Artificial Intelligence 271 (2019) 74–97

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

Cooperative games with overlapping coalitions: Charting the tractability frontier

Yair Zick [a,*], Georgios Chalkiadakis [b], Edith Elkind [c], Evangelos Markakis [d]

[a] School of Computing, National University of Singapore, Singapore
[b] School of Electronic and Computer Engineering, Technical University of Crete, Greece
[c] Department of Computer Science, University of Oxford, UK
[d] Department of Informatics, Athens University of Economics and Business, Greece

ARTICLE INFO

Article history:
Received 6 July 2016
Received in revised form 24 January 2018
Accepted 21 November 2018
Available online 30 January 2019

Keywords:
Cooperative games
Overlapping coalition formation
Core
Treewidth
Arbitration functions

ABSTRACT

The framework of cooperative games with overlapping coalitions (OCF games), which was proposed by Chalkiadakis et al. [1], generalizes classic cooperative games to settings where agents may belong to more than one coalition. OCF games can be used to model scenarios where agents distribute resources, such as time or energy, among several tasks, and then divide the payoffs generated by these tasks in a fair and/or stable manner. As the framework of OCF games is very expressive, identifying settings that admit efficient algorithms for computing 'good' outcomes of OCF games is a challenging task. In this work, we put forward two approaches that lead to tractability results for OCF games. First, we propose a discretized model of overlapping coalition formation, where each agent i has a weight $W^i \in \mathbb{N}$ and may allocate an integer amount of weight to any task. Within this framework, we focus on the computation of outcomes that are socially optimal and/or stable. We discover that the algorithmic complexity of this task crucially depends on the amount of resources that each agent possesses, the maximum coalition size, and the pattern of communication among the agents. We identify several constraints that lead to tractable subclasses of discrete OCF games, and supplement our tractability results by hardness proofs, which clarify the role of our constraints. Second, we introduce and analyze a natural class of (continuous) OCF games—the *Linear Bottleneck Games*. We show that such games always admit a stable outcome, even assuming a large space of feasible deviations, and provide an efficient algorithm for computing such outcomes.

1. Introduction

Consider the following simple market exchange. Two sellers (Alice and Bob) each own a ton of coffee beans (a divisible good), which they would like to sell to two potential buyers (Claire and Dave). Claire and Dave operate in different markets, so Alice and Bob can justifiably offer them different prices. Suppose that having agreed on a transaction schedule and payments, Alice decides that she is unhappy with her revenue from the deal with Claire; she wants to cancel the deal. However, Dave, upon hearing that Alice reneged on her deal with Claire, no longer wishes to work with Alice, canceling

* Corresponding author.
E-mail addresses: zick@comp.nus.edu.sg (Y. Zick), gehalk@ece.tuc.gr (G. Chalkiadakis), elkind@cs.ox.ac.uk (E. Elkind), markakis@gmail.com (E. Markakis).

https://doi.org/10.1016/j.artint.2018.11.006
0004-3702/

his agreement with her as well. Dave's motivation for doing so may stem from many reasons, e.g. a potential business partnership with Claire, or market 'best practices'.

This setting features several interesting characteristics. First, an agent may sell resources to several agents, and may also buy from several other agents. In other words, agents may allocate resources to several profit-generating tasks. Second, agents may withdraw some of their resources from some agreements. For example, Alice may wish to sell less coffee to some customer, but not change her interactions with other parties. Finally, when trying to strategically change an agreement, agents must be aware of how their actions affect the contracts they still maintain with other (possibly unaffected) parties.

In our example, agents must collaborate in order to generate revenue. Having generated revenue by making an exchange, agents are free to share the profits from the exchange as they see fit. Profit sharing can be done directly, if the agents jointly produce a new good using their resources, sell it, and distribute the revenue among themselves. It can also be indirect: a seller shares the profit from her transaction with a buyer by setting a price for her good. In either case, agents must identify a way to generate profits, and, subsequently, share profits among themselves in a reasonable manner. When sharing profits, agents should account for individuals or groups of agents who feel that they are underpaid. Indeed, a group of agents that can get more money by deviating from the proposed agreement may destabilize the entire market, resulting in a cascade of deviations, which may eventually produce a less desirable state (not to mention the cost of actual deviation). However, what constitutes a profitable deviation strongly depends on how non-deviators respond to the deviators' actions.

Reasoning about this system of incentives and reactions is a significant challenge. While many group interaction scenarios can be represented by the framework of transferable utility cooperative games (TU games), with stable profit-sharing schemes captured by the notion of the core (see, e.g., Peleg and Sudhölter [2]), the standard setting of TU games is not expressive enough to deal with agents participating in several collaborative agreements simultaneously. A few years ago, Chalkiadakis et al. [1] proposed a novel approach to modeling scenarios where agents can divide resources among several joint tasks, by introducing the framework of *overlapping coalition formation games* (OCF games) and defining several variants of the notion of core stability for this model, which capture different reactions to deviations by non-deviating agents; these include the conservative core, the refined core, the optimistic core, and the sensitive core. Following their work, Zick et al. [3] proposed the notion of an arbitration function, which provides a general model for handling deviations in OCF games. These two works offer a comprehensive conceptual model for analyzing stability in settings where agents work on several concurrent projects and may deviate in a complex manner.

In contrast, there has been a very limited amount of work on computing solution concepts for OCF games. The theory of OCF games is a generalization of classic cooperative game theory [2], where computational issues are relatively well-understood (see [4] for an overview). However, as Chalkiadakis et al. [1] show, more elaborate reactions to deviation may increase computational complexity: even if a solution concept is easy to compute for classic cooperative games, computing its OCF analogue may be NP-hard. For example, Chalkiadakis et al. [1] study a class of games called *threshold task games*: these are games where each agent is associated with an integer weight, and the value of a coalition is a piece-wise constant function of its total weight. They show that a payoff division in the core of a threshold task game can be found in pseudopolynomial time (i.e. in time polynomial in the number of agents and in the largest agent weight), if one assumes that when a set of agents deviates, no other agent will want to work with agents in this set—a reaction termed *conservative*, which closely approximates the setting of TU games (as explained by Zick et al. [3]). In contrast, if non-deviators agree to work with the members of the deviating set in coalitions that are unharmed by the deviation (a reaction termed *refined*), the same problem becomes NP-hard.

It follows that, when assessing the computational complexity of finding stable outcomes in OCF games, one needs to consider not only structural properties of the characteristic function (i.e. the way agents generate profits), but also the way agents react to deviations. In our work, we study the influence of these two aspects of OCF games on the complexity of answering stability-related questions in OCF games.

1.1. Our contribution

The computational complexity of any given problem depends on how the input is represented. For instance, a general TU game with n agents is represented by 2^n real values: we need one number for each subset (coalition) of players. Thus, no algorithm that needs to know the value of each coalition can run in time that is polynomial in the number of agents. This issue is even more prominent for games with overlapping coalitions: to describe an OCF game, we need to specify a number for every *partial coalition*, i.e., agreement of the form 'agent 1 contributes an x_1 fraction of her resources, ..., agent n contributes an x_n fraction of her resources', for every possible combination of (real) values $x_1, \ldots, x_n \in [0, 1]$. Thus, a general OCF game cannot be described by a finite list of numbers, and hence one cannot meaningfully reason about the complexity of general OCF games. To mitigate this issue, we explore two different strategies:

1. We introduce and study *discrete OCF games*, which place restrictions on how finely an agent can split her resources (Sections 3 to 6). In these games, each agent i is associated with an integer weight $W^i \in \mathbb{N}$ and can only allocate an integer weight between 0 and W^i to each task. By definition, a discrete OCF game can be represented as a finite list of numbers, and thus discrete OCF games are amenable to traditional complexity-theoretic analysis.
2. We identify a large class of (standard, i.e., non-discrete) OCF games that can be succinctly represented (Section 7). Specifically, we study a rich and expressive class of games that we term *linear bottleneck games* (LBGs). A game in this

class can be described by a list of tasks; the payoff from each task depends on the inherent value of each task and the players' contributions to it. These games capture interesting combinatorial optimization scenarios where overlapping coalitions can naturally arise.

For both of these classes of OCF games, we investigate the complexity of several closely related challenges:

1. Given an OCF game, find an optimal coalition structure—i.e. an optimal way for agents to divide into groups and generate profits.
2. Given an OCF game, an outcome of this game (i.e. a coalition structure together with a payoff division) and a subset of agents, compute the most that this subset can get by deviating (given how non-deviators are expected to react to deviations).
3. Given an OCF game and an outcome of this game, decide whether this outcome is stable, i.e., whether there exists a subset of agents that can benefit from deviating (for a given non-deviators' reaction).
4. Given an OCF game, find a stable outcome if one exists (for a given non-deviators' reaction).

In the first part of the paper, we focus on discrete OCF games. We make no assumptions on how agents generate revenue (i.e. the characteristic function can take any form), and focus instead on the *structure* of their interaction. In Section 3, we show that, for tractability, we need to bound the size of each coalition, and, moreover, impose a constraint on the overall agent interaction pattern. Specifically, inspired by the work of Myerson [5] and Demange [6], we study settings where agents are connected in a social network (mathematically, an undirected graph), and feasible coalitions correspond to connected subgraphs of this graph. We show that when this social network is acyclic and the maximum coalition size is 2, it becomes possible to find an optimal coalition structure in polynomial time. However, these constraints are insufficient for tractability of stability-related problems: in Section 4 we show that, for these problems to admit efficient algorithms, the non-deviators' reaction must also be limited in scope, i.e. complex reactions to deviation are a source of computational complexity in and of themselves.

We show that, when all of our conditions on agent interaction and the non-deviators' reaction are satisfied, the stability-related problems we consider admit efficient algorithms. We also show that none of these conditions can be dropped, by providing NP-hardness proofs for settings where they are not satisfied. Our results extend to the case where the social network formed by the agents has bounded treewidth (Section 6); our algorithms for this case run in time that is polynomial n and $(W_{\max})^k$, where n is the number of agents, $W_{\max}$ is the maximum agent weight, and k is the treewidth of the social network.

In the second part of our paper, we consider linear bottleneck games. In these games, there is a set of agents N and a list of tasks $(T_1, \ldots, T_m)$. Each task T_j is associated with a set of agents $A_j \subseteq N$ who are needed to complete it, as well as a *value* $\pi_j \in \mathbb{R}_+$. Each agent $i \in N$ has a weight $\omega^i \in \mathbb{Q}$, which she can freely distribute among the tasks. If each agent $i \in A_j$ contributes x^i units of weight to T_j, then the payoff that the agents in A_j earn from T_j is $\pi_j \cdot \min_{i \in A_j} x^i$. That is, the payoff from the task is determined by the *smallest* contribution to it, i.e., the bottleneck.

LBGs capture many interesting game-theoretic settings, including, for instance, *multicommodity flow games* [7,8]. Briefly, in multicommodity flow games pairs of vertices in a network want to send and receive flow, which has to be transmitted by edges of the network. This setting can be modeled by a linear bottleneck game, where both vertices and edges are players. The weight of an edge player is the capacity of his edge, while the weight of a vertex player is the amount of commodity she would like to send or receive. Further examples of settings that can be represented in the framework of LBGs are described in Section 7.

It is known that multicommodity flow games admit outcomes that are stable in the sense of classic cooperative game theory [8]. We generalize and strengthen this result, by showing that linear bottleneck games admit stable outcomes even if we assume a very lenient reaction to deviation (in general, as argued by Chalkiadakis et al. [1] and Zick et al. [3], the more lenient non-deviators are, the smaller is the space of stable outcomes). Moreover, we provide a linear programming-based algorithm that finds stable outcomes of LBGs in polynomial time.

1.2. Related work

Our work expands and builds upon two previous papers on OCF games: the paper of Chalkiadakis et al. [1], which introduced the OCF model, and the more recently published paper by Zick et al. [3]. While the primary focus of Chalkiadakis et al. [1] is not algorithmic, they present some complexity results for threshold task games. For example, Chalkiadakis et al. show that it is possible to find an outcome in the core of a threshold task game in pseudopolynomial time, assuming that the reaction to deviation is conservative; that is, agents in a deviating set do not expect the non-deviators to collaborate with them. As Zick et al. [3] show, this conservative assumption makes stability concepts in OCF games 'collapse' to their classic (i.e., non-OCF) equivalents. However, if one assumes a more refined reaction to deviation—namely, that non-deviators will continue collaborating with deviators if they are not affected by the deviation—the problem of finding a core stable outcome for TTGs becomes NP-hard. This work is the first indication that different notions of stability (as captured by arbitration functions) not only lead to different outcomes, but may also have different computational properties. While Zick et al. [3] do not study computational aspects of OCF games, their work presents us with some useful tools for the

design of efficient algorithms for this type of games. In particular, Zick et al. [3] show that certain classes of OCF games are guaranteed to have a non-empty core; however, their proofs rely on balancedness conditions, and are not computational in nature. Thus, even though some games are guaranteed to be stable with respect to some arbitration functions, computing stable outcomes may be hard.

The related problem of computing an optimal coalition structure has received plenty of attention in the AI literature, starting with the influential papers of Sandholm et al. [9] and Larson and Sandholm [10]; we point the readers to the surveys of Elkind et al. [11] and Rahwan et al. [12]. Some authors have studied this problem for games with overlapping coalitions as well; we mention the important early paper by Shehory and Kraus [13], as well as the work by Lin and Hu [14] and Zhang et al. [15]. However, these papers operate under a different methodology and objectives. Specifically, they aim to find optimal or approximately optimal coalition structures, but do not consider their stability, whereas we are also interested in computing stable revenue division schemes. Furthermore, these works focus on specific problem domains, whereas our results hold for more general classes of games.

Computational issues in classic cooperative games have been an object of extensive study. Early computational results can be attributed to some of the field's 'founding fathers': Mann and Shapley [16,17] study methods to compute the Shapley value (an important solution concept in cooperative game theory) both exactly and approximately. Interestingly, their results, while not phrased in the language of modern computational complexity, are computational in nature. The important early paper of Deng and Papadimitriou [18] was the precursor of several works on the subject; a non-comprehensive list includes papers by Deng et al. [19], Ieong and Shoham [20], Matsui and Matsui [21], Elkind et al. [22], Greco et al. [23] (see the survey by Chalkiadakis et al. [4]). Stability in OCF environments has been recently considered by Zhang et al. [24], who employ the OCF model in order to analyze wireless networks, and show some stability results in this setting; however, their work again focuses on a specific problem formulation, rather than general classes of OCF games. Ackerman and Brânzei [25] study a pairwise collaboration model that is similar to the OCF model studied here; however, their paper studies pairwise equilibria in this model, rather than core stability.

