Coalition Structures in Weighted Voting Games

Edith Elkind and **Georgios Chalkiadakis** and **Nicholas R. Jennings** [1]

Abstract. Weighted voting games are a popular model of collaboration in multiagent systems. In such games, each agent has a weight (intuitively corresponding to resources he can contribute), and a coalition of agents wins if its total weight meets or exceeds a given threshold. Even though coalitional stability in such games is important, existing research has nonetheless only considered the stability of the grand coalition. In this paper, we introduce a model for weighted voting games with coalition structures. This is a natural extension in the context of multiagent systems, as several groups of agents may be simultaneously at work, each serving a different task. We then proceed to study stability in this context. First, we define the CS-core, a notion of the core for such settings, discuss its non-emptiness, and relate it to the traditional notion of the core in weighted voting games. We then investigate its computational properties. We show that, in contrast with the traditional setting, it is computationally hard to decide whether a game has a non-empty CS-core, or whether a given outcome is in the CS-core. However, we then provide an efficient algorithm that verifies whether an outcome is in the CS-core if all weights are small (polynomially bounded). Finally, we also suggest heuristic algorithms for checking the non-emptiness of the CS-core.

1 Introduction

Coalitional games [8] provide a rich framework for the study of cooperation both in economics and politics, and have been successfully used to model collaboration in multiagent systems [11, 3]. In such games, teams (or *coalitions*) of agents come together to achieve a common goal, and derive individual benefits from this activity.

A particularly simple, yet expressive, class of coalitional games is that of *weighted voting games (WVGs)* [13]. In a weighted voting game each player (or *agent*) has a weight, and a coalition *wins* if its members' total weight meets or exceeds a certain threshold, and loses otherwise. Weighted voting has straightforward applications in a plethora of societal and computer science settings ranging from real-life elections to computer operating systems, as well as a variety of settings involving multiagent coordination. In particular, an agent's weight can be thought of as the amount of resources available to this agent, and the threshold indicates the amount of resources necessary to achieve a task. A winning coalition then corresponds to a team of agents that can successfully complete this task.

Originally, research in weighted voting games was motivated by a desire to model decision-making in governmental bodies. In such settings, the threshold is usually at least 50% of the total weight, and the issues of interest relate to the distribution of payoffs within the *grand coalition*, i.e., the coalition of all agents. Perhaps for this reason, to date, all research on weighted voting games tacitly assumes that the grand coalition will form. However, in multiagent settings such as those described above, the threshold can be significantly smaller than 50% of the total weight, and several winning coalitions may be able to form simultaneously. Moreover, in this situation the formation of the grand coalition may not, in fact, be a desirable outcome: instead of completing several tasks, forming the grand coalition concentrates all agent resources on finishing a single task. In contrast, the overall efficiency will be higher if the agents form a *coalition structure (CS)*, i.e., a collection of several disjoint coalitions.

To model such scenarios, in this paper we introduce a model for *WVGs with coalition structures*. We then focus on the issue of *stability* in this setting. A structure is stable when rational agents are not motivated to depart from it, and thus they can concentrate on performing their task, rather than looking for ways to improve their payoffs. Therefore, stability provides a useful balance between individual goals and overall performance. To study it, we extend the notion of the *core*—a classic notion of stability for coalitional games—to our setting, by defining the *CS-core* for WVGs. We then provide a detailed study of this concept, comparing it with the classic core and analyzing its computational properties.

Our main contributions are as follows: (1) we define a new model that allows weighted voting games to admit coalition structures (Sec. 3); (2) we define the CS-core for such games, relate it to the classic core, and describe sufficient conditions for its non-emptiness (Sec. 4); (3) we show that several natural CS-core-related problems are intractable—namely, it is NP-hard to decide the non-emptiness of the CS-core and coNP-complete to check whether a given outcome is in the CS-core (Sec. 5). Interestingly, this contrasts with what holds in weighted voting games without coalition structures, where both of these problems are polynomial-time solvable; (4) we provide a polynomial-time algorithm to check if a given outcome is in the CS-core in the important special case of polynomially-bounded weights. We then show how to use this algorithm to efficiently check if a given coalition structure admits a stable payoff distribution, and suggest a heuristic algorithm to find an allocation in the core (Sec. 6). We begin with some background and a brief review of related work.

2 Background and Related Work

In this section, we provide an overview of the basic concepts in coalitional game theory. Let I, $|I| = n$, be a set of players. A subset $C \subseteq I$ is called a *coalition*. A *coalitional game with transferable utility* is defined by its *characteristic function* $v : 2^I \mapsto \mathbb{R}$ that specifies the *value* $v(C)$ of each coalition C [14]. Intuitively, $v(C)$ represents the maximal payoff the members of C can jointly receive by cooperating, and it is assumed that the agents can distribute this payoff between themselves in any way.

