function minimization [19]).

- The Shapley value can be efficiently computed for the minimum coloring game on a forest (Section 3.5). However, the same technique cannot be applied to bipartite graphs in general (Proposition 19).

Perhaps, it was Megiddo [28] 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). Comparing several cost allocations from the algorithmic point of view was proposed by Deng & Papadimitriou [9]. They considered that bounded rationality in economics should be reflected as a cost allocation which admits an efficient algorithm in cooperative game theory, and studied several allocation rules from the algorithmic point of view for a certain limited class of cooperative games. Indeed, cooperative games arising from combinatorial optimization problems fit into the framework of algorithm theory very much. Such cooperative games are called "combinatorial optimization games" and some of them are discussed in a book by Curiel [6]. However, none of the past works dealt with minimum coloring games in spite of the importance, except for few papers [1, 7, 8, 32]. Since these papers studied the core only, this paper is the first treatment of solution concepts for minimum coloring games other than cores, and sheds more light on relationship of algorithm theory with cooperative games.

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

2 Preliminaries

First, we present a general notation. 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 $S = \emptyset$, we 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. For $S, T \subseteq N$, it holds that $\mathbb{1}_S(T) = |S \cap T|$.

2.1 Graphs

We assume that the reader has a moderate familiarity with graphs. In this paper, a graph is always simple and finite. Let $G = (V, E)$ be a graph with vertex set V and edge set E. The *(open) neighborhood* of a vertex $v \in V$ in G is the set of vertices adjacent to v, and denoted by $N_G(v)$. The *closed neighborhood* of $v \in V$ in G is $N_G(v) \cup \{v\}$, and denoted by $N_G[v]$. For a subset $U \subseteq V$, the subgraph of G induced by U is denoted by $G[U]$.

A *k-coloring* of a graph $G = (V, E)$ is a function $c\colon V \to \{1, \dots, k\}$. A k-coloring c of G is called *proper* if $c(u) \neq c(v)$ for any edge $\{u, v\} \in E$. From the definition of a proper k-coloring, we can immediately see that if a graph G has a proper k-coloring then G also has a proper k'-coloring for $k' \geq k$. A graph G is *k-colorable* if there exists a proper k-coloring of G. The *chromatic number* of a graph G is the minimum k such that a proper k-coloring of G exists. The chromatic number of G is denoted by $\chi(G)$.

A *clique* of a graph G is a vertex subset $K \subseteq V$ such that $G[K]$ is a complete graph. An *independent set* of a graph G is a vertex subset $I \subseteq V$ such that $G[I]$ has no edge. A clique (and an independent set, respectively) of G is *maximal* if there is no other clique (and independent set, respectively) of G which contains it. A clique (and an independent set, respectively) of G is called *maximum* if it has the largest size among all cliques (and independent sets, respectively) of G. The size of the maximum clique of a graph G is denoted by $\omega(G)$. Note that it holds that $\omega(G) \leq \chi(G)$ for any graph G.

A graph G is called *perfect* if $\omega(H) = \chi(H)$ for all induced subgraphs H of G. The class of perfect graphs includes a lot of interesting subclasses of graphs like forests (i.e., graphs without cycles), bipartite graphs (i.e., 2-colorable graphs), complete multipartite graphs, and chordal graphs. A graph G is *complete r-partite* if $\chi(G) = r$ and a proper r-coloring of G is unique. This is equivalent to saying that there exists a partition of the vertex set V of G into r classes $V_1, \dots, V_r$ such that the edge set of G is $\{\{u, v\} : u \in V_i,\ v \in V_j$ for some distinct $i, j \in \{1, \dots, r\}\}$. A graph is *chordal* if every induced cycle is a triangle. Facts on perfect graphs can be found in books by Golumbic [18] and by Grötschel, Lovász & Schrijver [19].

2.2 Cooperative games

A *cooperative game* (a *transferable utility (TU) game* or simply a *game*) is a pair (N, γ) of a finite set N and a function $\gamma \colon 2^N \to \mathbb{R}$ 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 all 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 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 $\gamma(S)$'s into account. Here, we 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 game (N, γ) is a vector $\boldsymbol{x} \in \mathbb{R}^N$ satisfying $\boldsymbol{x}(N) = \gamma(N)$. The component x_i expresses how much the player $i \in N$ should owe according to the cost allocation $\boldsymbol{x}$.

Now we define some fair allocations, namely a core allocation, a nucleolus, a τ-value, and a Shapley value. See a book by Driessen [11] for details (why they are considered 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 the cost game (N, γ) is called the *core* of (N, γ). The core was introduced by Gillies [17], and many aspects of the core are surveyed by Kannai [22] and Peleg [30]. 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*.[2] Moreover, we

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

[2] Originally, Shapley [36] introduced a balanced game in a different form, and showed that it is equivalent to a game with a nonempty core. Note that Bondareva [2] also showed the same statement. In this paper, we adapt the

call a cost game *totally balanced* if every subgame has a nonempty core. (Here, a *subgame* of a cost game (N, γ) is a cost game $(T, \gamma^{(T)})$ for some nonempty set $T \subseteq N$ 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 is formed by the submodular games (Shapley [37]). A cost game (N, γ) is called *submodular* 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 the following chain of implications:

$$\text{submodularity} \Longrightarrow \text{total balancedness} \Longrightarrow \text{balancedness}.$$

These implications are fundamental in cooperative game theory.

Next, we define the nucleolus, which was introduced by Schmeidler [34]. A detailed survey of the nucleolus is given by Maschler [26]. It is known that the nucleolus always belongs to the core if the core is not empty. For our purpose, it is more convenient to use an algorithmic definition of the nucleolus due to Peleg (as Kopelowitz [23] referred to him) than the original one by Schmeidler [34]. 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

$$\begin{array}{lll}
(\mathsf{P}_i)\text{:} & \text{maximize} & \epsilon \\
& \text{subject to} & \boldsymbol{x}(N) = \gamma(N), \\
& & \boldsymbol{x}(S) = \gamma(S) - \epsilon_\ell \qquad \text{for all } S \in \mathcal{C}_\ell \text{ and} \\
& & \qquad\qquad\qquad\quad\;\; \text{for all } \ell \in \{1, \ldots, i-1\}, \\
& & \boldsymbol{x}(S) \leq \gamma(S) - \epsilon \qquad \text{for all } S \in \mathcal{C}_0 \setminus \displaystyle\bigcup_{\ell=1}^{i-1} \mathcal{C}_\ell,
\end{array}$$

where $\mathcal{C}_0 := 2^N \setminus \{N, \emptyset\}$, ϵ_ℓ is the optimal value of (P_ℓ), and

$$\mathcal{C}_\ell := \left\{ S \in \mathcal{C}_0 \setminus \bigcup_{j=1}^{\ell-1} \mathcal{C}_j : \boldsymbol{x}(S) = \gamma(S) - \epsilon_\ell \text{ for all optimal solutions } (\boldsymbol{x}, \epsilon_\ell) \text{ of } (\mathsf{P}_\ell) \right\}$$

for every $\ell \in \{1, \ldots, i-1\}$. It is known that finally, say at the t-th step, the problem (P_t) has a unique optimal solution $(\boldsymbol{x}^*, \epsilon_t)$. 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 [13]. On the other hand, the nucleolus can be computed in polynomial time for a submodular game [14, 24].

Next, we introduce the τ-value due to Tijs [39]. Let (N, γ) be a cost game. Define two vectors $\underline{\boldsymbol{m}}, \overline{\boldsymbol{m}} \in \mathbb{R}^N$ as

$$\begin{aligned}
\underline{m}_i &:= \gamma(N) - \gamma(N \setminus \{i\}), \\
\overline{m}_i &:= \min\{\gamma(S) - \underline{\boldsymbol{m}}(S \setminus \{i\}) : i \in S \subseteq N\}
\end{aligned}$$

for each $i \in N$. It is known [39] 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 $\underline{\boldsymbol{m}}(N) \leq \gamma(N) \leq \overline{\boldsymbol{m}}(N)$.[3] Then, the τ-*value* of a balanced cost game (N, γ) is

core nonemptiness as the definition of balancedness.

[3]Actually, Tijs [39] called a cost game satisfying these two conditions *quasi-balanced*, and showed that a balanced game is quasi-balanced.

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 $\lambda \in [0,1]$ in such a way that $\boldsymbol{x}(N) = \gamma(N)$ can be satisfied. From the definition we can see that the τ-value can be computed in polynomial time for submodular games by a polynomial-time algorithm for the submodular function minimization [19].