Coalitional games where possible interactions among the agents are described by a network were first considered by Myerson [5]; in the literature such networks are called communication graphs. Demange [6] studies cooperative games whose communication graphs are trees; not only do these games have a non-empty core, but for superadditive games a core outcome can be found in polynomial time. Brafman et al. [26] exploit the underlying communication graph structure to derive polynomial-time algorithms for a class of cooperative games that are motivated by planning scenarios. However, the positive algorithmic results of Brafman et al. [26] and Demange [6] do not extend to communication networks of bounded treewidth: there exist classes of coalitional games whose underlying communication network has a bounded treewidth, but for which computing a core-stable outcome is computationally intractable [27]. Nevertheless, games on bounded-treewidth communication networks are more stable, in the sense that their cost of stability (the minimum subsidy to the grand coalition that ensures its stability, see the work of Bachrach et al. [28]) can be bounded in terms of the treewidth of the communication network [29,30]. Recently, Igarashi and Elkind [31] analyzed the complexity of computing stable outcomes in graph-restricted hedonic games, obtaining a number of easiness results for a variety of solution concepts for the case where the communication network is a tree; subsequently, similar results have been derived for another coalition formation task that is known as the group activity selection problem [32–34].

2. Preliminaries

In what follows, we use uppercase letters to refer to sets, and boldface letters to refer to vectors. Given two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, we write $\mathbf{x} \leq \mathbf{y}$ if $x^i \leq y^i$ for all $i \in \{1, \ldots, n\}$. Given a set of agents $S \subseteq \{1, \ldots, n\}$ and a vector $\mathbf{x} \in \mathbb{R}^n$, let $x(S) = \sum_{i \in S} x^i$ and let $\mathbf{x}^S$ be the vector in $\mathbb{R}^n$ whose i-th entry is x^i if $i \in S$ and 0 otherwise. Let $\mathbf{e}^S$ be the indicator vector of S in $\mathbb{R}^n$, i.e. the i-th entry of $\mathbf{e}^S$ is 1 if $i \in S$ and 0 otherwise.

We begin by recalling the definition of a classic cooperative game (see, e.g., [2]). A *cooperative game* $\mathcal{G}$ is a tuple $\langle N, u \rangle$, where $N = \{1, \ldots, n\}$ is a set of agents and $u : 2^N \to \mathbb{R}_+$ is a function that assigns a value to every subset of agents $S \subseteq N$. Subsets of N are also referred to as *coalitions*. A *coalition structure* is a partition of agents $\Pi = \{S_1, \ldots, S_m\}$ into disjoint subsets; that is, $\bigcup_{j=1}^{m} S_j = N$ and for all $S_j, S_k \in \Pi$ with $j \neq k$ it holds that $S_j \cap S_k = \emptyset$. An *imputation* for Π is a vector $\mathbf{p} = (p^1, \ldots, p^n)$ that satisfies the following two conditions: (1) $p^i \geq u(\{i\})$ for all $i \in N$ (*individual rationality*), and (2) $p(S_j) = u(S_j)$ for all $S_j \in \Pi$ (*coalitional efficiency*). In words, an imputation is a division of the payoffs generated by the agents forming Π such that agents are individually incentivized to form a coalition structure, and the payoff from forming a coalition S_j is allocated to the members of S_j only. An *outcome of* $\mathcal{G}$ is a pair $(\Pi, \mathbf{p})$, where Π is a coalition structure and $\mathbf{p}$ is an imputation for Π. Observe that in every coalition structure each agent participates in exactly one coalition. While this is a valid assumption in many multi-agent scenarios, it is often the case that agents split their resources among several projects, forming overlapping coalitions.

Overlapping coalition formation (OCF) games [1] lift the assumption that each agent participates in exactly one coalition. An OCF game is also given by a tuple $\mathcal{G} = \langle N, v \rangle$, where $N = \{1, \ldots, n\}$ is a set of agents and v is the characteristic function. However, in contrast with the classic case, the function v is defined on all vectors of the form $[0, 1]^n$, and assigns a non-negative real value to every *partial coalition* $\mathbf{c} = (c^1, \ldots, c^n) \in [0, 1]^n$. The quantity c^i is the contribution of agent i to $\mathbf{c}$, and indicates which fraction of her resources is allocated to $\mathbf{c}$. A *coalition structure* for $\mathcal{G}$ is a (possibly infinite) list of partial coalitions CS such that for each $i \in N$ it holds that $\sum_{\mathbf{c} \in CS} c^i \leq 1$.

Note that, in the definition above, to fully describe v, we need to specify its value at every real point in $[0,1]^n$, and a coalition structure may contain infinitely many coalitions. Both of these issues make it difficult to represent OCF games and to reason about them. To overcome these difficulties, we will now introduce a discretized version of OCF games and show how to adapt the existing concepts developed for OCF games to the discrete setting.

In what follows, we provide formal definitions of the notions used in this paper, focusing on the discrete model. Conveniently, many definitions for the continuous model apply without change to the discrete setting; when this is not the case, we provide the definition for the continuous case in brackets (these definitions will be used in the second half of the paper, where we present some algorithmic results for a special class of continuous OCF games).

A *discrete overlapping coalition formation (dOCF) game* is a tuple $\mathcal{G} = \langle N, \mathbf{W}, v\rangle$. Here, $N = \{1, \ldots, n\}$ is the set of agents; each agent $i \in N$ has a *weight* W^i, and the agents' weights are collectively described by the vector $\mathbf{W} = (W^1, \ldots, W^n)$. One can think of W^i as the amount of some resource that agent i possesses. Let $\mathcal{W} = \{\mathbf{c} \in \mathbb{Z}_+^n \mid \mathbf{c} \leq \mathbf{W}\}$, or, equivalently,

$$\mathcal{W} = \{0, \ldots, W^1\} \times \cdots \times \{0, \ldots, W^n\};$$

the set $\mathcal{W}$ is the set of all possible ways in which agents can contribute resources to a *single* task. A vector $\mathbf{c} \in \mathcal{W}$ is called a *partial coalition* (for succinctness, we will often omit the qualifier 'partial' when speaking of such coalitions); c^i is the *contribution of agent* i. If $c^i = 0$ then agent i contributes nothing to completing the associated task, and if $c^i = W^i$ then all of agent i's resources are assigned to this task. The *characteristic function* $v : \mathcal{W} \to \mathbb{R}_+$ receives as input a coalition $\mathbf{c} \in \mathcal{W}$ and outputs a value $v(\mathbf{c})$, describing the profit that this coalition can generate.

Given a partial coalition $\mathbf{c}$ in $\mathcal{W}$ (or, in the continuous setting, in $[0,1]^n$), we define the *support* of $\mathbf{c}$ to be the set of all agents in N that contribute some of their resources to $\mathbf{c}$: we write $supp(\mathbf{c}) = \{i \in N \mid c^i > 0\}$. Agents in $supp(\mathbf{c})$ are the only ones who may receive a share of the profits made by $\mathbf{c}$, and may therefore be affected by changes to $\mathbf{c}$.

A *coalition structure* for a discrete OCF game is a finite list of coalitions $CS = (\mathbf{c}_1, \ldots, \mathbf{c}_m)$; we write $|CS|$ to denote the number of coalitions in CS. Since no agent can contribute more than the total amount of resources she possesses, we require that $\sum_{\mathbf{c}\in CS} c^i \leq W^i$ for all $i \in N$ (this is the analogue of the condition $\sum_{\mathbf{c}\in CS} c^i \leq 1$ for the continuous case).

Example 2.1. Consider the following three-player game, where players form overlapping coalitions in order to complete a set of tasks. Players' weights are as follows: $W_1 = W_2 = 2$; $W_3 = 1$. There are four types of tasks:

- A task of type t_1 can be completed by player 1 alone, requires all of her resources, and is worth 5.
- A task of type t_{12} requires 50% of both player 2 and player 1's resources, and is worth 10.
- A task of type T_{12} requires all of the resources of players 1 and 2, and is worth 20.
- A task of type t_{23} requires all of player 3's resources and 50% of player 2's resources, and is worth 9.

Consider coalition structures $CS = (\mathbf{c}_1, \mathbf{c}_2)$ and $CS' = (\mathbf{c}'_1)$, where

$$\mathbf{c}_1 = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}, \mathbf{c}_2 = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}, \mathbf{c}'_1 = \begin{pmatrix} 2 \\ 2 \\ 0 \end{pmatrix}.$$

It is easy to see that both CS and CS' maximize the players' total payoff. Indeed, it is best for players 1 and 2 to work together and earn a total of 20 (by completing t_{12} twice or T_{12} once), while player 3 makes no profit.

Note that CS is a list rather than a set; this is because some coalitions may form more than once (this corresponds to agents completing several identical tasks). Nevertheless, we use standard set notation to refer to elements of CS. That is, we write $\mathbf{y} \in CS$ if $\mathbf{y}$ is one of the coalitions listed in CS, and we write $CS' \subseteq CS$ if CS' is obtained from CS by removing some of its elements. We overload notation and write $v(CS)$ to refer to $\sum_{\mathbf{c}\in CS} v(\mathbf{c})$. Given a subset of agents $S \subseteq N$, we write $\mathcal{CS}(S)$ to denote the set of all coalition structures that can be formed by members of S; that is, $\mathcal{CS}(S)$ consists of all coalition structures CS such that $supp(\mathbf{c}) \subseteq S$ for all $\mathbf{c} \in CS$. We set $\mathcal{CS} = \mathcal{CS}(N)$. The *weight* of a coalition structure CS is defined as $\mathbf{w}(CS) = \sum_{\mathbf{c}\in CS} \mathbf{c}$. We say that $CS \in \mathcal{CS}(S)$ is *efficient for* S if all agents in S contribute all of their resources to CS, that is, $\mathbf{w}(CS) = \mathbf{W}^S$ (in the continuous case, this condition becomes $\mathbf{w}(CS) = \mathbf{e}^S$).

Given a coalition structure $CS \in \mathcal{CS}$ and a set $S \subseteq N$, let $CS|_S$ be the coalition structure CS *reduced to* S, $CS|_S$, i.e. we set

$$CS|_S = (\mathbf{y} \in CS \mid supp(\mathbf{y}) \subseteq S)\,.$$

The coalition structure $CS|_S$ includes all coalitions in CS that are fully controlled by the members of S. If S decides to deviate, the resources used by $CS|_S$ are freely available to S. In contrast, any coalition $\mathbf{y} \in CS \setminus CS|_S$ has non-S members, and any changes to such coalitions may lead to negative repercussions for S.

Given a dOCF game $\mathcal{G} = \langle N, \mathbf{W}, v\rangle$ (respectively, an OCF game $\mathcal{G} = \langle N, v\rangle$), we define the *superadditive cover* of v to be the function $v^* : \mathcal{W} \to \mathbb{R}_+$ (respectively, $v^* : [0,1]^n \to \mathbb{R}_+$) such that for every coalition $\mathbf{c}$

$$v^*(\mathbf{c}) = \sup\{v(CS) \mid \mathbf{w}(CS) = \mathbf{c}\}.$$

Simply put, $v^*(\mathbf{c})$ is the maximum profit that agents can generate by forming coalitions when their total resources are given by $\mathbf{c}$. We note that in the discrete model, for any $\mathbf{c} \in \mathcal{W}$ there exists a coalition structure $CS \in \mathcal{CS}$ such that $\mathbf{w}(CS) = \mathbf{c}$ and $v^*(\mathbf{c}) = v(CS)$, i.e. in the expression above 'sup' can be replaced with 'max'; however, in the continuous setting this is not the case.

Observe that if $W^i = 1$ for all $i \in N$, the resulting game is a classic cooperative game with coalition structures [35].

2.1. Payoff division

Having formed a coalition structure, agents have to divide the profits generated by their joint work. Given a coalition structure $CS = (\mathbf{c}_1, \ldots, \mathbf{c}_m)$, an *imputation* $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_m)$ for CS is a list of $|CS| = m$ vectors in $\mathbb{R}^n_+$ (in the continuous case, where the number of coalitions in CS may be infinite, we require that there is a bijection between CS and $\mathbf{x}$). The vector $\mathbf{x}_j$ describes how the profits from the coalition $\mathbf{c}_j$ are divided among the agents. Given a coalition $\mathbf{c} \in CS$ and an imputation $\mathbf{x}$ for CS, we denote the division of profits from $\mathbf{c}$ by $\mathbf{x}(\mathbf{c})$; the payoff to agent i from $\mathbf{c}$ is $x^i(\mathbf{c})$. We require a division of profits to satisfy the following rules:

Coalitional efficiency: for all $\mathbf{c} \in CS$, the total payoff from $\mathbf{x}(\mathbf{c})$ must equal $v(\mathbf{c})$; that is, $\sum_{i \in N} x^i(\mathbf{c}) = v(\mathbf{c})$.

No side payments: for all $\mathbf{c} \in CS$, if $i \notin supp(\mathbf{c})$ then $x^i(\mathbf{c}) = 0$. This condition simply means that agents who did not contribute any resources to the coalition $\mathbf{c}$ may not partake in the profits generated by $\mathbf{c}$.

Individual rationality: for all $i \in N$, it holds that $\sum_{\mathbf{c} \in CS} x^i(\mathbf{c}) \geq v^*(\mathbf{W}^i)$, i.e. the total payoff to i is at least the amount that i can get by working alone (for the continuous case, the right-hand side of this inequality becomes $v^*(\mathbf{e}^i)$).

The set of all possible imputations for a coalition structure CS is denoted $I(CS)$; an *outcome* is a pair $(CS, \mathbf{x})$, where CS is a coalition structure and $\mathbf{x}$ is an imputation in $I(CS)$. We also define $p^i(CS, \mathbf{x})$ to be the total payoff to agent i under $(CS, \mathbf{x})$: $p^i(CS, \mathbf{x}) = \sum_{\mathbf{c} \in CS} x^i(\mathbf{c})$. We extend this notation to sets of agents: given a set $S \subseteq N$, we define $p^S(CS, \mathbf{x})$ to be the total payoff to the set S under $(CS, \mathbf{x})$, i.e. $p^S(CS, \mathbf{x}) = \sum_{i \in S} p^i(CS, \mathbf{x})$. Note that the definition of an outcome in an OCF game is a generalization of the respective definition in a classic cooperative game.

Example 2.2. Consider the coalition structure $CS = (\mathbf{c}_1, \mathbf{c}_2)$ from Example 2.1, and let $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2)$, where $x^1(\mathbf{c}_1) = 3$, $x^2(\mathbf{c}_1) = 7$, $x^1(\mathbf{c}_2) = 8$, $x^2(\mathbf{c}_2) = 2$. That is the payoff from the first copy of t_{12} is divided so that player 1 gets 3 and player 2 gets 7, whereas the payoff from the second copy of this task is divided so that player 2 gets 8 and player 1 gets 2. Then $\mathbf{x}$ is an imputation for CS, and $(CS, \mathbf{x})$ is an outcome of our game. Note that we have $p^1(CS, \mathbf{x}) = 11$, $p^2(CS, \mathbf{x}) = 9$.

2.2. Deviation and arbitration functions

In a classic cooperative game $\mathcal{G} = \langle N, u \rangle$, a subset of agents has an incentive to deviate from an outcome $(\Pi, \mathbf{p})$ when its total payoff $p(S)$ is less than the profits it can generate on its own, i.e. $u(S)$. However, in OCF games, deviation is a much more complicated matter. The classic notion of deviation in cooperative games implicitly assumes that when a set of agents deviates, it does not retain any ties to non-deviators; that is, the deviators measure the desirability of a deviation against the most they can make on their own, and assume that non-deviators will no longer collaborate with them. This is not necessarily the case in the OCF setting: we assume that when a set $S \subseteq N$ deviates from an outcome $(CS, \mathbf{x})$, it may still retain some resources in coalitions with agents in $N \setminus S$. Thus, to decide whether retaining connections with non-deviators is worthwhile, agents in S need to know how the non-deviators will react to their deviation. Chalkiadakis et al. [1] were the first to point out this aspect of agents' behavior, and have shown how different types of non-deviators' reactions to deviation lead to different notions of stability. Zick et al. [3] suggest a general framework for handling deviation in OCF games, which is based on the concept of *arbitration functions*.