While the characteristic function describes the payoffs available to coalitions, it does not prescribe a way of distributing these payoffs. We say that an *allocation* is a vector of payoffs $\boldsymbol{x} = (x_1, \ldots, x_n)$ assigning some payoff to each $i \in I$. We write $x(S)$ to denote

[1] School of Electronics and Computer Science, University of Southampton, UK; email: {ee, gc2, nrj}@ecs.soton.ac.uk

$\sum_{i \in S} x_i$. An allocation is *feasible* for the grand coalition if $x(I) \leq v(I)$. An *imputation* is a feasible allocation that is also *efficient*, i.e., $x(I) = v(I)$.

A *weighted voting game (WVG)* is a coalitional game G given by a set of agents $I = \{1, \ldots, n\}$, their *weights* $\mathbf{w} = \{w_1, \ldots, w_n\}$, $w_i \in \mathbb{R}^+$, and a *threshold* $T \in \mathbb{R}$; we write $G = (I; \mathbf{w}; T)$. We use $w(S)$ to denote $\sum_{i \in S} w_i$. For a coalition $S \subseteq I$, its value $v(S)$ is 1 if $w(S) \geq T$; otherwise, $v(S) = 0$. Without loss of generality, the value of the grand coalition I is 1 (i.e., $w(I) \geq T$).

One of the best-known solution concepts describing coalitional stability is the *core*[8].

Definition 1. *An allocation* $\boldsymbol{x}$ *is in the* core *of* G iff $x(I) = v(I)$ **and** *for any* $S \subseteq I$ *we have* $x(S) \geq v(S)$.

If an allocation $\boldsymbol{x}$ is in the core, then no subgroup of agents can guarantee all of its members a higher payoff than the one they receive in the grand coalition under $\boldsymbol{x}$. This definition of the core can therefore be used to characterize the stability of the grand coalition.

The setting where several coalitions can form at the same time can be modeled using *coalition structures*. Formally, a coalition structure (CS) is an exhaustive partition of the set of agents. $\mathcal{CS}(G)$ denotes the set of all coalition structures for G. Given a structure $CS = \{C_1, \ldots, C_k\}$, an allocation $\mathbf{x}$ is *feasible for* CS if $x(C_i) \leq v(C_i)$ for $i = 1, \ldots, k$ and *efficient for* CS if this holds with equality.

Games with coalition structures were introduced by Aumann and Dreze [2], and are obviously of interest from an AI/multiagent systems point of view, as illustrated in Section 1. Indeed, in this context dealing with coalition structures other than the grand coalition is of uttermost importance: simply put, there is a plethora of realistic application scenarios where the emergence of the grand coalition is either not guaranteed, is plainly impossible, or might be perceivably harmful (for instance, it usually makes little sense to allocate all available robots on a single task). In particular, in the context of WVGs, by forming several disjoint winning coalitions, the agents generate more payoff than in the grand coalition. Additional motivation from an economics perspective is given in [2], which contains a thorough and insightful discussion on why coalition structures arise.

Now, there exists a handful of approaches in the multiagent literature that do take coalition structures explicitly into account. Sandholm and Lesser [11] discuss the stability of coalition structures when examining the problem of allocating *computational resources* to coalitions. Apt and Radzik [1] also do not restrain themselves to problems where the outcome is the grand coalition only. Instead, they introduce various stability notions for abstract games whose outcomes can be coalition structures, and discuss simple transformations by which stable partitions of the set of players may emerge. Dieckmann and Schwalbe [5] also propose a version of the core with coalition structures when studying dynamic coalition formation, and so do Chalkiadakis and Boutilier when tackling coalition formation under uncertainty [4]. None of these papers studies WVGs, however.

A thorough discussion of weighted voting games can be found in [13]. The stability-related solution concepts for WVGs (*without* coalition structures) have recently been studied by Elkind et al. [6], who also investigate them from computational perspective. However, there is no existing work in the literature studying WVGs with coalition structure—a class of games that we now proceed to define.

3 Coalition structures in WVGs

We now extend the traditional model for WVGs to allow for coalition structures. First, an *outcome* of a game is now a pair of the form (coalition structure, allocation) rather than just an allocation. Furtheremore, in the traditional model, any allocation of payoffs among the participating agents is required to be an exhaustive partition of the value of the grand coalition. In other words, it is always an imputation, i.e., an allocation of payoffs that is feasible and efficient for the grand coalition I. As we now allow WVGs to admit coalition structures, we replace the aforementioned requirement with similar requirements with respect to a coalition structure:

First, we no longer require an allocation to be an imputation in the classic sense. Instead, we demand that, for a given outcome $(CS, \boldsymbol{x})$, the allocation $\boldsymbol{x}$ of payoffs for I is feasible for CS. In this way, CS may contain *zero or more* winning coalitions. Furthermore, we define an *imputation for a coalition structure* CS as a vector $\boldsymbol{p}$ of non-negative numbers $(p_1, \ldots, p_n)$ (one for each agent in I), such that for *every* $C \in CS$ it holds $p(C) = v(C) \leq 1$; we write $\mathbf{p} \in \mathcal{I}(CS)$. That is, an imputation is now a feasible and efficient allocation of the payoff of any coalition $C \in CS$.