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

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

for each $i \in N$. The Shapley value was first introduced by Shapley [35] (why not!), and is surveyed by Winter [41]. It is not difficult to show that the Shapley value is indeed a cost allocation (i.e., a preimputation). 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 the algorithmic side, there exists an exponential lower bound for the computation of the Shapley value in general [12].

3 Minimum coloring games

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

3.1 Definition and a hardness statement

Let $G = (V, E)$ be a graph. The *minimum coloring game* on G is a cost game (V, χ_G) where $\chi_G \colon 2^V \to \mathbb{R}$ 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 note that we set $\chi_G(\emptyset) = 0$.

Here, before proceeding to the algorithmic aspects of minimum coloring games, we would like to make a remark on the issue of inputs. Usually, when we think of a computational task for a cooperative game (N, γ), we have the characteristic function γ as an oracle which outputs the value $\gamma(S)$ for a queried set $S \subseteq N$, and consider that one oracle call can be done in a constant time. This is because, in order to store the function values $\gamma(S)$ for all $S \subseteq N$, we need $2^{|N|}$ space, and it is already exponential in $|N|$. The remarks given so far in this paper about the computational complexity on cooperative games rely on this model. However, in a minimum coloring game, we take a graph G as an input, and the running time of algorithms is measured by the encoding length of G, not by the oracle complexity model. This makes a real difference since the computation of $\chi_G(V) = \chi(G)$ is generally NP-hard but in the oracle model it could be done just by a single query (i.e., constant time).

Deng, Ibaraki & Nagamochi [7] proved that it is NP-complete to decide whether the minimum coloring game on a given graph is balanced. Later, Deng, Ibaraki, Nagamochi & Zang [8] 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, which is recently shown to be done in polynomial time [3, 4]. Furthermore, Okamoto [32] 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 nucleolus, the τ-value, 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.*

Proof. Suppose that a preimputation $\boldsymbol{x}$ can be computed in polynomial time. Since we have $\boldsymbol{x}(V) = \chi_G(V) = \chi(G)$ (the first equality is due to the definition of a preimputation, and the second equality is due to the definition of a minimum coloring game), we obtain $\chi(G)$ in polynomial time. However, this would be possible only if P = NP since we deal with a class of graphs in which the minimum coloring problem 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, we should concentrate on a class of graphs for which the chromatic number can be computed in polynomial time. Perfect graphs form such a class [19]. From now on, we concentrate on perfect graphs, and in Sections 3.2–3.5, we investigate the core, the nucleolus, the τ-value, 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 [8], 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 [7] on bipartite graphs to all perfect graphs. They showed that the core of the minimum coloring game on a bipartite graph is the convex hull of the characteristic vectors of the edges of the graph.

To prove Theorem 2, we use two facts. The first one is due to Deng, Ibaraki & Nagamochi [7], which is related to the linear programming relaxation of an integer programming formulation of the minimum coloring problem. A minimum coloring game was originally introduced as a subclass of the minimum covering games by Deng, Ibaraki & Nagamochi [7]. Namely, they used a formulation of the minimum coloring problem as a minimum covering problem. Denote by $\mathcal{M}_G$ the family of all maximal independent sets of a graph G and consider the following minimum covering problem formulation of the minimum coloring problem: for a graph $G = (V, E)$ find $\boldsymbol{y} \in \mathbb{R}^{\mathcal{M}_G}$ to

$$
\begin{array}{lll}
(\mathsf{IP}(G)): & \text{minimize} & \displaystyle\sum_{I \in \mathcal{M}_G} y_I \\
& \text{subject to} & \displaystyle\sum_{\substack{I \in \mathcal{M}_G \\ \text{s.t. } v \in I}} y_I \geq 1 \quad \text{for all } v \in V, \\
& & y_I \in \{0,1\} \qquad \text{for all } I \in \mathcal{M}_G.
\end{array}
$$

Let us denote the linear programming relaxation of $(\mathsf{IP}(G))$ by $(\mathsf{LP}(G))$, which looks as follows:

$$\begin{aligned}(\mathsf{LP}(G))\colon\quad &\text{minimize} && \sum_{I\in\mathcal{M}_G} y_I \\ &\text{subject to} && \sum_{\substack{I\in\mathcal{M}_G\\ \text{s.t. } v\in I}} y_I \geq 1 \quad && \text{for all } v\in V, \\ & && y_I \geq 0 && \text{for all } I\in\mathcal{M}_G.\end{aligned}$$

Further, denote the dual of $(\mathsf{LP}(G))$ by $(\mathsf{DLP}(G))$, namely, the problem finding $\boldsymbol{x}\in\mathbb{R}^V$ to

$$\begin{aligned}(\mathsf{DLP}(G))\colon\quad &\text{maximize} && \sum_{v\in V} x_v \\ &\text{subject to} && \sum_{v\in I} x_v \leq 1 \quad && \text{for all } I\in\mathcal{M}_G, \\ & && x_v \geq 0 && \text{for all } v\in V.\end{aligned}$$

Deng, Ibaraki & Nagamochi [7] proved the following statement which we use for the proof of Theorem 2.

Lemma 3 (Deng, Ibaraki & Nagamochi [7]). *Let $G=(V,E)$ be a graph and (V,χ_G) be the minimum coloring game on G. Then, the core of (V,χ_G) is nonempty if and only if there exists an integral optimal solution to $(\mathsf{LP}(G))$. In case it is nonempty, the core is exactly the set of all optimal solutions of $(\mathsf{DLP}(G))$.*

The second fact which we use is due to Lovász [25], Fulkerson [15] and Chvátal [5].

Lemma 4 (Chvátal [5], Fulkerson [15] and Lovász [25]). *Let $Q(G)$ be the convex hull of the characteristic vectors of the cliques of G. Then G is perfect if and only if $Q(G)=\{\boldsymbol{x}\in\mathbb{R}^V : \boldsymbol{x}(I)\leq 1$ for all independent sets I of G, $\boldsymbol{x}\geq\mathbf{0}\}$.*

Lemma 4 is well known in the field of combinatorial optimization and polyhedral combinatorics. You can find a proof of Lemma 4 in several books (e.g., Grötschel, Lovász & Schrijver [19]).

Now it is time to prove Theorem 2.

Proof of Theorem 2. Since it is known that the minimum coloring game on a perfect graph G is totally balanced [8], the core is the set of the optimal solutions of $(\mathsf{DLP}(G))$ by Lemma 3.

Let us denote by Feasible($(\mathsf{DLP}(G))$) the set of feasible solutions of $(\mathsf{DLP}(G))$. We claim that Feasible($(\mathsf{DLP}(G))$) $=Q(G)$ if G is a perfect graph. First, from Lemma 4, we can see that the set of constraints for $Q(G)$ includes those for Feasible($(\mathsf{DLP}(G))$). Therefore, it holds that Feasible($(\mathsf{DLP}(G))$) $\supseteq Q(G)$. To show the reverse inclusion, choose an arbitrary vector $\tilde{\boldsymbol{x}}\in$ Feasible($(\mathsf{DLP}(G))$). Namely, $\tilde{\boldsymbol{x}}$ satisfies that $\tilde{\boldsymbol{x}}(I)\leq 1$ for all maximal independent sets I of G and $\tilde{\boldsymbol{x}}\geq\mathbf{0}$. Now we have to show that $\tilde{\boldsymbol{x}}(I)\leq 1$ for all (not necessarily maximal) independent sets I of G. If I is maximal, we are done. If it is not, there exists a maximal independent set I' of G such that $I\subseteq I'$. Since $\tilde{\boldsymbol{x}}\geq\mathbf{0}$, it holds that $\tilde{\boldsymbol{x}}(I)\leq\tilde{\boldsymbol{x}}(I')\leq 1$. Hence we have Feasible($(\mathsf{DLP}(G))$) $\subseteq Q(P)$. This concludes the proof of the claim.

Let $G = (V, E)$ be a perfect graph. From the claim above, the problem $(\mathsf{DLP}(G))$ is seen to be equivalent to the following one:

$$\begin{aligned} (\mathsf{DLP'}(G))\text{:} \quad & \text{maximize} \quad && \sum_{v \in V} x_v \\ & \text{subject to} && \boldsymbol{x} \in Q(G). \end{aligned}$$

By Lemma 3, the core is the set of optimal solutions of $(\mathsf{DLP'}(G))$.