We begin by formally defining a deviation in OCF games. Given an outcome $(CS, \mathbf{x})$ and an agent set S, a *deviation of S from* $(CS, \mathbf{x})$ is a coalition structure CS' whose coalitions describe the resources that S withdraws from each coalition $\mathbf{c} \in CS \setminus CS|_S$. We let $\mathbf{d}_{CS'}(\mathbf{c})$ denote the coalition in CS' that describes the resources that S withdraws from the coalition $\mathbf{c}$, omitting the CS' subscript when it is understood from the context. A deviation CS' by a coalition S has to satisfy the following constraints: (1) $\mathbf{d}_{CS'}(\mathbf{c}) \leq \mathbf{c}$ and (2) $\mathbf{d}_{CS'}(\mathbf{c}) \leq \mathbf{e}^S$. The first constraint ensures that no agent in S withdraws more resources than it has invested in $\mathbf{c}$, while the second constraint ensures that agents that do not belong to S do not withdraw any resources from $\mathbf{c}$. Note that the formal definition of deviation does not refer to the imputation $\mathbf{x}$.

Example 2.3. Consider again the coalition structure CS from Example 2.1 and the imputation $\mathbf{x}$ from Example 2.2; recall that $(CS, \mathbf{x})$ is an outcome of our game. Let $S = \{1\}$ and let $CS'' = (\mathbf{c}_3, \mathbf{c}_4)$, where

$$\mathbf{c}_3 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \mathbf{c}_4 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix},$$

and $\mathbf{d}_{CS''}(\mathbf{c}_1) = \mathbf{c}_3$, $\mathbf{d}_{CS''}(\mathbf{c}_2) = \mathbf{c}_4$. The coalition structure CS'' describes the deviation in which player 1 withdraws all of her resources from both of her joint projects with player 2. Observe that CS'' does not indicate how player 1 will use the withdrawn resources.

Given a deviation by S from the coalition structure CS, the non-deviating agents, i.e. the members of $N \setminus S$, have to decide how to react to the deviation. Zick et al. [3] define an *arbitration function* $\mathcal{A}$ to be a mapping whose input is an outcome $(CS, \mathbf{x})$, a deviating set $S \subseteq N$, and a deviation CS' by S from CS. For each $\mathbf{c} \in CS \setminus CS|_S$, $\mathcal{A}$ outputs a value $\alpha_{\mathbf{c}} = \alpha_{\mathbf{c}}(CS, \mathbf{x}, S, CS')$, which specifies the total payoff that the coalition $\mathbf{c}$ offers S as a result of its deviation. A deviation CS' is said to be *$\mathcal{A}$-profitable* if the members of S have a way to form a coalition structure using all resources available to them and then divide the payoffs from that coalition structure as well as the (possibly negative) payoffs provided by $(\alpha_{\mathbf{c}})_{\mathbf{c} \in CS \setminus CS|_S}$ so that every agent in S receives a strictly higher payoff compared to what she receives under $(CS, \mathbf{x})$. An outcome $(CS, \mathbf{x})$ is called *$\mathcal{A}$-stable* if no subset $S \subseteq N$ can $\mathcal{A}$-profitably deviate from $(CS, \mathbf{x})$. A game $\mathcal{G}$ is called *$\mathcal{A}$-stable* if there exists an outcome $(CS, \mathbf{x})$ that is $\mathcal{A}$-stable.

Suppose that the vector of resources available to S after it has withdrawn from CS according to CS' is given by $\mathbf{t}$; then the total payoff that S can receive from the deviation CS' under $\mathcal{A}$ is at most

$$\mathcal{A}(CS, \mathbf{x}, S, CS') = v^*(\mathbf{t}) + \sum_{\mathbf{c} \in CS \setminus CS|_S} \alpha_{\mathbf{c}}(CS, \mathbf{x}, S, CS').$$

Let

$$\mathcal{A}^*(CS, \mathbf{x}, S) = \sup\{\mathcal{A}(CS, \mathbf{x}, S, CS') \mid CS' \text{ is a deviation of } S \text{ from } (CS, \mathbf{x})\};$$

this quantity is a tight upper bound on the payoff that S can earn by deviating. Zick et al. [3] provide a simple characterization of continuous OCF games that are $\mathcal{A}$-stable, which is given in the following theorem; this result remains true in the discrete model.

Theorem 2.4. *An OCF game $\mathcal{G} = \langle N, v \rangle$ is $\mathcal{A}$-stable if and only if there exists an outcome $(CS, \mathbf{x})$ such that for all $S \subseteq N$ it holds that*

$$p^S(CS, \mathbf{x}) \geq \mathcal{A}^*(CS, \mathbf{x}, S).$$

Theorem 2.4 implies that, to verify stability, there is no need to look for an explicit coalition structure formed using the resources available to S post-deviation, a division of payoffs from that coalition structure, and a division of payoffs from non-deviators such that all deviators are strictly better off. Instead, it suffices to verify that the total payoff obtained by S is at least as large as the total payoff S can receive by deviating—no matter how it deviates.

Before we proceed, let us describe some arbitration functions, which will play an important role in this paper; formal definitions can be found in the work of Zick et al. [3] (Section 3.2).[1]

***The Conservative Arbitration Function*:** under this function, denoted $\mathcal{A}_c$, deviators receive nothing from non-deviators, i.e. $\alpha_{\mathbf{c}} \equiv 0$ for every deviation. When reasoning about the desirability of deviation under $\mathcal{A}_c$, S has no incentive to retain any of its resources in coalitions with non-deviators: it will not receive any payments from such coalitions.

***The Sensitive Arbitration Function*:** rather than assuming that agents will outright refuse to cooperate with deviators, one can take a slightly more 'lenient' approach. Under the sensitive arbitration function, denoted $\mathcal{A}_s$, if a coalition $\mathbf{c}$ is changed by S's deviation, then all agents in $supp(\mathbf{c})$ refuse to cooperate with the agents in S in other coalitions, i.e., $\alpha_{\mathbf{c}'} = 0$ for all $\mathbf{c}'$ with $supp(\mathbf{c}') \cap supp(\mathbf{c}) \neq \emptyset$. However, if none of the agents in a coalition $\mathbf{c}'$ were affected by the deviation, S retains all of its payoffs from $\mathbf{c}'$ under $(CS, \mathbf{x})$.

***The Refined Arbitration Function*:** this arbitration function, denoted $\mathcal{A}_r$, assumes an even more lenient reaction to deviation. Under $\mathcal{A}_r$, if a coalition $\mathbf{c}$ is unchanged by a deviation of a set S, S keeps all its original payoffs from $\mathbf{c}$ under $(CS, \mathbf{x})$.

***The Optimistic Arbitration Function*:** this arbitration function, denoted $\mathcal{A}_o$, describes the behavior of agents who are highly flexible concerning changes to their coalitions. Consider a deviation $\mathbf{d}(\mathbf{c})$ by a set S from a coalition $\mathbf{c} \in CS \setminus CS|_S$; after S deviates, the value of $\mathbf{c}$ is reduced to $v(\mathbf{c} - \mathbf{d}(\mathbf{c}))$. Under $\mathcal{A}_o$, the payoff to S from $\mathbf{c}$ is simply $v(\mathbf{c} - \mathbf{d}(\mathbf{c})) - \sum_{i \in N \setminus S} x^i(\mathbf{c})$. That is, to receive payoff from $\mathbf{c}$, S must cover the loss it caused by withdrawing resources from $\mathbf{c}$; once it ensures that agents in $N \setminus S$ retain their original payoffs from $\mathbf{c}$, it is free to divide the remaining payoffs from this coalition.

The following example highlights the reasoning behind some of the arbitration functions presented above.

[1] All arbitration functions listed here were introduced by Chalkiadakis et al. [1]. However, Chalkiadakis et al. [1] do not use the term 'arbitration function' to describe agents' reaction to deviation; this term was subsequently introduced by Zick et al. [3].

Example 2.5. Consider a dOCF game with $N = \{1, 2, 3\}$ and $W^i = 10$ for $i = 1, 2, 3$. Let $CS = (\mathbf{c}_{\{1,3\}}, \mathbf{c}_{\{2,3\}}, \mathbf{c}_{\{1,2,3\}})$ be a coalition structure in this game, where for every coalition $\mathbf{c}_S$ in CS we have $supp(\mathbf{c}_S) = S$; for instance, $\mathbf{c}_{\{1,3\}}$ is a coalition in which players 1 and 3 collaborate. Now, suppose that player 3 wishes to withdraw some of her resources from the coalition $\mathbf{c}_{\{1,3\}}$. Under the conservative arbitration function, player 3 will receive no payments from $\mathbf{c}_{\{1,3\}}$, $\mathbf{c}_{\{2,3\}}$ or $\mathbf{c}_{\{1,2,3\}}$. Under the sensitive arbitration function, player 3 can only expect payoff from $\mathbf{c}_{\{2,3\}}$: since she changed her contribution to $\mathbf{c}_{\{1,3\}}$—a coalition containing a player in the support of $\mathbf{c}_{\{1,2,3\}}$—she cannot expect to receive any payoff from $\mathbf{c}_{\{1,2,3\}}$. Under the refined arbitration function, she will be able to retain all of her payoff from $\mathbf{c}_{\{2,3\}}$ and $\mathbf{c}_{\{1,2,3\}}$, as they were unaffected by the deviation. Under the optimistic arbitration function, she will retain her payoffs from $\mathbf{c}_{\{2,3\}}$ and $\mathbf{c}_{\{1,2,3\}}$, and may even be eligible to keep some payoff from $\mathbf{c}_{\{1,3\}}$, depending on how much she withdrew from it, and how it affected the value of her joint project with player 1.

We observe that the amount that coalition $\mathbf{c}$ needs to pay a deviating set S may depend on the effect that S's deviation has on *other* coalitions. For example, under the sensitive arbitration function, $\mathbf{c}$ may refuse to pay S, even if $\mathbf{c}$ was not affected at all by the deviation.

In classic cooperative games if a coalition structure Π can be stabilized (i.e. if there exists an imputation $\mathbf{p}$ such that $(\Pi, \mathbf{p})$ is in the core), then every coalition structure that has the same total payoff as Π can be stabilized as well. In contrast, in games with overlapping coalitions this is not the case, as illustrated by the following example.

Example 2.6. Consider again the game described in Example 2.1 and coalition structures CS and CS' constructed in that example. We claim that CS cannot be stabilized with respect to the refined arbitration function. The reason is that for an outcome $(CS, \mathbf{x})$ to be in the refined core, it must be the case that player 2 gets at least 9 from every coalition in CS: otherwise, player 2 can threaten to deviate to a coalition with player 3, which has value 9. However, this means that player 1 gets at most 2 from working with player 2, while it can get 5 by working alone. On the other hand, let $\mathbf{y}$ be the imputation for CS' that offers an even split of the revenue from T_{12} between players 1 and 2. Then $(CS', \mathbf{y})$ is in the refined core. Notably, CS can be stabilized with respect to the sensitive and the conservative core, again, an even split of the revenue will suffice. Indeed, if player 2 withdraws resources from one of the coalitions in CS, both the sensitive and the conservative arbitration function would deny her any payoffs from the other coalition in CS'.

To conclude this section, we remark that, from the perspective of welfare maximization, discrete OCF games can be viewed as *games with types* [36–39], i.e., games where players are partitioned into k non-overlapping groups so that the agents in each group are interchangeable. Specifically, by replacing a player i with weight W^i by W^i non-splittable players and modifying the characteristic function accordingly, we obtain a classic coalitional game with $n \sum_{i=1}^{n} W^i$ players and n player types such that there is a natural correspondence between partial coalitions in the original game and (ordinary) coalitions in the new game. This connection can be used to compute an optimal overlapping coalition structure in time that is exponential in n (but not in $W_{\max}$). However, this approach is unlikely to lead to algorithms whose running time is polynomial in n, since the standard assumption in the analysis of games with types is that the number of types is small and therefore algorithms that run in time exponential in the number of types are acceptable. Moreover, the connection breaks down once we consider coalitional stability, for all but the most permissive arbitration functions: for instance, under the conservative arbitration function if player i withdraws a unit of his weight from a given partial coalition, he receives no payoff from other coalitions he contributes to; under our transformation, this would mean that all W^i 'avatars' of player i will be held responsible for the actions of any one of them, which is very different from the classic model of cooperative games.

3. Finding an optimal coalition structure

We now begin our formal computational analysis of discrete OCF games, starting with the fundamental problem of finding optimal coalition structures. We assume that the reader is familiar with standard notions of computational complexity and complexity classes (see the classic book of Garey and Johnson [40] for an overview).

We start by describing the computational model that we employ. Consider a discrete OCF game $\mathcal{G} = \langle N, \mathbf{W}, v \rangle$ with $|N| = n$, and let $W_{\max} = \max_{i \in N} W^i$. This game $\mathcal{G}$ can be described by a list of $|\mathcal{W}| = \prod_{i \in N} (W^i + 1) \leq (W_{\max} + 1)^n$ non-negative values, one for each possible coalition that the agents may form. From now on, we also assume that the characteristic function takes values in $\mathbb{Q}^+$, and for every coalition $\mathbf{c} \in \mathcal{W}$ the number of bits required to represent $v(\mathbf{c})$ is polynomial in n and $W_{\max}$; we also assume that whenever a payoff vector $\mathbf{x}$ is part of an input to a computational problem, its entries are rational numbers, and the number of bits required to represent each entry of $\mathbf{x}$ is polynomial in n and $W_{\max}$.

Since $W^i \geq 1$ for all $i \in N$, the size of this representation is exponential in n, which is unacceptable for most applications. This issue is not specific to OCF games: it also arises in the context of classic coalitional games, where it is usually tackled by focusing on classes of games that allow for a succinct encoding in a suitable representation formalism (see Chapter 3 in the book of Chalkiadakis et al. [4] and references therein). We will pursue a variant of this approach in our work: specifically, following Shehory and Kraus [41], we will limit our attention to games where there is an apriori bound on the size of a successful coalition.

Definition 3.1 *(k-OCF Games).* A discrete OCF game $\mathcal{G} = \langle N, \mathbf{W}, v \rangle$ is called a *k-OCF game* if for all $\mathbf{c} \in \mathcal{W}$ with $|supp(\mathbf{c})| > k$ it holds that $v(\mathbf{c}) = 0$.