4 Core and CS-core of weighted voting games

In this section we define the core of WVG games with coalition structures, relate it to the "classic" core of WVG games without coalition structures, and obtain some core characterization results for a few interesting classes of WVG games.

The definition of the core (Def. 1) takes the following simple form in the traditional WVGs setting (see, e.g., [6]):

Definition 2. *The core of a WVG game* $G = (I; \mathbf{w}; T)$ *is the set of imputations* $\boldsymbol{p}$ *such that,* $\forall S \subseteq I$, $w(S) \geq T \Rightarrow p(S) \geq 1$.

Intuitively, an imputation $\boldsymbol{p}$ is in the core whenever the payoffs defined by $\boldsymbol{p}$ are such that any winning coalition already receives collective payoff of 1 (and therefore no coalition can improve its payoff by breaking away from the grand coalition).

This notion of the core cannot be directly used for coalition structures: indeed, it demands that an allocation is an imputation in the traditional sense, and therefore no imputation for a coalition structure with more than one winning coalition can ever be in the core. We will now extend this definition to the setting with coalition structures. Namely, we define the *core of weighted voting games with coalition structures*, or *CS-core*, as follows:

Definition 3. *The CS-core of a WVG game* $G = (I; \mathbf{w}; T)$ *with coalition structures is the set of outcomes* $(CS, \boldsymbol{p})$ *such that* $\forall S \subseteq I$, $w(S) \geq T \Rightarrow p(S) \geq 1$ **and** $\forall C \in CS$ *it holds* $p(C) = v(C)$.

Intuitively, given an outcome that is in the CS-core, no coalition has an incentive to break away from the coalition structure.

Now, it is well-known (see, e.g., [6]) that in weighted voting games the core is non-empty if and only if there exists a *veto* player, i.e., a player that belongs to all winning coalitions, and an imputation is in the core if and only if it distributes the payoff in some way between the veto players. This directly implies the following result.

Observation 1 (An imputation in the core induces an outcome in the CS-core)**.** *Let* $G = (I; \mathbf{w}; T)$*. If the core of* G *is non-empty, then, for any* $\boldsymbol{p}$ *in the core, the outcome* $(\{I\}, \boldsymbol{p})$ *is in the CS-core of G.*

However, it turns out that the CS-core may be non-empty even when the core is empty.

Example 1. *Consider a weighted voting game* $G = (I; \mathbf{w}; T)$*, where* $I = \{1, 2, 3\}$, $\mathbf{w} = (1, 1, 2)$ *and* $T = 2$*. It is easy to see that none of the players in* G *is a veto player, so* G *has an empty core. On* *the other hand, the outcome* $(CS, \mathbf{p})$*, where* $CS = \{\{1, 2\}, \{3\}\}$, $\mathbf{p} = (1/2, 1/2, 1)$ *is in the CS-core of* G*. Indeed, agent* 3 *is getting a payoff of* 1 *under this outcome, so his payoff cannot improve. Therefore, the only deviation available to the other two players is to form singleton coalitions, and this is clearly not beneficial.*

We now show that if the threshold T is strictly greater than 50% the CS-core and the core coincide.

Proposition 1 (In absolute majority games, the cores coincide)**.** *Let* $G = (I; \mathbf{w}; T)$ *be a WVG game with* $T > w(I)/2$*. Then there is an outcome* $(CS, \mathbf{p})$ *in the CS-core of* G *if and only if* $\mathbf{p}$ *is in the core of* G*. Consequently,* G *has a non-empty core if and only if it has a non-empty CS-core.*

Proof. Suppose that an outcome $(CS, \mathbf{p})$ is in the CS-core of G. As $T > w(I)/2$, CS can contain at most one winning coalition C, and hence $p(I) = 1$. Consider any player $i \in C$ such that $p_i > 0$. If p_i is not a veto player, we have $w(I \setminus \{i\}) \geq T$, $p(I \setminus \{i\}) < 1$, so $(CS, \mathbf{p})$ is not in the CS-core of G, a contradiction. Hence, under $\mathbf{p}$ only the veto players get any payoff, which implies that $\mathbf{p}$ is in the core of G. Conversely, if $\mathbf{p}$ is in the core of G, it is easy to see that $(\{I\}, \mathbf{p})$ is in the CS-core of G. □

We can also prove the following sufficient condition for non-emptiness of the CS-core.