Consider the following $\{0,1\}$-version of $(\mathsf{DLP'}(G))$:

$$\begin{aligned} (\mathsf{DIP'}(G))\text{:} \quad & \text{maximize} \quad && \sum_{v \in V} x_v \\ & \text{subject to} && \boldsymbol{x} \in Q(G), \\ & && \boldsymbol{x} \in \{0,1\}^V. \end{aligned}$$

Then, we can find that the feasible solutions to $(\mathsf{DIP'}(G))$ exactly correspond to the cliques of G by the definition of $Q(G)$. Namely, the feasible solutions to $(\mathsf{DIP'}(G))$ are the characteristic vectors of the cliques of G. Since the objective function of $(\mathsf{DLP'}(G))$ is the sum of the components of the characteristic vector of a clique in G, namely the size of a clique, the optimal solutions of $(\mathsf{DIP'}(G))$ are the characteristic vectors of the maximum cliques of G.

Now let us go back to $(\mathsf{DLP'}(G))$. By the definitions of $(\mathsf{DLP'}(G))$ and $(\mathsf{DIP'}(G))$ and the claim above, we can see that the set of optimal solutions to $(\mathsf{DLP'}(G))$ is the convex hull of the set of optimal solutions to $(\mathsf{DIP'}(G))$. From the observation above, this is equivalent to saying that the set of optimal solutions of $(\mathsf{DLP'}(G))$ is the convex hull of the characteristic vectors of the maximum cliques of G. Hence, $\boldsymbol{x}$ is an optimal solution of $(\mathsf{DLP'}(G))$ if and only if $\boldsymbol{x}$ can be written as a convex combination of the characteristic vectors of the maximum cliques of G. □

Combining Theorem 2 with a result by Grötschel, Lovász & Schrijver [19], we conclude the following algorithmic consequences.

Corollary 5. *The following problems are solvable 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.*

Proof. The consequence (1) is due to Theorem 2 and the fact that a maximum clique can be computed in polynomial time for a perfect graph. The consequence (2) is due to Theorem 2 and the fact that we can solve the membership problem for the polytope $Q(G)$, when G is perfect, in polynomial time. Both facts are explained by Grötschel, Lovász & Schrijver [19]. □

3.3 The nucleolus of a minimum coloring game

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

Let $G = (V, E)$ be a perfect graph, $\mathcal{I}_G$ be the family of nonempty (not necessarily maximal) independent sets of G, and $K_1, K_2, \dots, K_k$ be the maximum cliques of G. For every nonempty independent set $I \in \mathcal{I}_G$, define

$$g(I) := |\{i \in \{1, \dots, k\} : |I \cap K_i| = 1\}|.$$

Then, consider the set $\Gamma := \{g(I) : I \in \mathcal{I}_G\}$. Note that Γ is a set, not a multiset, so it may happen that $|\Gamma| < |\mathcal{I}_G|$ when some independent sets have identical $g(I)$'s. Let us enumerate the elements of Γ in decreasing order as $g_1 > g_2 > \cdots > g_{|\Gamma|}$. The following lemma shows $g_1 = k$.

Lemma 6. *For a graph G with $\chi(G) = \omega(G)$, there exists an independent set of G which has a nonempty intersection with each of the maximum cliques of G.*

Proof. Let $G = (V, E)$ be a graph with $\chi(G) = \omega(G)$, and consider a minimum coloring of G. Then, take the vertices colored by an identical color. Denote by I the set of these vertices. Clearly, I is an independent set. Moreover, in each maximum clique K of G all colors used to color G should be found since $\chi(G) = \omega(G) = |K|$. Namely, K intersects I. □

On the other hand, we have less control for g_2 in general. This motivates the definition of goodness. We call a perfect graph G *good* if either $|\Gamma| = 1$ 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 $|\{i \in \{1, \ldots, k\} \setminus \{j\} : |I^{(j)} \cap K_i| = 1\}| = g_2$. Note that $|\Gamma| = 1$ if and only if the considered perfect graph is complete.

The main theorem on the nucleolus is as follows. Remember that the *barycenter* of vectors $\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^m$ is defined as $\sum_{i=1}^{m} \boldsymbol{x}^i/m$.

Theorem 7. *The nucleolus of the minimum coloring game on a good perfect graph is the barycenter of the characteristic vectors of the maximum cliques of the graph.*

To prove Theorem 7, first we show that we can restrict our attention to the so-called essential coalitions. Let (N, γ) be a cost game. A nonempty set $S \subseteq N$ is called an *essential coalition* if, for every proper partition $\mathcal{P}$ of S, we have $\gamma(S) < \sum\{\gamma(P) : P \in \mathcal{P}\}$. (Notice that if $|S| = 1$ then S is an essential coalition.) Huberman [21] showed that when we compute the nucleolus through a sequence of the linear programming problems (P_1), (P_2), ..., we only have to take the essential coalitions into account rather than handling all nonempty proper subsets of N. Namely, instead of considering $\mathcal{C}_0$ in (P_1), we consider the family of essential coalitions, and keep the same track as the original case through the computation. The next lemma characterizes the essential coalitions of a minimum coloring game.

Lemma 8. *Let $G = (V, E)$ be a graph (not necessarily perfect). Then $S \subseteq V$ is an essential coalition of the minimum coloring game (V, χ_G) on G if and only if S is a nonempty independent set of G.*

Proof. We show the if-part first. Assume that S is a nonempty independent set of G. Then, every nonempty proper subset P of S satisfies that $\chi_G(P) = 1$ since P is also an independent set of G. Therefore, for every proper partition $\mathcal{P}$ of S it holds that $\chi_G(S) = 1 < 2 \leq |\mathcal{P}| = \sum\{\chi_G(P) : P \in \mathcal{P}\}$. Hence, S is an essential coalition.

Next we show the converse. Assume that S is not an independent set of G. Then we have $\chi_G(S) \geq 2$. Fix a minimum coloring of $G[S]$. Then the color classes of this minimum coloring yield a proper partition of S since $\chi_G(S) \geq 2$. Let us denote this proper partition by $\mathcal{P}$. Since $\chi_G(P) = 1$ for each $P \in \mathcal{P}$, it follows that $\chi_G(S) = |\mathcal{P}| = \sum\{\chi_G(P) : P \in \mathcal{P}\}$. Thus we can see that S is not an essential coalition. □

With the help of Lemma 8, the problem (P_1) can be written as follows.

$$
\begin{aligned}
(\mathsf{P}_1):\quad & \text{maximize} && \epsilon \\
& \text{subject to} && \boldsymbol{x}(V) = \chi_G(V), \\
& && \boldsymbol{x}(I) \leq 1 - \epsilon \qquad \text{for all } I \in \mathcal{I}_0,
\end{aligned}
$$

where $\mathcal{I}_0 = \mathcal{I}_G$ is the family of all nonempty independent sets of G. Now we consider the optimal solutions of (P_1).

Lemma 9. *Let $G = (V, E)$ be a perfect graph and consider the minimum coloring game (V, χ_G) on G. Then a feasible solution $(\boldsymbol{x}, \epsilon) \in \mathbb{R}^V \times \mathbb{R}$ to the problem (P_1) is optimal if and only if $\boldsymbol{x}$ belongs to the core of (V, χ_G) and $\epsilon = 0$.*

Proof. First, notice that for any feasible solution $(\boldsymbol{x}, \epsilon)$ to (P_1), $\boldsymbol{x}$ belongs to the core if $\epsilon \geq 0$. This is a direct consequence from Section 3.2. Furthermore, if $\boldsymbol{x}$ belongs to the core and $\epsilon = 0$, then $(\boldsymbol{x}, \epsilon)$ is a feasible solution to (P_1). This means that the optimal value of (P_1) is at least zero, and if it is zero then for every optimal solution $(\boldsymbol{x}, 0)$ the vector $\boldsymbol{x}$ must belong to the core. Therefore, it suffices to show that the optimal value of (P_1) is indeed zero.