We note that in k-OCF games the number of coalitions that can have a positive value is bounded by $\binom{n}{k}(W_{\max}+1)^k$. Therefore, if k is a constant, then the characteristic function for a k-OCF game can be represented using a number of bits that is polynomial in n and $W_{\max}$ (by omitting all coalitions with value 0). On the other hand, there are several real-life scenarios that can be captured by k-OCF games where k is a small constant: in many market scenarios, transactions involve only a few parties; in social network applications, agents form pairwise coalitions; in many large-scale collaborative projects, small teams are formed to tackle various tasks, as large teams of collaborators tend to be less efficient. Thus, k-OCF games with small k provide a good balance between succinctness and expressivity.

In what follows, it will be convenient to use the following notation: given a set of agents $\{i_1, \dots, i_k\}$, we write $v_{i_1,\dots,i_k}(w_{i_1}, \dots, w_{i_k})$ to denote the value of v when agent i_1 contributes w_{i_1}, agent i_2 contributes w_{i_2}, etc. For example, if agents i and j embark on a joint project where agent i contributes w_i and agent j contributes w_j, the value of this collaboration is $v_{i,j}(w_i, w_j)$.

We are now ready to formulate the computational problem that is the focus of this section: computing the superadditive cover for a given resource vector $\mathbf{c}$, i.e. finding an overlapping coalition structure that maximizes the total profit. The decision version of this problem can be stated as follows.

Name: OptVal
Input: A discrete OCF game $\mathcal{G} = \langle N, \mathbf{W}, v \rangle$, a coalition $\mathbf{c} \in \mathcal{W}$, and a value $V \in \mathbb{Q}$.
Question: Is $v^*(\mathbf{c}) \geq V$?

In the rest of this section, we investigate the complexity of OptVal, aiming to identify the properties of dOCF games that make this problem tractable.

3.1. Complexity of OptVal in general k-OCF games

The following simple proposition is an important first step in understanding the complexity of our problem.

Proposition 3.2. *Given a game $\mathcal{G} = \langle N, \mathbf{W}, v \rangle$ and a coalition $\mathbf{c} \in \mathcal{W}$ with $|supp(\mathbf{c})| = m$, we can compute $v^*(\mathbf{c})$ in time $O((W_{\max}+1)^{2m})$.*

Proof. Observe that

$$v^*(\mathbf{c}) = \max\left\{v(\mathbf{c}), \max\{v^*(\mathbf{c}-\mathbf{d}) + v(\mathbf{d}) \mid \mathbf{d} \leq \mathbf{c}; \mathbf{d} \neq \mathbf{c}\}\right\}.$$

Further, the number of coalitions $\mathbf{d}$ with $\mathbf{d} \leq \mathbf{c}$ is at most $(W_{\max}+1)^m$. Using this recurrence, we can compute the value $v^*(\mathbf{c}')$ of each coalition $\mathbf{c}' \leq \mathbf{c}$ by dynamic programming in time $O((W_{\max}+1)^m)$; since there are at most $(W_{\max}+1)^m$ such coalitions, the bound on the running time follows. We can also find a coalition structure CS with $w(CS) = \mathbf{c}$ such that $v(CS) = v^*(\mathbf{c})$ using standard dynamic programming techniques. □

Proposition 3.2 implies that if $|supp(\mathbf{c})|$ is bounded by a constant, then $v^*(\mathbf{c})$ can be computed in time polynomial in n and $W_{\max}$. Therefore, from now on we focus on finding the value of an optimal coalition structure for a large group of players. Note that this problem is non-trivial even if we focus on k-OCF games with a small value of k: we need to distribute the resources of a large group of players over many small (overlapping) coalitions. Indeed, even in the context of non-overlapping coalitions (which corresponds to games with $W_{\max} = 1$ in our model) it is NP-hard to find an optimal coalition structure [9], and this hardness result holds even if all coalitions with positive value have size at most 3 (in our model this corresponds to 3-OCF games). For discrete OCF games we can prove a stronger result: OptVal is NP-hard even for 2-OCF games; moreover, this is true even if $W_{\max} = 3$.

Proposition 3.3. OptVal *is* NP*-complete. The hardness result holds even for instances $(\mathcal{G}, \mathbf{c}, V)$ such that $\mathcal{G}$ is a 2-OCF game with $W_{\max} = 3$.*

Proof. First, we observe that OptVal is in NP: it suffices to guess a coalition structure CS such that $\mathbf{w}(CS) = \mathbf{c}$ and check whether $v(CS) \geq V$. Note that the size of this coalition structure is at most $nW_{\max}$, which is polynomial in the input size.

For the hardness proof, we provide a reduction from Exact Cover by 3-Sets (X3C) [40]. Recall that an instance $\mathcal{X} = \langle A, \mathcal{S} \rangle$ of X3C is given by a finite set A, $|A| = 3\ell$, and a collection of subsets $\mathcal{S} = \{S_1, \dots, S_t\}$ such that $S_j \subseteq A$ and $|S_j| = 3$ for all $j = 1, \dots, t$. It is a "yes"-instance if A admits an exact cover by sets from $\mathcal{S}$; that is, if there exists a subset $\mathcal{S}' \subseteq \mathcal{S}$ such that $\bigcup_{S \in \mathcal{S}'} S = A$, and for any two distinct $S, T \in \mathcal{S}'$ we have $S \cap T = \emptyset$ (note that this implies $|\mathcal{S}'| = \ell$).

Given an instance $\mathcal{X} = \langle A, \mathcal{S} \rangle$ of X3C, we construct a discrete 2-OCF game $\mathcal{G}(\mathcal{X}) = \langle N, \mathbf{W}, v \rangle$ with $W_{\max} = 3$ as follows. We have an agent a_i of weight 1 for every element $i \in A$ and an agent a_S with weight 3 for every $S \in \mathcal{S}$. The characteristic

function is defined as follows. If $i \in A$, $S \in \mathcal{S}$ and $i \in S$, then the value of the coalition $\mathbf{c}$ with $supp(\mathbf{c}) = \{a_i, a_S\}$ and $c^{a_i} = c^{a_S} = 1$ is 2. Further, if $S \in \mathcal{S}$ then the value of the coalition $\mathbf{c}$ with $supp(\mathbf{c}) = \{a_S\}$ and $c^{a_S} = 3$ is 5. The value of every other partial coalition in the game $\mathcal{G}(\mathcal{X})$ is 0.

Consider a set $S = \{x, y, z\}$ in $\mathcal{S}$, and the respective set of agents $G_S = \{a_S, a_x, a_y, a_z\}$. Collectively, the agents in G_S can earn 6 if a_S forms a partial coalition with each of a_x, a_y, and a_z, and contributes one unit of weight to each of these coalitions; in any other coalition structure, agents in G_S earn at most 5. We will now use this observation to argue that $\mathcal{X} = \langle A, \mathcal{S}\rangle$ admits an exact cover if and only if $v^*(\mathbf{W}) \geq 6\ell + 5(t - \ell) = 5t + \ell$.

Indeed, if $\mathcal{X}$ is a "yes"-instance of X3C, then there exists some subset $\mathcal{S}' \subseteq \mathcal{S}$ of size ℓ that exactly covers A. In that case,

$$v^*(\mathbf{W}) \geq \sum_{S \in \mathcal{S}'} v^*(\mathbf{W}^{G_S}) + \sum_{S \notin \mathcal{S}'} v^*(\mathbf{W}^{a_S}) = 6\ell + 5(t - \ell).$$

Conversely, suppose that $\mathcal{X}$ is a "no"-instance of X3C, and consider some coalition structure CS for $\mathbf{W}$ such that $v^*(\mathbf{W}) = v(CS)$. We say that an agent a_S with $S \in \mathcal{S}$, $S = \{x, y, z\}$ is *happy in* CS if she forms coalitions with each of a_x, a_y and a_z, i.e. if CS contains coalitions $\mathbf{c}_x$, $\mathbf{c}_y$, and $\mathbf{c}_z$, where for each $\xi \in \{x, y, z\}$ we have $supp(\mathbf{c}_\xi) = \{a_S, a_\xi\}$, $c_\xi^{a_\xi} = c_\xi^{a_S} = 1$. Let $H(CS)$ denote the set of all agents who are happy in CS. Since every coalition with a positive value contains an agent a_S with $S \in \mathcal{S}$, we have

$$\begin{aligned} v^*(\mathbf{W}) = v(CS) &\leq \sum_{a_S \in H(CS)} \sum_{\substack{\mathbf{c} \in CS \\ a_S \in supp(\mathbf{c})}} v(\mathbf{c}) + \sum_{a_S \notin H(CS)} \sum_{\substack{\mathbf{c} \in CS \\ a_S \in supp(\mathbf{c})}} v(\mathbf{c}) \\ &\leq 6|H(CS)| + 5(t - |H(CS)|) = 5t + H(CS). \end{aligned}$$

Note that if a_S and a_T are both happy in CS and $S \neq T$, then S and T are disjoint. Since there are at most $\ell - 1$ pairwise disjoint sets in $\mathcal{S}$, this means that the value of any coalition structure for $\mathbf{W}$ does not exceed $5t + \ell - 1 < 5t + \ell$. □

We remark that, in contrast, in the non-overlapping setting finding an optimal partition of players into coalitions of size 1 and 2 reduces to the problem of finding a maximum-weight matching, and is therefore polynomial-time solvable [42].

Note that Proposition 3.3 is tight with respect to k: for 1-OCF games an optimal coalition structure consists of singletons and therefore OptVal reduces to computing $v^*(\mathbf{e}^i)$ for each $i \in N$, which is easy by Proposition 3.2. However, it is not clear if it is tight with respect to $W_{\max}$. Indeed, the case $W_{\max} = 1$ corresponds to the non-overlapping setting, and therefore, by the argument in the previous paragraph OptVal is easy for 2-OCF games with $W_{\max} = 1$. On the other hand, the complexity of OptVal in 2-OCF games with $W_{\max} = 2$ remains an open problem. However, even $W_{\max} = 3$ is a very strong constraint, and therefore it is fair to say that Proposition 3.3 severely limits our prospects of computing optimal coalition structures: even if agents are limited to pairwise interactions, and their weights are small constants, OptVal remains hard. Observe that the hardness of OptVal implies the hardness of most other problems of interest in the study of OCF games, such as computing the most that a set can gain by deviating, determining whether a given outcome of a game is stable, or deciding whether a given game admits a stable outcome. Thus, in order to proceed, we must first identify further constraints that make OptVal computationally tractable.

3.2. Constraining communication in OCF games

Demange [6] shows that if one assumes a hierarchical agent communication structure in a cooperative game, then the core of the game is not empty; moreover, if this game is superadditive, it is possible to find a core imputation in polynomial time. We will now show that this idea can also be used in the context of OCF games. Specifically, we make use of the concept of an *agent communication graph* [5,6]: this is a graph $\Gamma = \langle N, E\rangle$, where N is the set of agents and the edges in E represent valid agent interactions. Given an OCF game $\mathcal{G} = \langle N, \mathbf{W}, v\rangle$ and a communication graph $\Gamma = \langle N, E\rangle$, we define the game $\mathcal{G}$ *reduced to* Γ, denoted $\mathcal{G}|_\Gamma = \langle N, \mathbf{W}, v|_\Gamma\rangle$, as follows. For every $\mathbf{c} \in \mathcal{W}$, if the nodes in $supp(\mathbf{c})$ induce a connected subgraph of Γ then $v|_\Gamma(\mathbf{c}) = v(\mathbf{c})$, otherwise $v|_\Gamma(\mathbf{c}) = 0$. We say that a discrete k-OCF game $\mathcal{G}$ has *tree communication structure* if there exists some tree $T = \langle N, E\rangle$ such that $\mathcal{G} = \mathcal{G}|_T$.

We will now show that we can compute optimal coalition structures in discrete 2-OCF games with tree communication structure in time polynomial in n and $W_{\max}$, thus overcoming the hardness result of Proposition 3.3. Note that, given a discrete 2-OCF game $\mathcal{G}$, we can easily decide if it has tree communication structure, and identify the corresponding tree in time polynomial in n and $W_{\max}$: we construct Γ by adding an edge $\{i, j\}$ whenever there is coalition $\mathbf{c}$ in $\mathcal{G}$ with $v(\mathbf{c}) \neq 0$, $supp(\mathbf{c}) = \{i, j\}$, and then checking whether Γ is a tree.

Before we present our algorithm, we need to introduce additional notation for games on graphs, which will also be used later in the paper. Given a tree $T = \langle N, E\rangle$, pick an arbitrary vertex $r \in N$ to be a root. Let T_r be the resulting rooted tree; for each player $i \in N$, let T_r^i be the subtree of T_r rooted in i, and let $C^i(T_r)$ be the children of i in T_r. We omit the subscript r from the notation when it is clear from the context. Let $N(\Gamma)$ denote the set of nodes of a graph Γ. We can now state our positive result for 2-OCF games on trees.

Theorem 3.4. *Given a discrete 2-OCF game $\mathcal{G} = \langle N, \mathbf{W}, v\rangle$ with tree communication structure, we can compute $v^*(\mathbf{c})$ for every $\mathbf{c} \in \mathcal{W}$ in time polynomial in n and $W_{\max}$.*

Proof. It suffices to show how to compute $v^*(\mathbf{W})$: for an arbitrary $\mathbf{c} \in \mathcal{W}$ we can simply consider the game where agent weights are given by $\mathbf{c}$.

Let T be the communication graph of $\mathcal{G}$. We choose an arbitrary player $r \in N$ to be the root of T, and process the players in N from the leaves up to the root. The key observation used in our proof is that, in order to find an optimal allocation of agents' resources, we need to decide, for each node in T, how to split its weight among collaborating with its subtree, collaborating with its parent, and working alone. Note that, by Proposition 3.2, for every pair of agents $i, j \in N$ and every pair of values $x \in \{0, \dots, W^i\}$, $y \in \{0, \dots, W^j\}$ the quantities $v_i^*(x)$ and $v_{i,j}^*(x, y)$ can be computed in time polynomial in $W_{\max}$. Given a node $i \in N$, let $C^i = C^i(T)$ be the children of i in T, and let $v_{T^i}^*(w)$ be the most that the agents in the subtree T^i can make by working together (i.e. not counting the profit from i's collaboration with its parent) if agent i allocates w units of weight to working with agents in T^i (and $W^i - w$ units of weight to working with its parent). We have

$$v_{T^i}^*(w) = \max_{\substack{y^i + \sum_{a \in C^i} w^a = w \\ 0 \le x^a \le W^a \ \forall a \in C^i}} \left\{ v_i^*(y^i) + \sum_{a \in C^i} \left(v_{i,a}^*(w^a, x^a) + v_{T^a}^*(W^a - x^a) \right) \right\}. \tag{1}$$

Using this recurrence relation, we obtain the following dynamic programming algorithm for finding an optimal coalition structure.

Suppose that we have already computed $v_{T^a}^*(w)$ for all $a \in C^i$ and all $w = 0, \dots, W^a$. Now, let us write $C^i = \{a^1, \dots, a^m\}$, and for $j = 0, \dots, m-1$ let $T^{i,j}$ be the tree obtained from T^i by removing the subtrees $T^{a^{j+1}}, \dots, T^{a^m}$. Observe that $T^{i,0}$ is the tree obtained from T^i by removing all of its subtrees, i.e. the tree comprised of the singleton $\{i\}$. Let $v_{T^{i,j}}^*(w)$ be the maximum revenue that can be generated if agent i invests w units of weight in working with $T^{i,j}$.