Theorem 1. *Any WVG* $G = (I; \mathbf{w}; T)$ *that admits a partition of players into coalitions of weight* T *has a non-empty CS-core.*

Proof. Let $CS = \{C_1, \ldots, C_k\}$ be the corresponding partition such that $w(C_i) = T$ for all $i = 1, \ldots, k$. Define $\mathbf{p}$ by setting $p_j = w_j/T$ for all $j = 1, \ldots, n$. Consider any winning coalition S. We have $w(S) \geq T$, so $p(S) = w(S)/T \geq 1$, and hence S does not want to deviate. As this holds for any S with $v(S) = 1$, the outcome $(CS, \mathbf{p})$ is in the CS-core of G. □

However, it is not the case that the CS-core of a weighted voting game is always non-empty. In particular, this follows from the fact that the CS-core coincides with the core in games with $T > w(I)/2$, and such games may have an empty core. We now show that the CS-core can be empty also if $T < w(I)/2$:

Example 2. *Consider a WVG* $G = (I; \mathbf{w}; T)$*, where* $I = \{1, 2, 3, 4, 5\}$, $\mathbf{w} = (1, 1, 1, 1, 1)$ *and* $T = 2$*. We now show that this game has empty CS-core. Indeed, consider any* $CS \in \mathcal{CS}(G)$ *and any* $\mathbf{p} \in \mathcal{I}(CS)$*. Clearly,* CS *can contain at most two winning coalitions, so* $p(I) \leq 2$*. Now, if there is a coalition* $C \in CS$, $|C| \geq 3$*, such that* $p_i > 0$ *for all* $i \in C$*, any two players* $i, j \in C$ *can deviate by forming a winning coalition and splitting the surplus* $p(C \setminus \{i, j\})$*. If all coalitions have size at most 2, then there is a player* i *that forms a singleton coalition (and hence* $p_i = 0$*). There also exists another player* j *such that* $p_j < 1$ *(otherwise* $p(I) \geq 4$*). But then* $S = \{i, j\}$ *satisfies* $w(S) \geq T$, $p(S) < 1$*, so it is a successful deviation.*

5 Non-emptiness of the CS-core: hardness results

In the rest of the paper, we deal with computational questions related to the notion of the CS-core. This topic is important since in practical applications agents have limited computational resources, and may not be able to find a stable outcome if this requires excessive computation. To provide a formal treatment of complexity issues in our setting, we assume that all weights and the threshold are integers given in binary. As any rational weights can be scaled up to integers, this can be done without loss of generality.

In the previous section, we explained how to verify whether the core is non-empty or whether a given outcome is in the core. It is not hard to see that this verification can be done in polynomial time: e.g., to check the non-emptiness of the core, we simply check if $w(I \setminus \{i\}) \geq T$ for all $i \in I$. In WVGs with coalition structures, the situation is very different. Namely, we will show that it is NP-hard to decide whether a given WVG has a non-empty CS-core. Moreover, even if we are given an imputation, it is coNP-complete to decide whether it is in the CS-core of a given WVG. We now state these computational problems more formally.

Name: NONEMPTYCSCORE.
Instance: Weighted voting game $G = (I; \mathbf{w}; T)$.
Question: Does G have a non-empty CS-core?

Name: INCSCORE.
Instance: Weighted voting game $G = (I; \mathbf{w}; T)$, a coalition structure $CS \in \mathcal{CS}(G)$ and an imputation $\mathbf{p} \in \mathcal{I}(CS)$.
Question: Is $(CS, \mathbf{p})$ in the CS-core of G?

Both of our reductions rely on the well-known NP-complete PARTITION problem. An input to this problem is a pair $(A; K)$, where A is a list of positive integers $A = \{a_1, \ldots, a_n\}$ such that $\sum_{i=1}^n a_i = 2K$. It is a "yes"-instance if there is a subset of indices J such that $\sum_{i \in J} a_i = K$ and a "no"-instance otherwise [7, p.223].

Theorem 2. *The problem* NONEMPTYCSCORE *is NP-hard.*

Proof. We will describe a polynomial-time procedure that maps a "yes"-instance of PARTITION to a "yes"-instance of NONEMPTYCSCORE and a "no"-instance of PARTITION to a "no"-instance of NONEMPTYCSCORE. Suppose that we are given an instance $(a_1, \ldots, a_n; K)$ of PARTITION. If there is an i such that $a_i > K$, then obviously it is a "no"-instance of PARTITION, so we map it to a fixed "no"-instance of NONEMPTYCSCORE, e.g., by setting $G = (\{1, 2, 3, 4, 5\}; (1, 1, 1, 1, 1); 2)$ as in Example 2. Otherwise, we construct a game $G = (I; \mathbf{w}; T)$ by setting $I = \{1, \ldots, n\}$, $w_i = a_i$ for $i = 1, \ldots, n$, $T = K$. Note that in this case we have $w(I \setminus \{i\}) \geq T$ for any i, so there are no veto players in G.