For contradiction, suppose that the optimal value is more than zero; namely, there exists a feasible solution $(\boldsymbol{x}, \epsilon)$ to (P_1) such that $\epsilon > 0$. From the observation above, $\boldsymbol{x}$ should belong to the core. By Theorem 2, $\boldsymbol{x}$ can be written as a convex combination of the characteristic vectors of the maximum cliques $K_1, \ldots, K_k$ of G since G is perfect. Let

$$
\boldsymbol{x} = \sum_{i=1}^{k} \lambda_i \mathbb{1}_{K_i},
$$

where $\lambda_1 + \cdots + \lambda_k = 1$ and $\lambda_i \geq 0$ for all $i \in \{1, \ldots, k\}$. Now, we use Lemma 6, which states that there exists an independent set I which has a nonempty intersection with every maximum clique. Namely, we have $|K_i \cap I| = 1$ for all $i \in \{1, \ldots, k\}$. Therefore, it follows that

$$
\boldsymbol{x}(I) = \sum_{i=1}^{k} \lambda_i \mathbb{1}_{K_i}(I) = \sum_{i=1}^{k} \lambda_i |K_i \cap I| = \sum_{i=1}^{k} \lambda_i = 1 > 1 - \epsilon.
$$

Hence, $(\boldsymbol{x}, \epsilon)$ violates the constraint $\boldsymbol{x}(I) \leq 1 - \epsilon$ of (P_1). This is a contradiction to the feasibility of $(\boldsymbol{x}, \epsilon)$. Thus we conclude that the optimal value of (P_1) is zero. $\square$

Notice that so far we did not use goodness, but just the perfectness of G.

If a perfect graph G has only one maximum clique, then we can see that (P_1) has a unique optimal solution from Lemma 9 and Theorem 2. In such a case, the characteristic vector of this unique maximum clique is the nucleolus. Thus Theorem 7 holds for this case. (Indeed, this can be seen directly from Theorem 2 and the fact that the nucleolus belongs to the core.) Hence, we now consider the case in which a perfect graph G has at least two maximum cliques. Then we have to consider the second problem (P_2). To solve (P_2), we must know what is the set

$$
\mathcal{I}_1 := \{I \in \mathcal{I}_0 : \boldsymbol{x}(I) = 1 \text{ for all optimal solutions } (\boldsymbol{x}, 0) \text{ of } (\mathsf{P}_1)\},
$$

which is the analogue of $\mathcal{C}_1$ in the definition of the nucleolus. The next lemma characterizes $\mathcal{I}_1$.

Lemma 10. *For the minimum coloring game on a perfect graph G, we have*

$$\mathcal{I}_1 = \{I \in \mathcal{I}_0 : |K_i \cap I| = 1 \text{ for all } i \in \{1, \dots, k\}\},$$

where we denote all the maximum cliques of G by $K_1, \dots, K_k$.

Proof. From the definition of $\mathcal{I}_1$ and Lemma 9, it holds that

$$\mathcal{I}_1 = \{I \in \mathcal{I}_0 : \boldsymbol{x}(I) = 1 \text{ for all core allocations } \boldsymbol{x}\}.$$

Let $I \in \mathcal{I}_0$. First assume that $\boldsymbol{x}(I) = 1$ for all core allocations $\boldsymbol{x}$. Let K_i be an arbitrary maximum clique of G. Since $\mathbb{1}_{K_i}$ is a core allocation by Theorem 2, we have $|K_i \cap I| = \mathbb{1}_{K_i}(I) = 1$. Thus, I satisfies that $|K_i \cap I| = 1$ for all $i \in \{1, \dots, k\}$.

Conversely, assume that $|K_i \cap I| = 1$ for all $i \in \{1, \dots, k\}$. Take an arbitrary core allocation $\boldsymbol{x}$. By Theorem 2, $\boldsymbol{x}$ can be written as a convex combination of the characteristic vectors of the maximum cliques of G. Let

$$\boldsymbol{x} = \sum_{i=1}^{k} \lambda_i \mathbb{1}_{K_i},$$

where $\lambda_1 + \cdots + \lambda_k = 1$ and $\lambda_i \geq 0$ for all $i \in \{1, \dots, k\}$. Then it holds that

$$\boldsymbol{x}(I) = \sum_{i=1}^{k} \lambda_i \mathbb{1}_{K_i}(I) = \sum_{i=1}^{k} \lambda_i |K_i \cap I| = \sum_{i=1}^{k} \lambda_i = 1.$$

(Here, the third identity is due to our assumption that $|K_i \cap I| = 1$ for all $i \in \{1, \dots, k\}$.) Thus, we can see that I satisfies that $\boldsymbol{x}(I) = 1$ for all core allocations $\boldsymbol{x}$. To conclude, we have shown that $\mathcal{I}_1 = \{I \in \mathcal{I}_0 : |K_i \cap I| = 1 \text{ for all } i \in \{1, \dots, k\}\}$. □

Note that Lemma 6 ensures the nonemptiness of $\mathcal{I}_1$.

Now we are ready to solve (P_2). To make it clearer to us, we write down (P_2) here.

$$\begin{array}{lll} (\mathsf{P}_2)\text{:} & \text{maximize} & \epsilon \\ & \text{subject to} & \boldsymbol{x}(V) = \chi_G(V), \\ & & \boldsymbol{x}(I) = 1 \qquad \text{for all } I \in \mathcal{I}_1, \\ & & \boldsymbol{x}(I) \leq 1 - \epsilon \qquad \text{for all } I \in \mathcal{I}_0 \setminus \mathcal{I}_1. \end{array}$$

The following lemma tells us, surprisingly, (P_2) already gives the nucleolus for a good perfect graph. (That is why we call the graph "good!")

Lemma 11. *For the minimum coloring game on a good perfect graph G with at least two maximum cliques, the problem (P_2) has a* unique *optimal solution $(\boldsymbol{\nu}, (k - g_2)/k)$, where $\boldsymbol{\nu}$ is the barycenter of the characteristic vectors of the maximum cliques of G.*

Note that Lemma 11 completes the proof of Theorem 7.

Proof. First, observe that we can write $\boldsymbol{\nu}$ as

$$\boldsymbol{\nu} = \sum_{i=1}^{k} \frac{1}{k} \mathbb{1}_{K_i}.$$

Now we show the following claim.

Claim 11.1. *The vector* $(\boldsymbol{\nu}, (k - g_2)/k)$ *is a feasible solution to* (P_2).

Proof of Claim 11.1. The constraint "$\boldsymbol{x}(V) = \chi_G(V)$" can be verified as

$$\boldsymbol{\nu}(V) = \sum_{i=1}^{k} \frac{1}{k} \mathbb{1}_{K_i}(V) = \sum_{i=1}^{k} \frac{1}{k} |K_i| = \sum_{i=1}^{k} \frac{1}{k} \chi_G(V) = \chi_G(V),$$

since the definition of a perfect graph gives us $\chi_G(V) = |K_i|$ for all $i \in \{1, \ldots, k\}$.

The constraint "$\boldsymbol{x}(I) = 1$ for all $I \in \mathcal{I}_1$" can be verified as follows.

$$\boldsymbol{\nu}(I) = \sum_{i=1}^{k} \frac{1}{k} \mathbb{1}_{K_i}(I) = \sum_{i=1}^{k} \frac{1}{k} |K_i \cap I| = \sum_{i=1}^{k} \frac{1}{k} = 1.$$

At the third equality, we used Lemma 10. For the constraint "$\boldsymbol{x}(I) \leq 1 - \epsilon$ for all $I \in \mathcal{I}_0 \setminus \mathcal{I}_1$," we again use Lemma 10. By Lemma 10 and the definition of g_2, for every $I \in \mathcal{I}_0 \setminus \mathcal{I}_1$, it holds that $|\{i : |K_i \cap I| = 1\}| \leq g_2$. Therefore, we obtain

$$\boldsymbol{\nu}(I) = \sum_{i=1}^{k} \frac{1}{k} \mathbb{1}_{K_i}(I) = \sum_{i=1}^{k} \frac{1}{k} |K_i \cap I| \leq \frac{g_2}{k} = 1 - \frac{k - g_2}{k}.$$

This concludes the proof of the claim. ◻

Let us get back to the proof of Lemma 11. The next step is to perturb $(\boldsymbol{\nu}, (k - g_2)/k)$ a little bit to get another vector and show that the perturbed vector should have a smaller objective value than $(k - g_2)/k$ as long as it is feasible. To do that, first let us fix an index $j \in \{1, \ldots, k\}$ arbitrarily and consider a vector $\boldsymbol{x}_\delta^{(j)}$ defined as

$$\boldsymbol{x}_\delta^{(j)} := \sum_{i \neq j} \left(\frac{1}{k} + \delta \right) \mathbb{1}_{K_i} + \left(\frac{1}{k} - (k-1)\delta \right) \mathbb{1}_{K_j},$$

where δ is a very small positive real number. Note that, by the assumption that G has at least two maximum cliques (i.e., $k \geq 2$), the vector $\boldsymbol{x}_\delta^{(j)}$ is well-defined. Furthermore, observe that $\boldsymbol{x}_\delta^{(j)}$ is a convex combination of the characteristic vectors of the maximum cliques of G since δ is very small. Therefore, $\boldsymbol{x}_\delta^{(j)}$ is a core allocation (by Theorem 2). Now suppose that $(\boldsymbol{x}_\delta^{(j)}, \epsilon)$ is a feasible solution to (P_2) for some $\epsilon \geq (k - g_2)/k$.