Note that $v_{T^{i,0}}^*(w) = v_i^*(w)$, and is therefore easy to compute by Proposition 3.2. Now, suppose that we have already computed $v_{T^{i,j'}}^*(w)$ for all $j' = 0, \dots, j-1$ and all $w = 0, \dots, W^i$. Then we can compute $v_{T^{i,j}}^*(w)$ in time polynomial in $W_{\max}$ as

$$v_{T^{i,j}}^*(w) = \max_{\substack{x+z=w \\ 0 \le y \le W^{a^j}}} \left\{ v_{i,a^j}^*(x, y) + v_{T^{i,j-1}}^*(z) + v_{T^{a^j}}^*(W^{a^j} - y) \right\}. \tag{2}$$

Finally, we have $v_{T^i}^*(w) = v_{T^{i,m}}^*(w)$, i.e. for any given value of $w \in \{0, \dots, W^i\}$ we can compute $v_{T^i}^*(w)$ in time polynomial in $W_{\max}$ and $|C^i|$. We process all nodes from the leaves to the root in this manner; the value of an optimal coalition structure for $\mathbf{w}$ is given by $v_{T^r}^*(W^r)$. □

Example 3.5. Consider a discrete 2-OCF game, where $N = \{1, 2, 3, 4\}$ and the set of edges of the communication graph is

$$\{\{1, 3\}, \{2, 3\}, \{3, 4\}\}.$$

Suppose we choose player 4 to be the root. Then our dynamic program first computes the values $v_{T^1}^*(w)$ for $w = 0, \dots, W^1$ and $v_{T^2}^*(w)$ for $w = 0, \dots, W^2$, to determine what players 1 and 2 can accomplish on their own, for every possible weight.

It then considers player 3. For each $w = 0, \dots, W^3$, it computes what player 3 can accomplish alone given weight w. Then it considers the tree that consists of players 1 and 3, and for each $w = 1, \dots, W^3$, it computes the total profit that this tree can generate assuming player 3 allocates w units of weight to it. To do so, it 'guesses' how much weight player 3 and player 1 allocate to working together (by going over all $(W^1 + 1)(w + 1)$ possibilities), evaluates the productivity of this collaboration by applying $v_{1,3}^*$, looks up how much profit player 1 and player 3 can generate on their own given the remaining resources, and picks one of the $(W^1 + 1)(w + 1)$ possibilities with the highest total profit.

Next, the algorithm considers the tree that consists of players 1, 2, and 3. For each $w = 0, \dots, W^3$, it guesses how much weight each of the players 2 and 3 allocates to their joint project; player 2 allocates the rest of his weight to working alone, and player 3 allocates the rest of her weight to the tree that consists of 1 and 3, and the best such allocations have been computed at earlier steps; again, we choose the best of these $(W^2 + 1)(w + 1)$ possibilities with regard to the total profit.

Finally, the algorithm considers player 4. For each $w = 0, \dots, W^4$, it computes what he can earn on his own given weight w. It then goes over all possible collaborations between player 3 and player 4. For the case where player 3 allocates x units of weight to working with player 4, and player 4 allocates y units of weight to working with player 3, it computes $v_{3,4}^*(x, y)$ to evaluate this collaboration, and looks up (a) what player 4 can accomplish given weight $W^4 - y$ and (b) what players 1, 2, and 3 can accomplish given that player 3 allocates $W^3 - x$ units of weight to working alone or with players 1 and 2, and adds up these three quantities; note that (a) and (b) have been computed by our dynamic program during previous steps. Finally, it chooses the pair (x, y) that maximizes the total profit, and outputs the respective profit.

The reader may wonder if it is necessary to restrict both the structure of the communication graph and the maximum coalition size: indeed, perhaps OptVal is easy for discrete k-OCF games on trees even if k is large? Unfortunately, our next result shows that this is not the case, even for classic coalitional games (i.e. even if $W_{\max} = 1$).

Proposition 3.6. OptVal *is* NP*-hard even if the input game* $\mathcal{G} = \langle N, \mathbf{W}, v \rangle$ *has tree communication structure and* $W_{\max} = 1$.

Proof. Our reduction is from the Vertex Cover problem [40]. An instance of Vertex Cover is a tuple $\langle \Gamma, m \rangle$, where $\Gamma = \langle A, E \rangle$ is a graph and m is an integer. It is a "yes"-instance if Γ admits a vertex cover of size at most m, i.e. there exists a subset of vertices $S \subseteq A$ such that $|S| \leq m$ and $\{i, j\} \cap S \neq \emptyset$ for every $\{i, j\} \in E$.

Given an instance $\langle \Gamma, m \rangle$ of Vertex Cover with $\Gamma = \langle A, E \rangle$ and $A = \{1, \ldots, n\}$, we construct the following instance of OptVal. We set the player set to be $N = A \cup \{n+1\}$. The communication graph T is a star with center $n+1$; all vertices in A are leaves of T. We set $W^i = 1$ for $i \in N$, i.e. our game is equivalent to a classic coalitional game. The characteristic function v is defined as follows. If $supp(\mathbf{c}) \cap A$ is a vertex cover for Γ and $c^{n+1} = 1$, then $v(\mathbf{c}) = n^2$. If $|supp(\mathbf{c})| = 1$, then $v(\mathbf{c}) = 1$. For any other coalition $\mathbf{c} \in \mathcal{W}$ we set $v(\mathbf{c}) = 0$. Clearly, an optimal coalition structure in this game consists of a coalition $S \cup \{n+1\}$, where S is a minimum-size vertex cover for Γ, and $n - |S|$ singletons; its value is $n^2 + (n - |S|)$. Thus, we have $v^*(\mathbf{W}) \geq n^2 + n - m$ if and only if Γ admits a vertex cover of size m. □

To summarize, Propositions 3.3 and 3.6 indicate that, in order to find an optimal coalition structure in polynomial time, we need to both limit the size of each partial coalition in our coalition structure and impose a structural constraint on the overall communication pattern; dropping either of these two conditions makes our problem computationally hard (however, in Section 6 we will see that we can relax the latter constraint from trees to graphs of bounded treewidth). Note also that, to specify the profits that can be earned by a single agent i in an OCF game, we need to provide W^i values, so there is little hope that any of our problems admit an algorithm whose running time is $o(W_{\max})$.

In what follows, we only consider k-OCF games with $k = 2$. Extending our techniques to larger values of k would require us to reason about hypergraphs with far more complex interactions. Indeed, in Theorem 3.4, in order to decide on the best resource allocation for player i, one needs to consider how much i should allocate to its neighbors—i.e. its parent and its children—and to itself. Allowing for interactions beyond one's neighbors (as is the case for k-OCF games with $k > 2$) would require us to consider much more complex collaboration patterns. While we conjecture that many of the results in this paper can be generalized to the case $k > 2$, we leave the formal analysis of this case for future work.

4. Computing the profit from a deviation

In Section 3 we identified two key conditions for the computational tractability of *finding an optimal coalition structure* in an OCF game. The next computational problem that we would like to investigate is computing the maximum profit that a set of agents can obtain by deviating from a given outcome; understanding the complexity of this problem is a prerequisite for the computational analysis of stability in OCF games. Formally, for any given arbitration function $\mathcal{A}$, we consider the following problem.

Name: $\mathcal{A}$-ArbVal
Input: A discrete OCF game $\mathcal{G} = \langle N, \mathbf{W}, v \rangle$, a set $S \subseteq N$, an outcome $(CS, \mathbf{x})$ of $\mathcal{G}$, and a value $V \in \mathbb{Q}$.
Question: Is $\mathcal{A}^*(CS, \mathbf{x}, S) \geq V$?

Note that if $\mathcal{A}$ is the conservative arbitration function then $\mathcal{A}$-ArbVal is no harder than OptVal: if the deviators obtain no payoffs from their joint projects with non-deviators regardless of the nature of their deviation, then the most they can make by deviating is exactly what they can earn on their own. However, for more complicated arbitration functions, $\mathcal{A}$-ArbVal can be more complex than OptVal: in order to ensure that $\mathcal{A}$-ArbVal can be decided in polynomial time, one must make assumptions not only about the structure of the game $\mathcal{G}$, but also about the properties of the arbitration function $\mathcal{A}$. In particular, the following proposition illustrates that it is not sufficient to require that $\mathcal{A}$ is polynomial-time computable.

Proposition 4.1. *There exists a polynomial-time computable arbitration function* $\mathcal{A}$ *and a family of dOCF games with* 2 *players such that if there exists an algorithm for* $\mathcal{A}$-ArbVal *that runs in polynomial time on instances from this family then* P = NP.

Proof. We will show that if such an algorithm exists, it can be used to solve instances of Set Cover [40]. Recall that an instance of Set Cover is given by a set of elements A, a collection of subsets $\mathcal{S} = \{S_1, \ldots, S_t\} \subseteq 2^A$ and $\ell \in \mathbb{N}$; it is a "yes"-instance if A can be covered by at most ℓ sets from $\mathcal{S}$.

Given an instance $\langle A, \mathcal{S}, \ell \rangle$ of Set Cover with $|\mathcal{S}| = t$, consider a 2-player discrete OCF game where $W^1 = W^2 = t + 2$. We define v in the following manner. Either player gets a payoff of 1 for each unit of weight devoted to working alone, i.e. $v_1(x) = v_2(x) = x$ for all $x \in \{0, \ldots, t+2\}$. We also set $v_{1,2}(1,1) = 2$, and $v_{1,2}(2,2) = 10(t+2)$. All other coalitions have value 0.

We construct an outcome $(CS, \mathbf{x})$ as follows. Players 1 and 2 form $t+1$ coalitions: both of them allocate one unit of weight each to each of the first t coalitions, and two units of weight to the last coalition. That is, we set

$CS = (\mathbf{c}_1, \ldots, \mathbf{c}_t, \mathbf{c}_{t+1})$, where $\mathbf{c}_j = (1,1)$ for $j = 1, \ldots, t$, and $\mathbf{c}_{t+1} = (2,2)$. We define $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_t, \mathbf{y})$ as follows: for $j = 1, \ldots, t$, $\mathbf{x}_j = (0,2)$ is the payoff division that corresponds to $\mathbf{c}_j$, and $\mathbf{y} = (5(t+2), 5(t+2))$ corresponds to $\mathbf{c}_{t+1}$. In other words, we allocate the payoffs from $\mathbf{c}_1, \ldots, \mathbf{c}_t$ to player 2, and split the payoff from $\mathbf{c}_{t+1}$ equally between the players.

We define the arbitration function $\mathcal{A}$ as follows. If player 1 wishes to deviate from $(CS, \mathbf{x})$ by withdrawing from coalitions $\mathbf{c}_{j_1}, \ldots, \mathbf{c}_{j_s}$, she receives no payoff from coalitions $\mathbf{c}_1, \ldots, \mathbf{c}_t$, and only gets to keep her payoff from $\mathbf{c}_{t+1}$ if the collection $\{S_j \in \mathcal{S} \mid j \notin \{j_1, \ldots, j_s\}\}$ is a set cover for A. On all other inputs $\mathcal{A}$ behaves as the refined arbitration function. Note that $\mathcal{A}$ is polynomial-time computable. Under this arbitration function, player 1 has two reasonable choices: either not deviate at all or withdraw as much weight as possible from $\mathbf{c}_1, \ldots, \mathbf{c}_t$, while ensuring that the coalitions she keeps intact correspond to a set cover of A.

We observe that $\mathcal{A}^*(CS, \mathbf{x}, \{1\}) \geq 5(t+2) + t - \ell$ if and only if $\langle A, \mathcal{S}, \ell \rangle$ is a "yes"-instance of Set Cover. Indeed, if there is a set cover $\mathcal{S}' \subseteq \mathcal{S}$ such that $|\mathcal{S}'| \leq \ell$, then by withdrawing from the coalitions corresponding to $\mathcal{S} \setminus \mathcal{S}'$, and allocating the withdrawn resources to working alone, player 1 ensures that she receives a payoff of at least $5(t+2) + t - \ell$. On the other hand, if there exists a collection of coalitions $CS' \subseteq CS$ such that withdrawing from CS' ensures that the payoff to player 1 is at least $5(t+2) + t - \ell$, then $\mathbf{c}_{t+1}$ does not appear in CS', and, moreover, CS' corresponds to a set $\mathcal{S}' \subseteq \mathcal{S}$ with $|\mathcal{S}'| \geq t - \ell$ such that $\mathcal{S} \setminus \mathcal{S}'$ is a set cover of A. □

Remark 4.2. We contrast Proposition 4.1 with Proposition 3.2: computing the most that a set of agents can earn with a given resource vector is computationally much easier than deciding what is the most it stands to gain by deviating. This issue does not arise in classic cooperative games: a set S assesses the desirability of deviation by considering the most it can make on its own, which only requires computing $v^*(S)$.

The proof of Proposition 4.1 indicates that the hardness of deciding $\mathcal{A}$-ArbVal stems from the fact that the payoff from a coalition $\mathbf{c}$ to a deviating set S may be influenced by how the deviation of S affects other coalitions. Indeed, in the reduction used in Proposition 4.1, the payoff to player 1 from coalition $\mathbf{c}_{t+1}$ was determined by how agent 1 withdraws her weight from other coalitions. In other words, the arbitration function determines the payoff to a deviating set S based on the global behavior of S.

This observation motivates the definition of a *local* arbitration function. Intuitively, an arbitration function $\mathcal{A}$ is said to be local if the payoff that the deviating set S receives from a coalition $\mathbf{c}$ under $\mathcal{A}$ depends only on the effect of S on $\mathbf{c}$, and not on other inputs to $\mathcal{A}$. This intuition is formally captured by the following definition.

Definition 4.3. An arbitration function $\mathcal{A}$ is *local* if for every game $\mathcal{G} = \langle N, \mathbf{W}, v \rangle$, every outcome $(CS, \mathbf{x})$ of $\mathcal{G}$, every set $S \subseteq N$, every coalition $\mathbf{c} \in CS \setminus CS|_S$ and every pair of deviations CS', CS'' of S from CS such that $\mathbf{d}_{CS'}(\mathbf{c}) = \mathbf{d}_{CS''}(\mathbf{c})$ it holds that $\alpha_{\mathbf{c}}(CS, \mathbf{x}, S, CS') = \alpha_{\mathbf{c}}(CS, \mathbf{x}, S, CS'')$.

We note that the conservative, refined and optimistic arbitration functions are local: the payoff from the conservative arbitration function is 0 for all inputs; the payment from the refined arbitration function is $\sum_{i \in S} x^i(\mathbf{c})$ if $\mathbf{d}(\mathbf{c}) = 0^n$ and is 0 otherwise, and the payment from the optimistic arbitration function is $\max\{v(\mathbf{c} - \mathbf{d}(\mathbf{c})) - \sum_{i \notin S} x^i(\mathbf{c}), 0\}$. In contrast, the arbitration function used in the proof of Theorem 4.1 is non-local. Another example of a non-local arbitration function is the sensitive arbitration function, as the payoff to a set from a coalition $\mathbf{c}$ depends on which agents in the support of $\mathbf{c}$ were hurt by the deviation of S from other coalitions.