Suppose that we have started with a "yes"-instance of PARTITION, and let J be such that $\sum_{i \in J} a_i = K$. Consider the coalition structure $CS = \{J, I \setminus J\}$ and an imputation $\mathbf{p}$ given by $p_i = w_i/K$ for $i = 1, \ldots, n$. Note that $w(J) = w(I \setminus J) = K$, so $p(J) = p(I \setminus J) = 1$, i.e., $\mathbf{p}$ is a valid imputation. It is easy to see that $(CS, \mathbf{p})$ is in the CS-core of G. Indeed, for any winning coalition S we have $w(S) \geq K$, so $p(S) \geq 1$, i.e., the members of S would not want to deviate.

On the other hand, suppose that we have started with a "no"-instance of PARTITION. Consider any outcome $(CS, \mathbf{p})$ in the resulting game. Clearly, CS can contain at most one winning coalition: if there are two disjoint winning coalitions, each of them has weight K, i.e., it can be used as a "yes"-certificate for PARTITION. If CS contains no winning coalitions, then it is clearly unstable, as $w(I) \geq T$, $p(I) = 0$. Now, suppose that CS contains exactly one winning coalition S. In this case we have $p(S) = p(I) = 1$ and $p_i = 0$ for all $i \notin S$. We have $p_i > 0$ for some $i \in S$, so $p(I \setminus \{i\}) < 1$. Moreover, by construction, $w(I \setminus \{i\}) \geq T$. Hence, $I \setminus \{i\}$ can deviate, so $(CS, \mathbf{p})$ is not in the CS-core of G. □

Theorem 3. *The problem* INCSCORE *is coNP-complete.*

Proof. We will show that the complementary problem on checking that a given outcome is not in the core is NP-complete.

First, it is easy to see that this problem is in NP: we can guess a coalition S such that $w(S) \geq T$, but $p(S) < 1$; this coalition can successfully deviate from $(CS, \mathbf{p})$. Now, to show that this problem is NP-hard, we construct a reduction from PARTITION as follows. Given an instance $(a_1, \dots, a_n; K)$ of PARTITION, we set $I = \{1, \dots, n, n+1, n+2\}$ and $w_i = 2a_i$ for $i = 1, \dots, n$. Define also $I' = \{1, \dots, n\}$. The weights w_{n+1} and w_{n+2} and the quota T are determined as follows. We construct a coalition S by adding agents $1, 2, \dots$ to it one by one until the weight of S is at least $2K$. If the weight of S is exactly $2K$, this means that we have started with a "yes"-instance of PARTITION. In this case, we set $w_{n+1} = w_{n+2} = 0$, $T = 2K$, $CS = \{I\}$, and $p_i = w_i/T$ for all $i \in I$. It is easy to see that the outcome $(CS, \mathbf{p})$ is not stable: the agents in S can deviate and increase their total payoff from $1/2$ to 1. Hence, in this case we have mapped a "yes"-instance of PARTITION to a "no"-instance of INCSCORE.

Now, suppose that $w(S) > 2K$. As all weights are even, we have $w(S) = 2Q$ for some integer $Q > K$. Also, we have $w(I' \setminus S) = 4K - 2Q$. Set $T = 2Q$, and let $w_{n+1} = w_{n+2} = 2Q - 2K$. Now we have $w(I \setminus S) = 4K - 2Q + 4Q - 4K = 2Q$, i.e., both S and $I \setminus S$ are winning coalitions. Set $CS = \{S, I \setminus S\}$. Now, $\mathbf{p}$ is defined as follows: for all $i \in I'$ set $p_i = w_i/T$, set $p_{n+1} = w_{n+1}/(T+1)$, and set $p_{n+2} = 1 - p(I' \setminus S) - p_{n+1}$. We have $p(S) = w(S)/T = 1$, $p(I \setminus S) = p(I' \setminus S) + p_{n+1} + p_{n+2} = 1$, so $\mathbf{p}$ is an imputation. Note also that we have $p_{n+1} + p_{n+2} = 1 - p(I' \setminus S) = 1 - w(I' \setminus S)/T = (w_{n+1} + w_{n+2})/T$. Moreover, we have $p_{n+1} < w_{n+1}/T$, $p(I' \setminus S) = w(I' \setminus S)/T$, and hence $p_{n+2} > w_{n+2}/T$.

We now show that if $(a_1, \dots, a_n; K)$ is a "yes"-instance of PARTITION, then $\langle (I; \mathbf{w}; T), CS, \mathbf{p} \rangle$ is a "no"-instance of INCSCORE. Indeed, suppose there is a set J such that $\sum_{i \in J} a_i = K$. Consider the coalition $J' = J \cup \{n+1\}$. We have $w(J') = 2K + 2Q - 2K$, so it is a winning coalition. On the other hand, $p(J') = p(J) + p_{n+1} = w(J)/T + w_{n+1}/(T+1) < w(J')/T = 1$. Hence, J' can benefit from deviating, i.e., $(CS, \mathbf{p})$ is not in the core.