It is time to use the goodness of G. By the goodness of G, there exists an independent set $I^{(j)}$ of G such that $|I^{(j)} \cap K_j| = 0$ and $|\{i \in \{1, \ldots, k\} \setminus \{j\} : |I^{(j)} \cap K_i| = 1\}| = g_2$. From Lemma 10 and the definition of g_2, we can see that $I^{(j)} \notin \mathcal{I}_1$. Therefore, if $(\boldsymbol{x}_\delta^{(j)}, \epsilon)$ is a feasible solution to (P_2), then we should have that

$$\boldsymbol{x}_\delta^{(j)}(I^{(j)}) \leq 1 - \epsilon \leq 1 - \frac{k - g_2}{k} = \frac{g_2}{k}.$$

On the other hand, it follows that

$$\begin{aligned}
\boldsymbol{x}_\delta^{(j)}(I^{(j)}) &= \sum_{i\neq j}\left(\frac{1}{k}+\delta\right)\mathbb{1}_{K_i}(I^{(j)}) + \left(\frac{1}{k}-(k-1)\delta\right)\mathbb{1}_{K_j}(I^{(j)}) \\
&= \sum_{i\neq j}\left(\frac{1}{k}+\delta\right)|K_i\cap I^{(j)}| + \left(\frac{1}{k}-(k-1)\delta\right)|K_j\cap I^{(j)}| \\
&= g_2\left(\frac{1}{k}+\delta\right) > \frac{g_2}{k}.
\end{aligned}$$

They contradict each other. Hence, $(\boldsymbol{x}_\delta^{(j)}, \epsilon)$ cannot be feasible for any j unless $\epsilon < (k-g_2)/k$.

Now, observe that any feasible solution $(\boldsymbol{x}, \epsilon)$ to (P_2) close to $(\boldsymbol{\nu}, (k-g_2)/k)$ can be expressed as a convex combination of the vectors $(\boldsymbol{x}_\delta^{(j)}, \epsilon)$, $j \in \{1, \dots, k\}$. Hence, by the convexity of linear programming, we can conclude that $(\boldsymbol{\nu}, (k-g_2)/k)$ is a unique optimal solution of (P_2). □

We might wonder how naturally the good perfect graphs arise. In the next proposition, we show that several well-known classes of perfect graphs yield goodness.

Proposition 12. *The following perfect graphs are always good. (1) A perfect graph with only one maximum clique. (2) A forest. (3) A complete multipartite graph. (4) A chordal graph.*

Proof. (1) Immediately from the definition of goodness.

(2) Since a forest is a chordal graph, the statement follows from (4). However, this case is much simpler than the case of chordal graphs. So we would like to give such a simple proof here. We prove that for a tree. The proof for a forest can be similarly done.

Let G be a tree. In a tree, the maximum cliques are the edges as long as it has an edge. (If it does not have an edge, the statement immediately follows.) Let us choose an edge of G, say e. Then, delete e from G which results in two components. They are trees. Look at one of them. Let us denote this component by T. We can see that T contains a vertex r which was an endpoint of e in G. Now we consider the distance from r to each vertex of T, and collect all vertices in T to which the distance from r is an odd number. These vertices form an independent set of T, and have a nonempty intersection with all edges of T. We perform the same procedure for the other component. Then take the union of these two sets of vertices. This gives an independent set of G which has a nonempty intersection with all edges except for e. (Thus, g_2 is always the number of edges minus one for the tree.)

(3) Let G be a complete r-partite graph for some $r \geq 1$, and $V_1, V_2, \dots, V_r$ be a partition of the vertex set of G such that each V_i is independent. (Such a partition is unique by definition.) Also let $n_i := |V_i|$ for every $i \in \{1, \dots, r\}$. The first observation is that every independent set is a subset of V_i for some $i \in \{1, \dots, r\}$ due to the complete r-partiteness. Assume that an independent set I is a subset of V_i. Then, since every vertex of I is an element of $n_1 n_2 \cdots n_{i-1} n_{i+1} \cdots n_r$ maximum cliques of G, we have $g(I) = |I| n_1 n_2 \cdots n_{i-1} n_{i+1} \cdots n_r$. Thus we can see that $g_2 = \max_i n_1 n_2 \cdots n_{i-1} (n_i - 1) n_{i+1} \cdots n_r$. Let i^* be a maximizer of this expression.

Now consider a maximum clique K of G. Let v be a unique vertex lying on the intersection $V_{i^*} \cap K$. We choose the independent set $V_{i^*} \setminus \{v\}$. This set satisfies the desired property.

(4) We prove the following stronger statement.

Proposition 13. *Let G be a chordal graph and $K_1, K_2, \dots, K_k$ be the maximum cliques of G ($k \geq 1$). If $k \geq 2$, then for each $i \in \{1, \dots, k\}$ there exists an independent set $I^{(i)}$ of G such that $|I^{(i)} \cap K_i| = 0$ and $|\{j \in \{1, \dots, k\} \setminus \{i\} : |I^{(i)} \cap K_j| = 1\}| = k - 1$.*

We prove Proposition 13 by the induction on the number n of vertices of a chordal graph G. If $n = 1$, then $k = 1$. Therefore the statement is true.

Consider the case in which $n > 1$, and let us assume that the statement is true for all $n' < n$. Now we invoke a property of chordal graphs due to Dirac [10]. Remember that a vertex of a graph is called *simplicial* if the neighborhood of the vertex forms a clique.

Lemma 14 (Dirac [10]). *A chordal graph has a simplicial vertex.*

Let $G = (V, E)$ be a chordal graph with n vertices such that the number of maximum cliques is at least two (i.e., $k \geq 2$). By Lemma 14, G has a simplicial vertex v. Then $T := N_G[v]$ is a maximal clique of G (by the definitions of a simplicial vertex and a neighbor). Now we consider the following two cases.

Case 1: T is not a maximum clique of G. Then the maximum cliques of $G[V \setminus \{v\}]$ are exactly those of G. So, by applying the induction hypothesis to $G[V \setminus \{v\}]$, we obtain for each $i \in \{1, \dots, k\}$ there exists an independent set $I^{(i)}$ of $G[V \setminus \{v\}]$ such that $|I^{(i)} \cap K_i| = 0$ and $|\{j \in \{1, \dots, k\} \setminus \{i\} : |I^{(i)} \cap K_j| = 1\}| = k - 1$. Since $I^{(i)}$ is also an independent set of G, this is exactly what we have wanted.

Case 2: T is a maximum clique of G. Without loss of generality, let $T = K_k$. Then the maximum cliques of $G[V \setminus \{v\}]$ are $K_1, K_2, \dots, K_{k-1}$. Now we have two subcases.

Case 2-1: $k = 2$. Namely, G has two maximum cliques K_1 and $K_2 = T$. Then there exist vertices $v_1 \in K_1 \setminus K_2$ and $v_2 \in K_2 \setminus K_1$. The sets $I^{(1)} := \{v_2\}$ and $I^{(2)} := \{v_1\}$ satisfy the desired property.

Case 2-2: $k > 2$. Then, by induction hypothesis, for each $i \in \{1, \dots, k-1\}$ there exists an independent set $J^{(i)}$ of $G[V \setminus \{v\}]$ such that $|J^{(i)} \cap K_i| = 0$ and $|\{j \in \{1, \dots, k-1\} \setminus \{i\} : |J^{(i)} \cap K_j| = 1\}| = k - 2$. Now we turn $J^{(i)}$ into $I^{(i)}$ which we have wanted to obtain. In case $J^{(i)} \cap N_G(v) = \emptyset$, set $I^{(i)} := J^{(i)} \cup \{v\}$, which is an independent set of G. Then, it follows that $|I^{(i)} \cap K_i| = |(J^{(i)} \cup \{v\}) \cap K_i| = 0$ and

$$
\begin{aligned}
&|\{j \in \{1, \dots, k\} \setminus \{i\} : |I^{(i)} \cap K_j| = 1\}| \\
&\quad = |\{j \in \{1, \dots, k-1\} \setminus \{i\} : |I^{(j)} \cap K_j| = 1\}| + 1 \\
&\quad = |\{j \in \{1, \dots, k-1\} \setminus \{i\} : |(J^{(j)} \cup \{v\}) \cap K_j| = 1\}| + 1 \\
&\quad = |\{j \in \{1, \dots, k-1\} \setminus \{i\} : |J^{(j)} \cap K_j| = 1\}| + 1 \\
&\quad = (k-2) + 1 \\
&\quad = k - 1
\end{aligned}
$$