When one is limited to the class of efficiently computable local arbitration functions, it is indeed possible to decide $\mathcal{A}$-ArbVal in time polynomial in $|CS|$ and $(W_{\max} + 1)^r$, where r is the size of the deviating set. The next theorem presents an algorithm for the optimization version of this problem.

Theorem 4.4. *If $\mathcal{A}$ is a polynomial-time computable local arbitration function then for any discrete OCF game $\mathcal{G} = \langle N, \mathcal{W}, v \rangle$, an outcome $(CS, \mathbf{x})$ of this game and a set $S \subseteq N$ with $|S| = r$ we can compute $\mathcal{A}^*(CS, \mathbf{x}, S)$ in time polynomial in r and $(W_{\max} + 1)^r$.*

Proof. Let $CS^{\cap}$ be the set of coalitions that involve both S and $N \setminus S$, i.e.

$$CS^{\cap} = \{\mathbf{c} \in CS \mid supp(\mathbf{c}) \cap S \neq \emptyset; supp(\mathbf{c}) \cap (N \setminus S) \neq \emptyset\}.$$

Players in S invest a weight of $\mathbf{s} = \mathbf{w}(CS|_S)$ in partial coalitions among themselves. If they withdraw an additional weight of $\mathbf{t}$ from CS, their total payoff would then be $v^*(\mathbf{s} + \mathbf{t})$, plus the most that S can get from the arbitration function, which depends on the coalitions affected by this deviation. Thus, in order to determine the most that S can get by deviating from $(CS, \mathbf{x})$, given that it withdraws a total weight of $\mathbf{t}$, we must determine how to best withdraw this weight from $CS^{\cap}$. We write $CS^{\cap} = (\mathbf{c}_1, \ldots, \mathbf{c}_m)$; since CS has at most $(W_{\max} + 1)r$ coalitions that involve players in S, we have $m \leq (W_{\max} + 1)r$.

Given a coalition $\mathbf{c} \in CS^{\cap}$ and a weight vector $\mathbf{z} \leq \mathbf{c}$, we write $\alpha_{\mathbf{c}}(\mathbf{z})$ to denote the payoff from $\mathbf{c}$ that is allocated by $\mathcal{A}$ to S if S withdraws $\mathbf{z}$ from $\mathbf{c}$; this quantity is polynomial-time computable, and, since $\mathcal{A}$ is local, it does not depend on how S's deviation affects other coalitions. Given a weight vector $\mathbf{y}$ with $\mathbf{y} \leq \mathbf{W}^S$, let us denote by $A(\mathbf{y}; \ell)$ the most that the arbitration function $\mathcal{A}$ would give S if it withdraws $\mathbf{y}$ from the first ℓ coalitions in $CS^{\cap}$, where $\ell = 1, \ldots, m$; we set $A(\mathbf{y}; \ell) = -\infty$ if it is not possible to withdraw $\mathbf{y}$ from $\mathbf{c}_1, \ldots, \mathbf{c}_\ell$. Let $A^{\cap}(\mathbf{y}) = A(\mathbf{y}; m)$.

By definition, $A(\mathbf{y}; 1) = \alpha_{\mathbf{c}_1}(\mathbf{y})$ if $\mathbf{y} \leq \mathbf{c}_1$ and $A(\mathbf{y}; 1) = -\infty$ otherwise. For $\ell > 1$ we have

$$A(\mathbf{y}; \ell) = \max\{A(\mathbf{z}; \ell - 1) + \alpha_{\mathbf{c}_\ell}(\mathbf{y} - \mathbf{z}) \mid \mathbf{z} \geq 0^n, \mathbf{z} \leq \mathbf{y}, \mathbf{z} \leq \mathbf{c}_\ell\}.$$

This shows that we can compute $A^{\cap}(\mathbf{y}) = A(\mathbf{y}; m)$ in time $\mathcal{O}(m(W_{\max} + 1)^r)$. Finally, $\mathcal{A}^*(CS, \mathbf{x}, S)$ can be computed as $\max\{v^*(\mathbf{s} + \mathbf{t}) + A^{\cap}(\mathbf{t}) \mid 0^n \leq \mathbf{t} \leq \mathbf{W}^S - \mathbf{s}\}$, which concludes the proof. □

We have argued that it is easy to compute an optimal coalition structure for 2-OCF games where the communication graph is a tree. It turns out that a similar result holds for the problem of computing the maximum amount that a set can get by deviating. Indeed, the proof of Theorem 4.5 (below) builds on that of Theorem 3.4.

Theorem 4.5. *If $\mathcal{A}$ is a polynomial-time computable local arbitration function then $\mathcal{A}$-ArbVal is decidable in time polynomial in n and $W_{\max}$ whenever the input game $\mathcal{G}$ is a discrete 2-OCF game with a tree communication structure.*

Proof. Recall that an instance of our problem is given by a discrete 2-OCF game $\mathcal{G} = \langle N, \mathbf{W}, v\rangle$, an outcome $(CS, \mathbf{x})$, a set S, and a value V. Let T be the communication tree associated with $\mathcal{G}$. We can assume without loss of generality that S is connected in T; indeed, if this is not the case, we can consider each connected component of S separately, find an optimal deviation for it, and add up the resulting payoffs (this argument uses the assumption that our arbitration function is local and every coalition with non-zero value has at most two players in its support).

We choose an arbitrary $r \in S$ to be the root of the communication tree. Since S is connected, this ensures that the parent of each agent in $S \setminus \{r\}$ is also in S. We will now construct a new discrete 2-OCF game $\mathcal{G}' = \langle S, \mathbf{U}, u\rangle$ that has S as its set of players and satisfies $u^*(\mathbf{U}) = \mathcal{A}^*(CS, \mathbf{x}, S)$; we can then use the algorithm from the proof of Theorem 3.4 to compute $u^*(\mathbf{U})$. The game $\mathcal{G}'$ is constructed as follows. For each $i \in S$ we set $U^i = W^i$. Moreover, for every pair of agents $i, j \in S$ and weights w^i, w^j with $0 \leq w^i \leq U^i$, $0 \leq w^j \leq U^j$, we set $u_{i,j}(w^i, w^j) = v_{i,j}(w^i, w^j)$. It remains to describe how to compute $u_i(w^i)$ for $i \in S$; our goal is to define u on such coalitions so as to capture the payoff that an agent in S could earn in $\mathcal{G}$ by collaborating with her non-deviating neighbors and working on her own.

Consider an agent i, and let D^i be the set of non-S neighbors of i in T. Let $\mathbf{c}_1, \ldots, \mathbf{c}_m$ be the list of coalitions in CS that i forms with members of D^i. For each $\ell = 1, \ldots, m$ and $0 \leq z \leq c^i_\ell$, let $A^i(\mathbf{c}_\ell, z)$ denote the payoff from $\mathbf{c}_\ell$ that $\mathcal{A}$ allocates to i if i leaves z units of her weight invested in $\mathbf{c}_\ell$; this quantity is well-defined since $\mathcal{A}$ is a local arbitration function. Further, for $\ell = 0, \ldots, m$ and $0 \leq y \leq \sum_{j=1}^{\ell} c^i_j$, let $\beta^i(y; \ell)$ be the maximum payoff that i can obtain under $\mathcal{A}$ from $\mathbf{c}_1, \ldots, \mathbf{c}_\ell$ if she leaves y units of her weight invested in these coalitions. We have $\beta^i(0; 0) = 0$ and

$$\beta^i(y; \ell) = \max_{\substack{0 \leq z \leq \min\{y, c^i_\ell\} \\ y - z \leq \sum_{j=1}^{\ell-1} c^i_j}} \left\{\beta^i(y - z; \ell - 1) + A^i(\mathbf{c}_\ell, z)\right\}. \tag{3}$$

Now, we define

$$u_i(w) = \max_{0 \leq y \leq \min\{w, \sum_{j=1}^{m} c^i_j\}} \left\{\beta^i(y; m) + v^*_i(w - y)\right\}.$$

By construction, $u_i(w)$ is the maximum profit that i can make by keeping w units of her weight invested into collaborating with agents in $N \setminus S$ and working on her own; in the context of $\mathcal{G}'$, this is exactly what i can accomplish without interacting with other players in S. Thus, this construction makes each agent in S 'responsible' for maximizing its profit from its collaborations with agents in $N \setminus S$; it follows that $u^*(\mathbf{U}) = \mathcal{A}^*(CS, \mathbf{x}, S)$.

To complete the proof, we observe that $v^*_i(z)$ can be computed in time polynomial in $W_{\max}$ by Proposition 3.2, and $\beta^i(y; m)$ can be computed in time polynomial in n and $W_{\max}$ using the recurrence relation (3). Thus, the game $\mathcal{G}'$ can be constructed in time polynomial in n and $W_{\max}$; computing $u^*(\mathbf{U})$ is then easy by Theorem 3.4. □

It is not hard to verify that the proof of Theorem 4.5 goes through even if the overall communication graph is not a tree, as long as the deviating set S is an acyclic subgraph of the communication graph. This is because in 2-OCF games players interact in pairs; thus, if an agent $i \in S$ decides to withdraw resources from a coalition $\mathbf{c} \notin CS|_S$, that coalition can contain at most one other agent j, and therefore, no other agent in S is involved in $\mathbf{c}$.

5. Computing $\mathcal{A}$-stable outcomes

Having provided efficient procedures for computing optimal coalition structures and deviations in discrete 2-OCF games whose communication graphs are trees, we are ready to analyze the computational complexity of stability in this class of games. Recall that an OCF game is $\mathcal{A}$-stable if there exists some outcome $(CS, \mathbf{x})$ such that no subset of N can profitably deviate from $(CS, \mathbf{x})$ under $\mathcal{A}$, or, equivalently (Theorem 2.4), if for all $S \subseteq N$ we have

$$p^S(CS, \mathbf{x}) \geq \mathcal{A}^*(CS, \mathbf{x}, S).$$

We will now formulate the computational problem that is associated with checking the stability of a given outcome.

Name: $\mathcal{A}$-CheckCore
Input: A discrete OCF game $\mathcal{G} = \langle N, \mathbf{W}, v \rangle$ and an outcome $(CS, \mathbf{x})$ of $\mathcal{G}$.
Question: Is $(CS, \mathbf{x})$ in the $\mathcal{A}$-core of $\mathcal{G}$?

This problem is closely related to that of computing $\mathcal{A}^*$: an outcome $(CS, \mathbf{x})$ is in the $\mathcal{A}$-core if and only if the *excess* $e(CS, \mathbf{x}, S) = \mathcal{A}^*(CS, \mathbf{x}, S) - p^S(CS, \mathbf{x})$ is non-positive for all coalitions $S \subseteq N$. Thus, we need to check whether there exists a subset $S \subseteq N$ with $e(CS, \mathbf{x}, S) > 0$.

We will now present an algorithm for deciding $\mathcal{A}$-CheckCore in 2-OCF games on trees.

Theorem 5.1. *If $\mathcal{A}$ is a polynomial-time computable local arbitration function then $\mathcal{A}$-CheckCore is decidable in time polynomial in n and $W_{\max}$ whenever the input game $\mathcal{G}$ is a discrete 2-OCF game with a tree communication structure.*

Proof. Fix an outcome $(CS, \mathbf{x})$, and set $p^i = p^i(CS, \mathbf{x})$ for all $i \in N$.

Just as in the proof of Theorem 4.5, it suffices to consider deviations by connected coalitions: if $e(CS, \mathbf{x}, S) > 0$ and S is not connected, then some connected component S' of S also satisfies $e(CS, \mathbf{x}, S') > 0$. Again, we pick an arbitrary $r \in N$ as a root. We say that a connected subset $S \subseteq N$ is *rooted* at $i \in N$ if $i \in S$ and the members of S form a subtree of T^i (recall that T^i is the subtree of T rooted at i). We observe that every connected set $S \subseteq N$ is rooted at a unique $i \in N$. Given a vertex i, let E^i denote the maximum excess of a connected set rooted at i, that is,

$$E^i = \max \{e(CS, \mathbf{x}, S) \mid S \text{ is rooted at } i\}.$$

Clearly, $(CS, \mathbf{x})$ is not $\mathcal{A}$-stable if and only if $E^i > 0$ for some $i \in N$. It remains to show that each E^i can be computed in time polynomial in n and $W_{\max}$. As before, we proceed from the leaves to the root, and terminate (and report that $(CS, \mathbf{x})$ is not $\mathcal{A}$-stable) if we discover a vertex i with $E^i > 0$. If $E_i \leq 0$ for all $i \in N$, we report that $(CS, \mathbf{x})$ is $\mathcal{A}$-stable.

Given two agents $i, j \in N$, let $w^{i,j}$ denote the total weight that i assigns to interacting with j under CS, i.e.

$$w^{i,j} = \sum_{\mathbf{c}: supp(\mathbf{c}) = \{i,j\}} c^i.$$

Observe that since $\mathcal{G}$ is a 2-OCF game, there is no loss of generality in assuming that the support of every coalition that involves both i and j is exactly $\{i, j\}$. We now define two auxiliary quantities. First, given a (non-deviating) neighbor j of i and an integer w, $0 \leq w \leq w^{i,j}$, we define $\beta^{i,j}(w)$ to be the most that i can receive from $\mathcal{A}$ for her contribution to coalitions that she formed with j in $(CS, \mathbf{x})$ assuming that she keeps a total weight of w in these coalitions. Second, for each $w = 0, \ldots, W^i$, we define $D^i(w)$ to be the maximum excess of a subset rooted at i if i were to contribute w to T^i and receive nothing for whatever collaboration she has with her parent $p(i)$. In this notation,

$$E^i = \max_{0 \leq W^i - w \leq w^{i,p(i)}} \left\{ D^i(w) + \beta^{i,p(i)}(W^i - w) \right\};$$

the condition $W^i - w \leq w^{i,p(i)}$ means that we treat $p(i)$ as a non-deviator. By Theorem 4.4, $\beta^{i,p(i)}(W^i - w)$ is computable in time poly$(W_{\max})$. It remains to show how to compute $D^i(w)$ in time poly$(n, W_{\max})$ for all $i \in N$ and all w with $W^i - w^{i,p(i)} \leq w \leq W^i$.

Consider an agent i with children $C^i = \{i_1, \ldots, i_\ell\}$, and suppose that we have computed $D^{i_j}(z)$ for each $i_j \in C^i$ and each z such that $W^{i_j} - w^{i_j,i} \leq z \leq W^{i_j}$ (this encompasses the possibility that i is a leaf, as $C^i = \emptyset$ in that case). For $j = 0, \ldots, \ell$, let $T^{i,j}$ be the tree obtained from T^i by removing subtrees rooted at $i_{j+1}, \ldots, i_\ell$. Let $D^i(w; j)$ be the maximum excess of a set rooted at i that is fully contained in $T^{i,j}$, assuming that i contributes w to $T^{i,j}$ and receives nothing from her collaborations with her parent or her children $i_{j+1}, \ldots, i_\ell$; we have $D^i(w) = D^i(w; \ell)$. We will compute $D^i(w; j)$ by induction on j.