On the other hand, suppose that $\langle (I; \mathbf{w}; T), CS, \mathbf{p} \rangle$ is a "no"-instance of INCSCORE, i.e., there is a set J'' such that $w(J'') \geq T$, $p(J'') < 1$. Suppose that $w(J'') > T$, i.e., $w(J'') \geq T + 1$. We have $p_i \geq w_i/(T+1)$ for all $i \in I$ (indeed, we have $p_i \geq w_i/T$ for $i \neq n+1$ and $p_i = w_i/(T+1)$ for $i = n+1$), so $p(J'') \geq w(J'')/(T+1) \geq 1$, a contradiction. Hence, we have $w(J'') = T$. Moreover, if $n+1 \notin J''$, we have $p(J'') \geq w(J'')/T = 1$, a contradiction again. Therefore, $n+1 \in J''$. Finally, if $n+2 \in J''$, we have $p(J'') = p(J'' \cap I') + p_{n+1} + p_{n+2} = w(J'' \cap I')/T + (w_{n+1} + w_{n+2})/T = w(J'')/T = 1$, also a contradiction. We conclude that $w(J'') = T$, $n+1 \in J''$, $n+2 \notin J''$, and hence $w(J'' \cap I') = 2Q - (2K - 2Q) = 2K$, which means that $\sum_{i \in J'' \cap I'} a_i = K$, i.e., $J'' \cap I'$ is a witness that we have a "yes"-instance of PARTITION. □

6 Algorithms for the CS-core

The hardness results presented in the previous section rely on all weights being given in binary. However, in practical applications it is often the case that the weights are not too large, or can be rounded down so that the weights of all agents are drawn from a small range of values. In such cases, we can assume that the weights are given in unary, or, alternatively, are at most polynomial in n. It is therefore natural to ask if our problems can be solved efficiently in such settings. It turns out that for INCSCORE this is indeed the case.

Theorem 4. *There exists a pseudopolynomial[2] algorithm $\mathcal{A}_{\text{InCsCore}}$ for* INCSCORE*, i.e., an algorithm that correctly decides whether a given outcome $(CS, \mathbf{p})$ is in the CS-core of a weighted voting game $(I; \mathbf{w}; T)$ and runs in time $\text{poly}(n, w(I), |\mathbf{p}|)$, where $|\mathbf{p}|$ is the number of bits in the binary representation of $\mathbf{p}$.*

Proof. The input to our algorithm is an instance of INCSCORE, i.e., a weighted voting game $G = (I; \mathbf{w}; T)$, a coalition structure $CS \in \mathcal{CS}(G)$ and an imputation $\mathbf{p} \in \mathcal{I}(CS)$. The outcome $(CS, \mathbf{p})$ is not stable if and only if there exists a set S such that $w(S) \geq T$, but $p(S) < 1$. This means that our problem is essentially reducible to the classic KNAPSACK problem [7], which is known to have a pseudopolynomial time algorithm based on dynamic programming. In what follows, we present this algorithm for completeness.

Let $W = w(I)$. For $j = 1, \dots, n$ and $w = 1, \dots, W$, let $P(j, w)$ be the smallest total payoff of a coalition with total weight w all of whose members appear in $\{1, \dots j\}$: $P(j, w) = \min\{p(J) \mid J \subseteq \{1, \dots, j\}, w(J) = w\}$. Now, if $\min_{w=T,\dots,W} P(n, w) < 1$, it means that there is a winning coalition whose total payoff is less than 1. Obviously, this coalition would like to deviate from $(CS, \mathbf{p})$, i.e., in this case $(CS, \mathbf{p})$ is not in the CS-core. Otherwise, the payoff to any winning coalition (not necessarily in CS) is at least 1, so no group wants to deviate from CS, and thus $(CS, \mathbf{p})$ is in the CS-core.

It remains to show how to compute $P(j, w)$ for all $j = 1, \dots, n$, $w = 1, \dots, W$. For $j = 1$, we have $P(1, w) = p_1$ if $w = w_1$ and $P(1, w) = +\infty$ otherwise. Now, suppose we have computed $P(j, w)$ for all $w = 1, \dots, W$. Then we can compute $P(j+1, w)$ as $\min\{P(j, w), p_{j+1} + P(j, w - w_j)\}$. The running time of this algorithm is polynomial in n, W and $|\mathbf{p}|$, i.e., in the input size. □

We now show how to use the algorithm $\mathcal{A}_{\text{InCsCore}}$ to check whether for a given coalition structure CS there *exists* an imputation $\mathbf{p}$ such that the outcome $(CS, \mathbf{p})$ is in the CS-core. Our algorithm for this problem also runs in pseudopolynomial time.