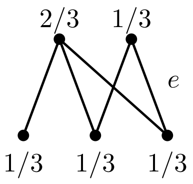

Figure 1: A bipartite graph which is not good.

because $I^{(i)} \cap K_k = (J^{(i)} \cup \{v\}) \cap (N_G(v) \cup \{v\}) = \{v\}$. In case $J^{(i)} \cap N_G(v) \neq \emptyset$, set $I^{(i)} := J^{(i)}$. Then it follows that $|I^{(i)} \cap K_i| = |J^{(i)} \cap K_i| = 0$ and

$$\begin{aligned}
&|\{j \in \{1, \dots, k\} \setminus \{i\} : |I^{(i)} \cap K_j| = 1\}| \\
&\quad = |\{j \in \{1, \dots, k-1\} \setminus \{i\} : |I^{(j)} \cap K_j| = 1\}| + 1 \\
&\quad = |\{j \in \{1, \dots, k-1\} \setminus \{i\} : |J^{(j)} \cap K_j| = 1\}| + 1 \\
&\quad = (k-2) + 1 \\
&\quad = k - 1
\end{aligned}$$

because it follows that $I^{(i)} \cap K_k = J^{(i)} \cap (N_G(v) \cup \{v\}) = J^{(i)} \cap N_G(v) \neq \emptyset$.

Finally, we have to find an independent set $I^{(k)}$ of the graph G such that $|I^{(k)} \cap T| = 0$ and $|\{j \in \{1, \dots, k-1\} : |I^{(i)} \cap K_j| = 1\}| = k - 1$. Remembering the proof of Lemma 6, we can find an independent set I of G such that $v \in I$ and I has a nonempty intersection with every maximum clique K_ℓ ($\ell \in \{1, \dots, k\}$). Now set $I^{(k)} := I \setminus \{v\}$. Then we can check that $I^{(k)}$ satisfies the conditions above. This concludes the proof of Proposition 13. □

Note that there exists a bipartite graph which is not good and for which the nucleolus is not given as in Theorem 7. Consider the bipartite graph in Figure 1. The associated numbers express the components of the nucleolus of the minimum coloring game on this graph. We can see that this is not the barycenter of the characteristic vectors of the maximum cliques. (Note that, since it is bipartite, the maximum cliques are the edges.) We can also see that $g_2 = 4$. However, for the rightmost edge e, every independent set avoiding e can intersect at most 3 other edges. Therefore, this graph is not good.

Now we describe how to compute the nucleolus for classes of good perfect graphs in Proposition 12. In a forest $G = (V, E)$ with $E \neq \emptyset$, the maximum cliques are the edges of G. Let $d_G(v)$ be the degree of $v \in V$ in G. Since a forest is good (Proposition 12.2), Theorem 7 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.

Let $G = (V, E)$ be a complete r-partite graph in which V is partitioned into $V_1, \dots, V_r$ and $E = \{\{u, v\} : u \in V_i,\ v \in V_j,\ i, j \in \{1, \dots, r\},\ i \neq j\}$. Then a maximum clique K is a vertex subset which satisfies that $|V_i \cap K| = 1$ for all $i \in \{1, \dots, r\}$. Since a complete multipartite graph is good (Proposition 12.3), with the help of Theorem 7, the v-th component of the nucleolus for the complete r-partite graph G can be computed as $1/|V_i|$ if $v \in V_i$. In a past work, the author investigated the Shapley value for a complete multipartite graph [32]. Since this is exactly the same as the expression above, we immediately obtain the following corollary.

Corollary 15. *The Shapley value and the nucleolus coincide for the minimum coloring game on a complete multipartite graph.* $\square$

Note that this is not true for a general submodular game.

For a chordal graph, we are not aware of a closed formula for the nucleolus. However, still we can 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 a book by Golumbic [18]), which is based on a proposition by Fulkerson & Gross [16]. (Note that the number of the maximal cliques of a chordal graph is bounded by the number of the vertices from above [16].) Therefore, from this enumeration, we are able to obtain the list of all maximum cliques of the chordal graph. Then, since a chordal graph is good (Proposition 12.4), we can compute the nucleolus from this list by utilizing Theorem 7. Thus we obtain the following corollary.

Corollary 16. *The nucleolus of the minimum coloring game on a chordal graph can be computed in polynomial time.* $\square$

3.4 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 a polynomial-time algorithm for the submodular function minimization. However, for minimum coloring games, this only works for complete multipartite graphs because of the characterization due to Okamoto [32]. In this subsection, we describe how to compute the τ-values in polynomial time for perfect graphs. Note that the τ-value for a perfect graph is well-defined since the minimum coloring game on a perfect graph is totally balanced [8]. The next theorem is crucial.

Theorem 17. *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, for each $v \in V$,*

$$\underline{m}_v = \begin{cases} 1 & (v \in K) \\ 0 & (\textit{otherwise}); \end{cases}$$

$$\overline{m}_v = \begin{cases} 1 & (v \in K \textit{ or } K \subseteq N_G(v)) \\ 0 & (\textit{otherwise}). \end{cases}$$

Proof. First we prove the formula for $\underline{\boldsymbol{m}}$. Assume that $v \in K$. Then $\omega(G[V \setminus \{v\}]) = \omega(G) - 1$. Therefore, we have $\underline{m}_v = \chi_G(V) - \chi_G(V \setminus \{v\}) = \omega(G) - \omega(G[V \setminus \{v\}]) = 1$ by the perfectness of G. On the other hand, assume that there exists an index $i \in \{1, \ldots, k\}$ such that $v \notin K_i$. Then K_i is still a maximum clique of $G[V \setminus \{v\}]$. Therefore, by the perfectness of G it holds that $\underline{m}_v = \chi_G(V) - \chi_G(V \setminus \{v\}) = \omega(G) - \omega(G[V \setminus \{v\}]) = 0$.

Next we prove the formula for $\overline{\boldsymbol{m}}$. First of all, we can see that

$$\begin{aligned} \overline{m}_v &= \min\{\chi_G(S) - \underline{\boldsymbol{m}}(S \setminus \{v\}) : v \in S \subseteq V\} \\ &\leq \chi_G(\{v\}) - \underline{\boldsymbol{m}}(\{v\} \setminus \{v\}) \\ &= \chi(G[\{v\}]) = 1. \end{aligned}$$

Here, from the first line to the second line, we have chosen $\{v\}$ as S. Now assume that $v \in K$. From the proof of the first half, we have $\underline{m}_v = 1$. Hence it holds that $1 = \underline{m}_v \leq \overline{m}_v \leq 1$ since the minimum coloring game on a perfect graph is balanced (implying that $\underline{m}_v \leq \overline{m}_v$). This concludes that $\overline{m}_v = 1$ if $v \in K$. Next assume that there exists an index $i \in \{1, \dots, k\}$ such that $v \notin K_i$. From the proof of the first half, we have $\underline{m}_v = 0$. If there exists a vertex $u \in K$ which is not a neighbor of v, then it follows that

$$\begin{aligned}
\overline{m}_v &= \min\{\chi_G(S) - \underline{\boldsymbol{m}}(S \setminus \{v\}) : v \in S \subseteq V\} \\
&\leq \chi_G(\{u, v\}) - \underline{\boldsymbol{m}}(\{u, v\} \setminus \{v\}) \\
&= \chi(G[\{u, v\}]) - \underline{m}_u \\
&= 1 - 1 = 0.
\end{aligned}$$

Here, from the first line to the second line, we have chosen $\{u, v\}$ as S, and from the third line to the fourth line, we used $\underline{m}_u = 1$, which follows from $u \in K$ and the first half of this theorem. Consider the case in which all the vertices in K are neighbors of v (i.e., $K \subseteq N_G(v)$). Suppose that $\overline{m}_v = 0$. This means that there exists $S \subseteq V$ such that $v \in S$ and $\chi_G(S) - \underline{\boldsymbol{m}}(S \setminus \{v\}) = 0$ by the definition of $\overline{\boldsymbol{m}}$. From the perfectness of G, we have

$$\begin{aligned}
0 &= \chi_G(S) - \underline{\boldsymbol{m}}(S \setminus \{v\}) \\
&= \omega(G[S]) - (\underline{\boldsymbol{m}}((S \setminus \{v\}) \cap K) + \underline{\boldsymbol{m}}((S \setminus \{v\}) \setminus K)) \\
&= \omega(G[S]) - (|(S \setminus \{v\}) \cap K| + 0) \\
&= \omega(G[S]) - |S \cap K|.
\end{aligned}$$