We have $D^i(w; 0) = v_i^*(w) - p^i$ for all $w = W^i - w^{i,p(i)}, \ldots, W^i$. Now, consider $j > 0$. Agent i can either include i_j in the deviating set or deviate (partially or fully) from the coalitions that she forms with i_j in $(CS, \mathbf{x})$. Thus, $D^i(w; j) = \max\{D_1, D_2\}$, where

$$D_1 = \max_{\substack{0 \leq y \leq w \\ 0 \leq z \leq W^{i_j}}} \{D^i(y; j-1) + v^*_{i,i_j}(w - y, z) + D^j(W^{i_j} - z)\}$$

and

$$D_2 = \max_{0 \leq z \leq w^{i,i_j}} \{D^i(w - z; j-1) + \beta^{i,i_j}(z)\}.$$

Since both quantities D_1 and D_2 can be computed in time polynomial in $W_{\max}$, we can efficiently compute $D^i(w; j)$, and hence also $D^i(w)$ and E^i. □

In classic coalitional games, one can use an algorithm for checking whether a given outcome is in the core in order to decide whether a given coalition structure CS can be stabilized. It turns out that one can use this approach for OCF games; however, it can only be applied if the arbitration function satisfies additional constraints.

Formally, we consider the following computational problem.

Name: $\mathcal{A}$-IsStableCS
Input: A discrete OCF game $\mathcal{G} = \langle N, \mathbf{W}, v\rangle$ and a coalition structure CS for $\mathcal{G}$.
Question: Is there an imputation $\mathbf{x}$ such that $(CS, \mathbf{x})$ in the $\mathcal{A}$-core of $\mathcal{G}$?

Consider a coalition structure $CS = (\mathbf{c}_1, \ldots, \mathbf{c}_m)$. Clearly, $\langle \mathcal{G}, CS\rangle$ is a "yes"-instance of $\mathcal{A}$-IsStableCS if and only if there exists a collection of vectors $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_m)$ that satisfies the following system of constraints.

$$\sum_{i \in supp(\mathbf{c}_j)} x_j^i = \quad v(\mathbf{c}_j) \qquad \forall j \in \{1, \ldots, m\} \tag{4}$$

$$\sum_{i \in S} \sum_{j=1}^{m} x_j^i \geq \mathcal{A}^*(CS, \mathbf{x}, S)\ \forall S \subseteq N \tag{5}$$

$$x_j^i \geq \quad 0 \qquad \forall j \in \{1, \ldots, m\}; \forall i \in N \tag{6}$$

The constraints in (4) are known as the *efficiency* constraints; together with the non-negativity constraints (6) they ensure that $\mathbf{x}$ is indeed a valid imputation. The constraints in (5) are referred to as *stability* constraints; they ensure that $(CS, \mathbf{x})$ is in the $\mathcal{A}$-core of $\mathcal{G}$. Note that the number of constraints in (5) is exponential in n.

In general, $\mathcal{A}^*(CS, \mathbf{x}, S)$ need not be a linear function of $\mathbf{x}$; however, if it is, the system (4)–(6) consists of linear constraints, and the algorithm in the proof of Theorem 5.1 can be used to obtain a polynomial-time separation oracle for it. Recall that a *separation oracle* for a linear program is an algorithm that takes a candidate solution of this linear program as an input; it then reports if this solution satisfies all constraints, and, if not, outputs a violated constraint. It is well-known that we can solve a linear program in polynomial time as long as we have a polynomial-time separation oracle for it [43]. In particular, for the conservative arbitration function $\mathcal{A}_c$ constraints (4)–(6) are linear, since $\mathcal{A}_c^*(CS, \mathbf{x}, S) = v^*(S)$ and hence the right-hand-side of each constraint does not depend on $\mathbf{x}$. Consequently, it follows from Theorem 5.1 that $\mathcal{A}_c$-IsStableCS is decidable in time polynomial in n and $W_{\max}$ whenever the input game $\mathcal{G}$ is a discrete 2-OCF game with a tree communication structure.

For the refined arbitration function $\mathcal{A}_r$ the situation is a bit more complicated. The following example shows that $\mathcal{A}_r^*(CS, \mathbf{x}, S)$ is not a linear function of $\mathbf{x}$.

Example 5.2. Consider a two-player discrete OCF game $\mathcal{G}$ with $N = \{1, 2\}$, $W^1 = 1$, $W^2 = 1$ and the characteristic function v such that $v(1, 1) = 10$, $v(1, 0) = v(0, 1) = 5$, $v(0, 0) = 0$ (i.e. $\mathcal{G}$ is equivalent to a non-overlapping game). Consider an outcome $(CS, \mathbf{x})$, where CS consists of a single coalition $(1, 1)$. Under the refined arbitration function, Player 1 would like to withdraw his contribution to this coalition if and only if $x_1^1 < 5$; thus, we have $\mathcal{A}_r^*(CS, \mathbf{x}, \{1\}) = 5$ if $x_1^1 < 5$ and $\mathcal{A}_r^*(CS, \mathbf{x}, \{1\}) = x_1^1$ if $x_1^1 \geq 5$. This behavior is clearly non-linear. Note a subtle difference between the behavior of the conservative and refined arbitration functions: under the former agent 1 is punished simply for announcing his desire to deviate, whereas under the latter the arbitration function checks if he actually withdrew any resources from existing coalitions.

However, for $\mathcal{A}_r$ we can obtain an equivalent system of constraints by replacing constraint (5) with a collection of linear constraints, thereby obtaining a linear program. We can then use Theorem 5.1 to obtain a separation oracle for this linear program. To prove this, we will now express $\mathcal{A}_r^*(CS, \mathbf{x}, S)$ as a maximum of a finite collection of linear functions of $\mathbf{x}$.

For every $CS' \subseteq CS \setminus CS|_S$, the payoff that S receives under $\mathcal{A}_r$ if it withdraws from coalitions in CS' (but not from other coalitions with non-deviators), is a linear function of $\mathbf{x}$, which we denote by $\rho_S(CS')$; we have

$$\rho_S(CS') = v^*(\mathbf{t}) + \sum_{i \in S} \sum_{\mathbf{c}_j \in CS \setminus CS'} x_j^i,$$

where $\mathbf{t}$ is the vector of resources available to S once they deviate from CS'. Then we have $\mathcal{A}_r^*(CS, \mathbf{x}, S) = \max_{CS' \subseteq CS \setminus CS|_S} \rho_S(CS')$, and constraint

$$\sum_{i \in S} \sum_{j=1}^{m} x_j^i \geq \mathcal{A}_r^*(CS, \mathbf{x}, S)$$

can be replaced by the collection of constraints

$$\sum_{i \in S} \sum_{j=1}^{m} x_j^i \geq \rho_S(CS') \text{ for each } CS' \subseteq CS \setminus CS|_S.$$

It remains to observe that the algorithm from the proof of Theorem 5.1 can be used to pinpoint which of these new constraints (if any) are violated by a given candidate solution; indeed, using standard dynamic programming techniques we can identify a specific deviation that corresponds to a positive excess.

A similar argument applies in the case of the optimistic arbitration function: in the discrete setting $\mathcal{A}_o^*(CS, \mathbf{x}, S)$ can be computed as a maximum of a finite number of linear functions of $\mathbf{x}$.

We can summarize these observations as follows.

Proposition 5.3. *For $\mathcal{A} \in \{\mathcal{A}_c, \mathcal{A}_r, \mathcal{A}_o\}$ the problem $\mathcal{A}$-IsStableCS is decidable in time polynomial in n and $W_{\max}$ whenever the input game $\mathcal{G}$ is a discrete 2-OCF game with a tree communication structure.*

Proposition 5.3 implies that we can check whether a given discrete 2-OCF game with a tree communication structure admits an $\mathcal{A}$-stable outcome for $\mathcal{A} \in \{\mathcal{A}_c, \mathcal{A}_r, \mathcal{A}_o\}$ by going over all possible coalition structures, and, for each of them, checking if this coalition structure can be stabilized; by Proposition 5.3 the latter step can be performed in time polynomial in n and $W_{\max}$. Nevertheless, this solution is likely to be impractical, since the number of possible coalition structures can be huge. We note that, unlike in non-overlapping cooperative games, it is not sufficient to identify an optimal coalition structure: recall that Example 2.6 describes a game $\mathcal{G}$ that admits two optimal coalition structures, CS and CS', such that CS' can be paired with an imputation to form an outcome in the refined core, while $(CS, \mathbf{x})$ is not in the refined core of $\mathcal{G}$ for any $\mathbf{x} \in I(CS)$ (the earliest example of this kind is due to Zick et al. [3], Example 5.17).

In fact, for $\mathcal{A} \in \{\mathcal{A}_r, \mathcal{A}_o\}$ we are not aware of any efficient algorithms for deciding whether the $\mathcal{A}$-core of an OCF game is not empty, even if the input game is a 2-OCF game with tree communication structure; nor do we have any hardness results for this problem. On the positive side, it is not hard to see that every discrete 2-OCF game with tree communication structure admits an outcome in the conservative core. Indeed, consider the *discrete superadditive cover* of any such game (as defined by Zick et al. [3]), which is a classic superadditive coalitional game on a tree. Theorem 5.3 of Zick et al. [3] says that the conservative core of an OCF game is not empty if and only if the core of its discrete superadditive cover is not empty. Our claim now follows from the celebrated result of Demange [6] that the core of a coalitional game on a tree is not empty. We also provide a direct proof that the conservative core of every discrete 2-OCF game on a tree is non-empty in the preliminary version of our paper [44].

6. Beyond tree communications

In previous sections, we have shown that for discrete 2-OCF games with tree communication structure many relevant stability-related questions can be answered in time polynomial in the number of players n, and the maximum player weight $W_{\max}$. We will now show how to extend our algorithms to 2-OCF games whose communication graphs are close to being trees, i.e., have bounded *treewidth* [45].

In what follows, we assume that we are given a 2-OCF game whose communication graph Γ is connected; if this is not the case, we can simply apply our methods to each of the connected components of Γ separately.

Given a graph $\Gamma = \langle N, E \rangle$, a *tree decomposition* of Γ is a tree $\mathcal{T}$ whose nodes are subsets of N (we write $V(\mathcal{T})$ to denote the nodes of $\mathcal{T}$ and $E(\mathcal{T})$ to denote its edges), which satisfies the following two conditions:

1. if $e \in E$ then there is some node $S \in V(\mathcal{T})$ such that $e \subseteq S$;
2. for every two nodes $S, S' \in V(\mathcal{T})$ each vertex in $S \cap S'$ appears in every node on the (unique) path between S and S'.

Given a tree decomposition $\mathcal{T}$ of a graph Γ, set

$$\text{width}(\mathcal{T}) = \max\{|S| \mid S \in V(\mathcal{T})\} - 1.$$

The *treewidth* of a graph Γ is defined as

$$\text{tw}(\Gamma) = \min\{\text{width}(\mathcal{T}) \mid \mathcal{T} \text{ is a tree decomposition of } \Gamma\}.$$

The role of -1 in the expression for width$(\mathcal{T})$ is to ensure that the treewidth of a tree is one; in fact, a graph is a tree if and only if its treewidth is one.

We say that a problem is *fixed parameter tractable* with respect to a parameter k if it can be solved in time $f(k)n^c$, where f is a computable function of k and c is a constant independent of k and n. Intuitively, this is the class of problems that admit efficient algorithms for small values of k.

Treewidth is often used as a parameter in the parameterized complexity analysis of graph-related combinatorial problems; for instance, Courcelle's theorem [46] states that any graph property that can be formulated using a fairly standard set of operators (monadic second order logic) is fixed parameter tractable with respect to the treewidth of the graph. Treewidth has also been invoked in the study of cooperative games (see Section 1.2).

We now generalize the algorithmic results derived in previous sections to 2-OCF games whose communication graphs have bounded treewidth; more specifically, we provide generalizations of Theorems 3.4, 4.5 and 5.1 to games whose communication graphs have a treewidth of k. Importantly, it is known that deciding whether the treewidth of a communication graph $\Gamma = \langle N, E\rangle$ equals k (and finding a tree decomposition of Γ of width at most k with at most $k|N|$ nodes) is fixed parameter tractable with respect to k [47].

Throughout this section, we overload notation and write $\mathrm{tw}(\mathcal{G})$ to denote the treewidth of the communication graph of a discrete 2-OCF game $\mathcal{G}$.

Theorem 6.1. *Given a 2-OCF game $\mathcal{G} = \langle N, \mathbf{W}, v\rangle$ and a coalition $\mathbf{c}$ we can compute $v^*(\mathbf{c})$ in time polynomial in n and $(W_{\max}+1)^{\mathrm{tw}(\mathcal{G})}$.*

Proof. As before, it suffices to consider the case $\mathbf{c} = \mathbf{W}$. Let $\mathcal{T}$ be a tree decomposition of the communication graph of $\mathcal{G}$ such that $\mathrm{width}(\mathcal{T}) = \mathrm{tw}(\mathcal{G})$. Let us choose some node $R \in V(\mathcal{T})$ to be the root of $\mathcal{T}$. Given a node $X \in V(\mathcal{T})$, we denote by $\mathcal{T}^X$ the subtree of $\mathcal{T}$ rooted at node X; the parent of X in $\mathcal{T}$ is denoted by $p(X)$ (with the convention that $p(R) = \emptyset$), and the set of children of X in $\mathcal{T}$ is denoted by C^X. We process the nodes from the leaves to the root; recall that the number of nodes is bounded by $n \cdot \mathrm{tw}(\mathcal{G})$.

Consider a node X. By the properties of the tree decomposition, the agents in X who appear in nodes outside of $\mathcal{T}^X$ are exactly the agents in $X \cap p(X)$. These agents need to decide how to split their resources between working with agents who appear in $V(\mathcal{T}^X)$ and working with agents who appear outside of $V(\mathcal{T}^X)$ (note that agents in $X \cap p(X)$ belong to both groups; however, we will see that this does not cause any difficulties). To capture this decision, for every vector $\mathbf{q}$ with $0 \le \mathbf{q} \le \mathbf{W}^{X\cap p(X)}$, we define $\mathrm{opt}(\mathcal{T}^X(\mathbf{q}))$ to be the value of an optimal coalition structure among the agents in $\cup V(\mathcal{T}^X)$, with the restriction that each agent $i \in X \cap p(X)$ contributes q^i units of resource to this coalition structure. Note that $v^*(\mathbf{W}) = \mathrm{opt}(\mathcal{T}^R(\mathbf{W}))$.