Theorem 5. *There exists a pseudopolynomial algorithm $\mathcal{A}_p$ that given a weighted voting game $G = (I; \mathbf{w}; T)$ and a coalition structure $CS \in \mathcal{CS}(G)$, correctly decides whether there exists an imputation $\mathbf{p} \in \mathcal{I}(CS)$ such that the outcome $(CS, \mathbf{p})$ is in the CS-core of G and runs in time $\text{poly}(n, w(I))$.*

Proof. Suppose $CS = \{C_1, \dots, C_k\}$. Consider the following linear feasibility program (LFP) with variables $p_1, \dots, p_n$:

$$\begin{aligned} p_i &\geq 0 && \text{for all } i = 1, \dots, n \\ \sum_{i \in C_j} p_i &= 1 && \text{for all } j \text{ such that } w(C_j) \geq T \\ \sum_{i \in C_j} p_i &= 0 && \text{for all } j \text{ such that } w(C_j) < T \\ \sum_{i \in J} p_i &\geq 1 && \text{for all } J \subseteq I \text{ such that } w(J) \geq T \end{aligned} \quad (1)$$

The first three groups of equations require that $\mathbf{p}$ is an imputation for CS: all payments are non-negative, the sum of payments to members of each winning coalition in CS is 1, and the sum of payments to members of each losing coalition in CS is 0. The last group of equations states that there is no profitable deviation: the payoff to each winning coalition (not necessarily in CS) is at least 1. Clearly, we can implement the algorithm $\mathcal{A}_p$ by solving this LFP, as follows:

The size of this LFP may be exponential in n, as there is a constraint for each winning coalition. Nevertheless, it is well-known that such LFPs can be solved in polynomial time by the ellipsoid method

[2] An algorithm whose running time is polynomial if all numbers in the input are given in unary is called *pseudopolynomial*.

provided that they have a polynomial-time *separation oracle*. A separation oracle is an algorithm that, given an alleged feasible solution, checks whether it is indeed feasible, and if not, outputs a violated constraint [12]. In our case, such an oracle will have to verify whether a given vector $\mathbf{p}$ violates one of the constraints in (1):

It is straightforward to verify whether all p_i are non-negative, and whether the payment to each winning coalition in CS is 1 and the payment to each losing coalition in CS is 0. If any of these constraints is violated, our separation oracle outputs the violated constraint. If this is not the case, we can use the algorithm $\mathcal{A}_{\mathrm{InCsCore}}$ described in the proof of Theorem 4 to decide whether there exists a winning coalition J such that $w(J) \geq T$, $p(J) < 1$; this algorithm can be easily adapted to return such coalition if one exists. If $\mathcal{A}_{\mathrm{InCsCore}}$ produces such a coalition, our separation oracle outputs the corresponding violated constraint. If $\mathcal{A}_{\mathrm{InCsCore}}$ reports that no such coalition exists, then $(CS, \mathbf{p})$ is in the CS-core of G, so we can output $\mathbf{p}$ and stop. □

The algorithm $\mathcal{A}_p$ described in the proof of Theorem 5 allows us to check whether a given weighted voting game G has a non-empty CS-core: we can enumerate all coalitional structures in $\mathcal{CS}(G)$, and for each of them check whether there is an imputation $\mathbf{p}$, which, combined with the coalition structure under consideration, results in a stable outcome. However, the number of coalition structures in $\mathcal{CS}(G)$ is exponential in n, and solving a linear feasibility problem for each of them using the ellipsoid method is prohibitively expensive. We now describe heuristics that can be used to speed up this process.

First, observe that we can exclude from consideration coalition structures that contain more than one losing coalition. Indeed, if any such coalition structure is stable, the coalition structure obtained from it by merging all losing coalitions will also be stable. Moreover, we can assume that each winning coalition C in our coalition structure is *minimal*, i.e., if we delete any element from C, it becomes a losing coalition. The argument is similar to the previous case: if any coalition structure with a non-minimal coalition C is stable, the coalition structure obtained by moving the extraneous element from C to the (unique) losing coalition is also stable.

Now, suppose that we have a coalition structure $CS = \{C_0, C_1, \dots, C_k\}$ such that $v(C_0) = 0$ (C_0 can be empty), $v(C_i) = 1$ for $i = 1, \dots, k$, and all C_i, $i > 0$, are minimal. Consider an agent $j \in C_i$, $i > 0$. If $p_j > 0$ and $w(C_0) \geq w_j$, then CS is not stable: the players in $C_0 \cup C_i \setminus \{j\}$ can deviate by forming a winning coalition and redistributing the extra payoff of p_j between themselves. Set $C'_i = \{j \in C_i \mid w_j \leq w(C_0)\}$. The argument above shows that the members of the sets C'_i get paid 0 under any imputation $\mathbf{p}$ such that $(CS, \mathbf{p})$ is stable. Now, set $C' = \cup_{i>0} C'_i$. If $w(C') + w(C_0) \geq T$, there is no imputation $\mathbf{p}$ such that $(CS, \mathbf{p})$ is stable: any such imputation would have to pay 0 to players in C_0 and each C'_i, but then the players in these sets can jointly deviate and form a winning coalition.