(Here, for the third equality we use the formula for $\underline{\boldsymbol{m}}$ which we have already proved. For the fourth equality we use the assumption that $v \notin K$.) Thus we can see that $\omega(G[S]) = |S \cap K|$. Since $S \cap K$ is a clique in G, so is in $G[S]$ as well. This means that $S \cap K$ is a maximum clique of $G[S]$. However, $S \cap (K \cup \{v\})$ is also a clique of $G[S]$ because $K \subseteq N_G(v)$. Since $v \notin S \cap K$, we can also see that $\omega(G[S]) = |S \cap K| < |S \cap (K \cup \{v\})|$. This is a contradiction to the maximality of $S \cap K$. Hence it follows that $\overline{m}_v = 1$ since $\overline{m}_v \leq 1$ and $\overline{\boldsymbol{m}}$ is integral. $\square$

Based on Theorem 17, we establish an algorithm to compute the τ-value of the minimum coloring game on a perfect graph. First we compute $\underline{\boldsymbol{m}}$. To do that, we 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 [19], $\underline{\boldsymbol{m}}$ can also be computed in polynomial time. From $\underline{\boldsymbol{m}}$, we can see what K is, by using Theorem 17, 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 17. Finally, we can also compute the appropriate λ in a straightforward manner because we have already known $\chi_G(V)$. Namely,

$$\lambda = \frac{\chi_G(V) - \overline{\boldsymbol{m}}(V)}{\underline{\boldsymbol{m}}(V) - \overline{\boldsymbol{m}}(V)}.$$

Thus we have obtained the τ-value and this algorithm runs in polynomial time. The dominant step for the running time is the computations of $\chi_G(V)$ and $\chi_G(V \setminus \{v\})$ for all $v \in V$. This gives the running time $O(nT_\chi(n))$, where $T_\chi(n)$ is the worst-case time complexity to compute the chromatic number of a perfect graph with n vertices.

Note that Theorem 17 enables us to compute the τ-value for a complete multipartite graph without using an algorithm for the submodular function minimization.

3.5 The Shapley value of a minimum coloring game

In the former work [32], the author provided the formula of the Shapley value for a complete multipartite graph which can be easily evaluated (see also Corollary 15). Here we 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, the following lemma holds.

Lemma 18. *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.

Proof. Recall the definition of the Shapley value $\boldsymbol{\phi}$ of the minimum coloring game (V, χ_G):

$$\phi_v = \sum_{S \subseteq V \setminus \{v\}} \frac{|S|!(|V|-|S|-1)!}{|V|!} (\chi_G(S \cup \{v\}) - \chi_G(S)).$$

Since G is a bipartite graph, we have

$$\chi_G(S) = \begin{cases} 0 & (S = \emptyset) \\ 1 & (S \text{ is nonempty and independent}) \\ 2 & (\text{otherwise}) \end{cases}$$

for each $S \subseteq V$. Therefore, it follows that

$$\chi_G(S \cup \{v\}) - \chi_G(S) = \begin{cases} 1 & (S = \emptyset \text{ or } S \text{ is independent, but } S \cup \{v\} \text{ is not}) \\ 0 & (\text{otherwise}) \end{cases}$$

for each $S \subseteq V \setminus \{v\}$.

So now, we manipulate the v-th component of the Shapley value as follows:

$$\begin{aligned} \phi_v &= \sum_{S \subseteq V \setminus \{v\}} \frac{|S|!(|V|-|S|-1)!}{|V|!} (\chi_G(S \cup \{v\}) - \chi_G(S)) \\ &= \frac{|\emptyset|!(|V|-|\emptyset|-1)!}{|V|!} + \sum_{\substack{S \subseteq V \setminus \{v\}: S \text{ independent}, \\ S \cup \{v\} \text{ not independent}}} \frac{|S|!(|V|-|S|-1)!}{|V|!} \\ &= \frac{1}{n} + \sum_{k=1}^{n-1} a_v^k(G) \frac{k!(n-k-1)!}{n!}. \end{aligned}$$

This concludes the proof. ☐

Therefore, if we have a polynomial-time algorithm to compute $a_v^k(G)$ for every $k \in \{1, \dots, n-1\}$, we are able to obtain the Shapley value in polynomial time when G is bipartite. However we are not aware of such a procedure for a general bipartite graph. Actually, we have an easy 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 [31], even if it is in addition planar and with maximum degree 4 [40].)

Proposition 19. *The following problem is #P-complete. Given an n-vertex planar bipartite graph G with maximum degree 4 and $k \in \{1, \dots, n-1\}$ as an input, compute the value $a_v^k(G)$.*

Proof. Clearly, the problem belongs to #P. So we only have to show the #P-hardness of the problem. For this purpose, we use the #P-complete problem of counting the number of independent sets in a planar bipartite graph with maximum degree 4 [40]. Let $H = (V_H, E_H)$ be an arbitrarily given planar bipartite graph with $n-2$ vertices and with maximum degree 4.

Now we construct a bipartite graph $G = (V_G, E_G)$ as follows: $V_G = V_H \cup \{x, y\}$ and $E_G = E_H \cup \{\{x, y\}\}$. (Here, we assume that $\{x, y\} \cap V_H = \emptyset$.) Apparently, G is still planar and bipartite, and has maximum degree 4. We can see that, for each $k \in \{1, \dots, n-1\}$, $a_x^k(G)$ is the number of independent sets in G of size k which contains y and does not contain x, and this is exactly the same as the number of independent sets in H of size $k-1$ by the construction. Therefore, the number of independent sets in H is equal to $\sum_{k=1}^{n-1} a_x^k(G)$. This concludes the reduction. $\square$

Therefore, we 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 $k \in \{1, \dots, n-1\}$ when G is a forest with n vertices.

To do that, let us first fix $k \in \{1, \dots, n-1\}$ and $v \in V(G)$, and concentrate on computing $a_v^k(G)$. Let $N_G(v) = \{v_1, v_2, \dots, v_d\}$. Every independent set I of G such that $v \notin I$ and $I \cup \{v\}$ is not independent must contain at least one of $v_1, \dots, v_d$. If I contains v_1, then it cannot contain any vertices in $N_G(v_1)$, since it is independent. Namely, it must contain other $k-1$ vertices among $V(G) \setminus N_G[v_1]$. (Remember that $N_G[v_1]$ denotes the closed neighborhood of v_1 in G.) If I does not contain v_1, then I is an independent set of size k in $G[V(G) \setminus \{v_1\}]$ which contains at least one of $v_2, \dots, v_d$. Therefore, this consideration gives the following formula:

$$a_v^k(G) = I^{k-1}(G[V(G) \setminus N_G[v_1]]) + a_v^k(G[V(G) \setminus \{v_1\}]),$$

where $I^k(G)$ represents the number of independent sets of size k in G. Now, applying the same trick to the second term of the right-hand side, we obtain the following lemma.

Lemma 20. *In the setting above, let G_i be the graph obtained from G by removing the vertices in $\{v_1, \dots, v_{i-1}\} \cup N_G[v_i]$. Then, it follows that*

$$a_v^k(G) = \sum_{i=1}^{d} I^{k-1}(G_i).$$

Note that, when G is a forest, G_i is also a forest for each $i \in \{1, \dots, d\}$. Therefore, to compute $a_v^k(G)$ in polynomial time, it is enough to calculate the number of independent sets of a given size for a forest in polynomial time.

To compute $I^k(G)$, we use the generating function method. Define the following polynomial

$$I(G; x) := \sum_{k=0}^{n} I^k(G) x^k,$$

where n is the number of vertices in G, and x is an indeterminate. We call $I(G;x)$ the *independent set polynomial* of G. (This concept seems to be first defined by Hoede & Li [20].) If we can compute $I(G;x)$ in polynomial time, then, from $I(G;x)$, we can obtain $I^k(G)$ for every $k \in \{0,\dots,n\}$. Therefore, we now concentrate on a polynomial-time computation of $I(G;x)$ when G is a forest.

For independent set polynomials, the following two facts are known.

Lemma 21 (Hoede & Li [20]). *Let $G=(V,E)$ be a graph, which is not necessarily a forest.*

1. *If G is disconnected and $G_1,\dots,G_\ell$ are the connected components of G, then it holds that*

$$I(G;x) = \prod_{j=1}^{\ell} I(G_j;x).$$

2. *For each vertex $v \in V$, it holds that*

$$I(G;x) = I(G[V \setminus \{v\}];x) + xI(G[V \setminus N_G[v]];x).$$

Now, we concentrate on the computation of $I(G;x)$. From the first part of Lemma 21, it is enough to consider $I(G;x)$ when G is connected, i.e., G is a tree, as far as the polynomial-time computation is concerned. Therefore, we assume that G is a tree. Let us fix one vertex $r \in V(G)$ and regard it as a root of G. Then, we can naturally associate the parent-child relations among pairs of adjacent vertices. Denote by v_i the i-th child of r in G, and $v_{i,j}$ the j-th child of the i-th child of r in G, if exists. (So, $v_{i,j}$ is a grandchild of r.) Let $G^{[v]}$ be the subgraph induced by the set of descendants of v and v itself. Then, by the second part of Lemma 21, we have

$$I(G;x) = \prod_{i} I(G^{[v_i]};x) + x\prod_{i,j} I(G^{[v_{i,j}]};x).$$

Due to this recurrence, we can establish the bottom-up dynamic programming algorithm. Namely, first we compute $I(G^{[v]};x)$ for each vertex v farthest from r. Next, we compute it for the second farthest vertices, which only uses the outcome from the farthest vertices. In this way, we keep computing until getting to the root r. Thus, we obtain $I(G;x)$ in polynomial time, and it concludes the polynomial-time algorithm to compute the Shapley value of the minimum coloring game on a forest.

Let us notice that a recent work [33] gives a cubic-time algorithm to compute $I(G;x)$ when G is chordal. This generalizes the above result for forests to chordal graphs, but does not give the Shapley value for a chordal graph. That is because chordal graphs are not necessarily bipartite.

4 Conclusion

We have investigated some fair cost allocations (the core, the nucleolus, the τ-value, 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 problems. This was indeed true in the work by Deng, Ibaraki & Nagamochi [7], and a subsequent work by Bietenhader and the author [1] makes much use of this point of view when they study core stability of minimum coloring games on perfect graphs. 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. assignment games [38]) or the nucleolus is a unique vector in the intersection of the core and the kernel (e.g. submodular games [13, 24]). 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 7). Since we are not aware of the characterization of the nucleolus for a class of games as a certain convex combination of the extreme points of the core, this is a unique feature of the current work. This viewpoint 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 [27]. 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 mention some open questions which this paper raises. First of all, we do not know how to compute the nucleolus for a general perfect graph or even for a bipartite graph in polynomial time. It could be NP-hard or #P-hard. The situation is the same for the Shapley values. 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.

Acknowledgments

The discussion with Hiroaki Mohri for an article [29] stimulated this work. The author gratefully acknowledges the help rendered by him. In addition, the author is grateful to Udo Adamy, Kenji Kashiwabara, Tadashi Sakuma and anonymous referees for discussion and/or useful comments.

References

[1] Th. Bietenhader and Y. Okamoto. Core stability of minimum coloring games. Math. Oper. Res. **31** (2006) 418–431.

[2] O.N. Bondareva. Some applications of linear programming methods to the theory of cooperative games. Problemy Kibernet. **10** (1963) 119–139 (in Russian).

[3] M. Chudnovsky, G. Cornuéjols, X. Liu, P. Seymour and K. Vušković. Recognizing Berge graphs. Combinatorica **25** (2005) 143–186.

[4] M. Chudnovsky, N. Robertson, P. Seymour and R. Thomas. The strong perfect graph theorem. Ann. Math. **164** (2006) 51–229.

[5] V. Chvátal. On certain polytopes associated with graphs. J. Combin. Theory Ser. B **18** (1975) 138–154.

[6] I.J. Curiel. Cooperative Game Theory and Applications: Cooperative Games Arising from Combinatorial Optimization Problems. Kluwer Academic Publishers, Dordrecht, 1997.

[7] X. Deng, T. Ibaraki and H. Nagamochi. Algorithmic aspects of the core of combinatorial optimization games. Math. Oper. Res. **24** (1999) 751–766.

[8] X. Deng, T. Ibaraki, H. Nagamochi and W. Zang. Totally balanced combinatorial optimization games. Math. Program. **87** (2000) 441–452.
[9] X. Deng and Ch.H. Papadimitriou. On the complexity of cooperative solution concepts. Math. Oper. Res. **19** (1994) 257–266.
[10] G.A. Dirac. On rigid circuit graphs. Abh. Math. Sem. Univ. Hamburg **25** (1961) 71–76.
[11] T. Driessen. Cooperative Games, Solutions and Applications. Kluwer Academic Publishers, Dordrecht, 1988.
[12] U. Faigle and W. Kern. The Shapley value for cooperative games under precedence constraints. Internat. J. Game Theory **21** (1992) 249–266.
[13] 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.
[14] 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.
[15] D.R. Fulkerson. Anti-blocking polyhedra. J. Combin. Theory Ser. B **12** (1972) 50–71.
[16] D.R. Fulkerson and O.A. Gross. Incidence matrices and interval graphs. Pacific J. Math. **15** (1965) 835–855.
[17] D.B. Gillies. Some theorems on n-person games. Ph.D. Thesis, Princeton University, 1953.
[18] M.C. Golumbic. Algorithmic Graph Theory and Perfect Graphs. Second edition. Elsevier, Amsterdam, 2004.
[19] M. Grötschel, L. Lovász and A. Schrijver. Geometric algorithms and combinatorial optimization. Second edition. Springer-Verlag, Berlin, 1993.
[20] C. Hoede and X. Li. Clique polynomials and independent set polynomials of graphs. Discrete Math. **125** (1994) 219–228.
[21] G. Huberman. The nucleolus and the essential coalitions. Analysis and Optimization of Systems, Lecture Notes in Control and Information Sci. **28** (1980) 416–422.
[22] Y. Kannai. The core and balancedness. In: Handbook of Game Theory with Economic Applications Volume 1, R.J. Aumann and S. Hart eds., North Holland, Amsterdam, 1992, 355–395.
[23] 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.
[24] J. Kuipers. A polynomial time algorithm for computing the nucleolus of convex games. Report M 96-12, Maastricht University, 1996.
[25] L. Lovász. Normal hypergraphs and the perfect graph conjecture. Discrete Math. **2** (1972) 253–267.
[26] M. Maschler. The bargaining set, kernel, and nucleolus. In: Handbook of Game Theory with Economic Applications Volume 1, R.J. Aumann and S. Hart eds., North Holland, Amsterdam, 1992, 591–667.
[27] 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.
[28] N. Megiddo. Computational complexity and the game theory approach to cost allocation for a tree. Math. Oper. Res. **3** (1978) 189–196.
[29] H. Mohri and Y. Okamoto. Discrete optimization and cooperative games (2). Comm. Oper. Res. Soc. Japan **48** (2003) 114–120 (in Japanese).
[30] B. Peleg. Axiomatizations of the core. In: Handbook of Game Theory with Economic Applications Volume 1, R.J. Aumann and S. Hart eds., North Holland, Amsterdam, 1992, 397–412.

[31] 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.

[32] Y. Okamoto. Submodularity of some classes of the combinatorial optimization games. Math. Methods Oper. Res. **58** (2003) 131–139.

[33] Y. Okamoto, T. Uno and R. Uehara. Counting the number of independent sets in chordal graphs. J. Discrete Algorithms, to appear.

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

[35] L.S. Shapley. A value for n-person games. In: Contributions to the Theory of Games II, H. Kuhn and A.W. Tucker eds., Princeton University Press, Princeton, New Jersey, 1953, 307–317.

[36] L.S. Shapley. On balanced sets and cores. Naval Research Logistics Quarterly **14** (1967) 453–460.

[37] L.S. Shapley. Cores of convex games. Internat. J. Game Theory **1** (1971) 11–26. Errata is in the same volume, 1972, pp. 199.

[38] T. Solymosi and T.E.S. Raghavan. An algorithm for finding the nucleolus of assignment games. Internat. J. Game Theory **23** (1994) 119–143.

[39] S.H. Tijs. Bounds for the core and the τ-value. In: Game Theory and Mathematical Economics, O. Moeshlin and P. Pallaschke, eds., North Holland, Amsterdam, 1981, 123–132.

[40] S.P. Vadhan. The complexity of counting in sparse, regular, and planar graphs. SIAM J. Comput. **31** (2001) 398–427.

[41] E. Winter. The Shapley value. In: Handbook of Game Theory with Economic Applications Volume 3, R.J. Aumann and S. Hart eds., North Holland, Amsterdam, 2002, 2025–2054.