Therefore, we can speed up the algorithm in the proof of Theorem 5 as follows: given a coalition structure $CS = \{C_0, C_1, \dots, C_k\}$, compute the sets C'_i, $i = 1, \dots, k$, and check whether $w(C') + w(C_0) \geq T$. If this is indeed the case, there is no imputation $\mathbf{p}$ such that $(CS, \mathbf{p})$ is stable. Otherwise, run the algorithm $\mathcal{A}_p$. Clearly, this preprocessing step is very fast (in particular, unlike $\mathcal{A}_p$, it runs in polynomial time even if the weights are large, i.e., given in binary), and in many cases we will be able to reject a candidate coalition structure without having to solve the LFP (which is computationally expensive).

We can also try to optimize the order in which we consider the candidate coalition structures. Heuristics for social welfare-maximizing coalition structure generation might be of use here [10, 9].

7 Conclusions

In this paper, we extended the model of weighted voting games (WVGs) to allow for the formation of coalition structures, thus permitting more than one coalition to be *winning* at the same time. We then studied the problem of stability of the resulting structure in such games. Specifically, we introduced *CS-core* (the core with coalition structures), and discussed its properties by relating it to the traditional concept of the core for WVGs and proving sufficient conditions for its non-emptiness. Following that, we showed that deciding CS-core non-emptiness or checking whether an outcome is in the CS-core are computationally hard problems (unlike what holds in the traditional WVGs setting). However, for specific classes of games, we presented polynomial-time algorithms for checking if a given outcome is in the CS-core, and discovering a CS-core element given a coalition structure. We then suggested heuristics that, combined with these algorithms, can be used to generate an outcome in the CS-core. We believe that the line of work presented here is important: Weighted voting games are well understood, and the addition of coalition structures increases the usability of this intuitive framework in multiagent settings (where weights can represent resources and thresholds do not necessarily exceed 50%).

In terms of future work, we intend, first of all, to come up with new heuristics to speed up our algorithms. In addition, notice that the algorithms and heuristics of Sec. 6 provide essentially centralized solutions to their respective problems. Therefore, we are interested in studying *decentralized* approaches; to begin, we intend to speed up, in the WVGs context, the exponential decentralized coalition formation algorithm of [5]. Finally, studying other solution concepts in this context, such as the Shapley value [8], is also within our intentions.

Acknowledgements This research was undertaken as part of the ALADDIN (Autonomous Learning Agents for Decentralised Data and Information Networks) project. ALADDIN is jointly funded by a BAE Systems and EPSRC strategic partnership (EP/C548051/1).

REFERENCES

[1] K. Apt and T. Radzik. Stable Partitions in coalitional games, 2006. Working Paper, available at http://arxiv.org/abs/cs.GT/0605132.

[2] R.J. Aumann and J.H. Dreze, 'Cooperative Games with Coalition Structures', *International Journal of Game Theory*, **3**(4), 217–237, (1974).

[3] P. Caillou, S. Aknine, and S. Pinson, 'Multi-agent models for searching pareto optimal solutions to the problem of forming and dynamic restructuring of coalitions', in *Proc. of ECAI'02*, pp. 13 – 17, (2002).

[4] G. Chalkiadakis and C. Boutilier, 'Bayesian Reinforcement Learning for Coalition Formation Under Uncertainty', in *Proc. of AAMAS'04*.

[5] T. Dieckmann and U. Schwalbe. Dynamic Coalition Formation and the Core, 1998. Economics Dept. Working Paper Series, Nat. Univ. of Ireland - Maynooth.

[6] E. Elkind, L.A. Goldberg, P.W. Goldberg, and M. Wooldridge, 'Computational complexity of weighted threshold games', in *Proc. of AAAI'07*.

[7] M. Garey and D. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*, W. H. Freeman & Co., N. York, 1990.

[8] R. Myerson, *Game Theory: Analysis of Conflict*, 1991.

[9] T. Rahwan, S. Ramchurn, A. Giovannucci, V. Dang, and N. R. Jennings, 'Anytime optimal coalition structure generation', in *Proc. of AAAI'07*.

[10] T. Sandholm, K. Larson, M. Andersson, O. Shehory, and F. Tohme, 'Anytime coalition structure generation with worst case guarantees', in *Proc. of AAAI'98*, (1998).

[11] T. Sandholm and V.R. Lesser, 'Coalitions Among Computationally Bounded Agents', *Artificial Intelligence*, **94**(1), 99 – 137, (1997).

[12] A. Schrijver, *Combinatorial Optimization: Polyhedra and Efficiency*, Springer, 2003.

[13] A. Taylor and W. Zwicker, *Simple Games: Desirability Relations, Trading, Pseudoweightings*, Princeton University Press, Princeton, 1999.

[14] J. von Neumann and O. Morgenstern, *Theory of Games and Economic Behavior*, Princeton University Press, Princeton, 1944.