We will now explain how to compute $\mathrm{opt}(\mathcal{T}^X(\mathbf{q}))$ for every node X and every $\mathbf{q}$ with $0 \le \mathbf{q} \le \mathbf{W}^{X\cap p(X)}$. Order the children of X in $\mathcal{T}$ as $Y_1, \dots, Y_m$. We say that a collection of vectors $(\mathbf{z}_1, \dots, \mathbf{z}_m)$ is *valid with respect to* $\mathbf{q}$ if $0 \le \mathbf{z}_\ell \le \mathbf{W}^{Y_\ell \cap X}$ for each $Y_\ell \in C^X$, and for each $i \in X \cap p(X)$ it holds that $\sum_{\ell=1}^m z_\ell^i \le q^i$; we denote the set of all such collections by $\mathcal{Z}(X, \mathbf{q})$. Then

$$\mathrm{opt}\left(\mathcal{T}^X(\mathbf{q})\right) = \max_{\substack{(\mathbf{z}_1,\dots,\mathbf{z}_m)\in\mathcal{Z}(X,\mathbf{q}),\quad 0\le\mathbf{y}\le\mathbf{W}^X,\\ y^i \le q^i - \sum_{\ell=1}^m z_\ell^i \text{ for all } i\in X\cap p(X)}} \left\{ v^*(\mathbf{y}) + \sum_{\ell=1}^m \mathrm{opt}\left(\mathcal{T}^{Y_\ell}(\mathbf{z}_\ell)\right)\right\};$$

note that, since the collection of vectors $(\mathbf{z}_1, \dots, \mathbf{z}_m)$ is valid, the set of vectors $\mathbf{y}$ satisfying our constraints is non-empty. Observe also that, since $|supp(\mathbf{y})| \le |X| \le \mathrm{tw}(\mathcal{G})$, by Proposition 3.2, we can compute $v^*(\mathbf{y})$ in time $O((W_{\max}+1)^{2\mathrm{tw}(\mathcal{G})})$. However, the expression above does not yet lead to an efficient algorithm for computing $\mathrm{opt}(\mathcal{T}^X(\mathbf{q}))$, since there are exponentially many ways to choose a valid collection of vectors.

To deal with this issue, similarly to the proof of Theorem 3.4, we employ dynamic programming. For $j = 0, \dots, m$, let $\mathcal{T}^{X;j}$ be the tree obtained from $\mathcal{T}^X$ by removing the subtrees rooted at all but the first j children of X, and for each $\mathbf{q}$ with $0 \le \mathbf{q} \le W^{X\cap p(X)}$ let $\mathrm{opt}(\mathcal{T}^X(\mathbf{q}; j))$ be the value of the best coalition structure formed by agents in $\cup V(\mathcal{T}^{X;j})$ assuming that each agent $i \in X \cap p(X)$ contributes q^i units of resource to this coalition structure. Then $\mathcal{T}^X(\mathbf{q}; 0)$ is simply $v^*(\min\{\mathbf{W}^X, \mathbf{q}\})$ (where the minimum is taken coordinate-wise), and for $j = 1, \dots, m$ we have

$$\mathrm{opt}\left(\mathcal{T}^X(\mathbf{q}; j)\right) = \max_{\substack{0\le\mathbf{z}\le\mathbf{W}^{Y_j\cap X},\quad 0\le\mathbf{y}\le\mathbf{q},\\ y^i+z^i\le q^i \text{ for all } i\in X\cap p(X)}} \left\{\mathrm{opt}\left(\mathcal{T}^X(\mathbf{y}; j-1)\right) + \mathrm{opt}\left(\mathcal{T}^{Y_j}(\mathbf{z})\right)\right\}.$$

Note that $\mathrm{opt}(\mathcal{T}^X(\mathbf{q}; m)) = \mathrm{opt}(\mathcal{T}^X(\mathbf{q}))$. Thus, assuming we have computed $\mathrm{opt}(\mathcal{T}^Y(\mathbf{z}))$ for all $Y \in C^X$ and all $\mathbf{z}$ with $0 \le \mathbf{z} \le \mathbf{W}^{Y\cap X}$, we can compute $\mathrm{opt}(\mathcal{T}^X(\mathbf{q}))$ in time polynomial in $(W_{\max}+1)^{\mathrm{tw}(\mathcal{G})}$ and linear in $|C^X|$. Hence the total running time of our algorithm is polynomial in $(W_{\max}+1)^{\mathrm{tw}(\mathcal{G})}$ and linear in $n \cdot \mathrm{tw}(\mathcal{G})$. □

The problem of computing the maximum amount that a set can get by deviating is also fixed parameter tractable with respect to treewidth. To prove this, we use the same approach as in the proof of Theorem 4.5: in order to compute the most that a set S can get by deviating from an outcome $(CS, \mathbf{x})$, we construct an auxiliary game $\mathcal{G}' = \langle S, \mathbf{U}, u\rangle$ that satisfies $U^i = W^i$ for $i \in S$ and $u^*(\mathbf{U}) = \mathcal{A}^*(CS, \mathbf{x}, S)$ (the algorithm to construct u is given in the proof of Theorem 4.5 and does not depend on the structure of the communication graph), and apply the algorithm described in the proof of Theorem 6.1 to this game. Since $\mathrm{tw}(\mathcal{G}') \le \mathrm{tw}(\mathcal{G})$, we obtain the following theorem.

Theorem 6.2. *If $\mathcal{A}$ is a polynomial-time computable local arbitration function then $\mathcal{A}$-ArbVal is decidable in time polynomial in n and $(W_{\max}+1)^{\mathrm{tw}(\mathcal{G})}$ whenever the input game $\mathcal{G}$ is a discrete 2-OCF game.*

Finally, we provide an algorithm for deciding whether a given outcome is in the $\mathcal{A}$-core of a given 2-OCF game $\mathcal{G}$ that runs in time polynomial in n and $(W_{\max}+1)^{\mathrm{tw}(\mathcal{G})}$.

Theorem 6.3. *If $\mathcal{A}$ is a polynomial-time computable local arbitration function then $\mathcal{A}$-CheckCore is decidable in time polynomial in n and $(W_{\max}+1)^{\mathrm{tw}(\mathcal{G})}$ whenever the input game $\mathcal{G}=\langle N,\mathbf{W},v\rangle$ is a discrete 2-OCF game.*

Proof. Suppose that we are given an outcome $(CS,\mathbf{x})$ of $\mathcal{G}$. Our goal is to decide whether there exists a subset $T\subseteq N$ such that $e(CS,\mathbf{x},T)>0$; as argued in the proof of Theorem 4.5, it suffices to restrict our attention to connected subsets of N. Again, let $\mathcal{T}$ be a tree decomposition of the communication graph of $\mathcal{G}$ whose width is $\mathrm{tw}(\mathcal{G})$, and choose some $R\in V(\mathcal{T})$ to be the root of $\mathcal{T}$. For each $X\in V(\mathcal{G})$ let $\mathcal{T}^X$ be the subtree of $\mathcal{T}$ rooted at X, and let C^X denote the set of children of X. Given a connected subset $T\subseteq N$, we say that *T is rooted at* $(X;S)$, where $X\in V(\mathcal{G})$, $S\subseteq X$ if $T\subseteq V(\mathcal{T}^X)$, $T\nsubseteq V(\mathcal{T}^Y)$ for each $Y\in C^X$ (i.e., X is the deepest node such that $T\subseteq V(\mathcal{T}^X)$) and $T\cap X=S$. Observe that, since T is connected, the properties of tree decompositions imply that S is not empty. By construction, for each connected subset T of N there is a unique pair $(X;S)$ with $X\in V(\mathcal{G})$, $S\subseteq X$ such that T is rooted at $(X;S)$; this provides us with a convenient way to systematically go over all connected subsets of agents.

Given a node X, a non-empty subset $S\subset X$, a vector $\mathbf{q}$ with $0\le\mathbf{q}\le\mathbf{W}^S$, and a subset T rooted at $(X;S)$, let $\mathcal{A}^*_{\mathbf{q}}(T)$ be the maximum payoff that T can earn by deviating from $(CS,\mathbf{x})$ under $\mathcal{A}$ with the restriction that post-deviation each agent $i\in S$ only allocates q^i units of resource to productive work, and receives no payoff for the remaining $1-q^i$ units of resource. While this definition may seem counterintuitive at first (why would we want an agent to waste her resources?), we need it to explain what happens when an agent in S splits her resources between agents that appear within $\mathcal{T}^X$ and agents that appear outside of that tree. Let

$$E^{X;S}(\mathbf{q})=\max_{T\text{ is rooted at }(X;S)}\mathcal{A}^*_{\mathbf{q}}(T).$$

Observe that it suffices to decide whether $E^{X;S}(\mathbf{W}^S)>0$ for some $X\in V(\mathcal{T})$ and $\emptyset\neq S\subseteq X$, i.e., to compute at most $n\cdot\mathrm{tw}(\mathcal{G})\cdot 2^{\mathrm{tw}(\mathcal{G})}$ quantities.

Again, we process the nodes from the leaves to the root. Consider a node X and a non-empty subset $S\subseteq X$, and suppose that we have already computed $E^{Y;S'}(\mathbf{z})$ for all $Y\in C^X$, all $S'\subseteq Y$ with $S'\neq\emptyset$ and all $\mathbf{z}$ with $0\le\mathbf{z}\le\mathbf{W}^{S'}$. Let us order the nodes in C^X as $Y_1,\dots,Y_m$, and for $j=1,\dots,m$ set $S_j=S\cap Y_j$, $\overline{S}_j=Y_j\setminus(X\setminus S)$. For $j=0,\dots,m$, let $\mathcal{T}^{X;j}$ be the tree obtained from $\mathcal{T}^X$ by removing the subtrees rooted at all but the first j children of X.

It follows from the properties of tree decompositions that for every connected set T rooted at $(X;S)$ we have $S_j\subseteq T\cap Y_j\subseteq\overline{S}_j$ for each $j=1,\dots,m$ and the sets $\overline{S_j}\setminus S_j$ are pairwise disjoint. Consequently, for each $\mathbf{q}$ with $0\le\mathbf{q}\le\mathbf{W}^S$ we define

$$E^{X;S;j}(\mathbf{q})=\max_{\substack{T_\ell\text{ is rooted at }S'_\ell\\ S_\ell\subseteq S'_\ell\subseteq\overline{S}_\ell,\ell=1,\dots,j}}\mathcal{A}^*_{\mathbf{q}}(\cup_{\ell=1,\dots,j}T_\ell).$$

We then have $E^{X;S}(\mathbf{q})=E^{X;S;m}(\mathbf{q})$.

To compute $E^{X;S;0}(\mathbf{q})$, it suffices to determine what is the maximum amount that the agents in S can make when each $i\in S$ optimally withdraws her resources from existing collaborations with agents in $N\setminus S$ so as to invest them into collaborating with other agents in S, given that she has to discard W^i-q^i units of resource. The algorithm in the proof of Theorem 4.4 can be adapted to compute this quantity in time polynomial in $(W_{\max}+1)^{|S|}$: specifically, we have

$$E^{X;S;0}(\mathbf{q})=\max_{0\le\mathbf{s}+\mathbf{t}\le\mathbf{W}^S-\mathbf{q}}v^*(\mathbf{s}+\mathbf{t}-(\mathbf{W}^S-\mathbf{q}))+A^{\cap}(\mathbf{t})),$$

where $\mathbf{s}$ and $A^{\cap}(\mathbf{t})$ are defined in the proof of Theorem 4.4.

Now, consider computing $E^{X;S;j}$ for $j\ge 1$. We need to decide which of the agents in $\overline{S}_j\setminus S_j$ will join the deviating set, how much they should contribute, and how much of S's resources should be allocated to working with these agents. We make this decision by considering all available choices (the number of these choices is bounded by $2^{|Y_j|}(W_{\max}+1)^{|S|+|S'|}\le(W_{\max}+1)^{3\cdot\mathrm{tw}(\mathcal{G})}$):

$$E^{X;S;j}(\mathbf{q})=\max_{\substack{S_j\subseteq S'\subseteq\overline{S}_j,0\le\mathbf{z}\le\mathbf{W}^{S'}\\ 0\le\mathbf{y}\le\mathbf{q},y^i+z^i\le q^i\text{ for all }i\in S}}\left\{E^{X;S;j-1}(\mathbf{y})+E^{Y_j;S'}(\mathbf{z})\right\}.$$

Note that we choose S' from $\overline{S}_j$: by choosing S, we have already decided which of the agents in X are deviators, and our decision for Y_j should be consistent with this choice.

It follows that we can compute $E^{X;S}(\mathbf{q})$ in time polynomial in $|C^X|$ and $(W_{\max}+1)^{\mathrm{tw}(\mathcal{G})}$; this completes the proof. □

7. Linear bottleneck games and the optimistic core

In this section, we move away from the discrete setting considered so far, and shift our attention to the standard model of OCF games, where agents have rational weights. We describe a class of games motivated by fractional combinatorial

optimization scenarios, which we call *linear bottleneck games*, and show that every game in this class has a non-empty optimistic core, and, moreover, OptVal, ArbVal, CheckCore and IsStableCS admit polynomial-time algorithms.

We first show that for linear bottleneck games an optimal coalition structure can be found using linear programming. We then use the dual LP solution to find an imputation in the optimistic core. Our results in this section build on prior work on classic cooperative game theory, where dual solutions have been used to derive payoff divisions that guarantee core stability [19,48,8]; indeed, one interpretation of our results is that linear bottleneck games admit outcomes that are stable not just when the deviators are forced to make all-or-nothing decisions concerning their collaborations with other agents (as in the classic setting), but also when the deviators are given considerably more freedom in reallocating their resources.

We start by formally defining the class of games that will be the focus of this section.

Definition 7.1. A *Linear Bottleneck Game (LBG)* is a tuple $\mathcal{G} = (N, \omega, \mathbf{T})$, where $N = \{1, \ldots, n\}$ is a set of players, $\omega = (\omega^1, \ldots, \omega^n)$ is a list of players' *weights*, and $\mathbf{T} = (T_1, \ldots, T_m)$ is a list of *tasks*, where each task T_j is associated with a set of players $A_j \subseteq N$ who are needed to complete it, as well as a *value* $\pi_j \in \mathbb{R}_+$. We assume that $A_j \neq A_{j'}$ for $j \neq j'$, and for each $i \in N$ there is a task $T_k \in \mathbf{T}$ with $A_k = \{i\}$. The characteristic function of this game is defined as follows: given a partial coalition $\mathbf{c} \in [0, 1]^n$, we set

$$v(\mathbf{c}) = \begin{cases} \pi_j \cdot \min\limits_{i \in A_j} c^i \omega^i & \text{if } supp(\mathbf{c}) = A_j \text{ for some } j \in [m] \\ 0 & \text{otherwise.} \end{cases}$$

The term 'bottleneck' refers to the fact that the value of a coalition is determined by the agent(s) making the smallest contribution; these games are linear in the sense that if each agent increases the amount of weight she contributes to a partial coalition by a factor of α, the value of the coalition would increase by a factor of α. The assumption that $A_j \neq A_{j'}$ for $j \neq j'$ ensures that the characteristic function is well-defined; that is, each task is associated with a unique set of players that can complete it. Finally, since each player can work on her own (possibly earning a payoff of 0), all resources are used. This assumption will be useful when proving our results, since it allows us to invest unused agent resources in dummy tasks.

7.1. Examples

LBGs can be used to describe a variety of settings; an overview of their descriptive power is provided by Deng et al. [19]. For the sake of exposition, we present three examples below.

First, LBGs capture *multicommodity flow games* [7,8]. We have described these games informally in Section 1; we will now present them in more detail.

There are two types of agents in a multicommodity flow game: *suppliers* and *distributors*. An instance of the game is described by a directed graph Γ with an edge set $E(\Gamma)$ and a vertex set $V(\Gamma)$, where each edge $e \in E(\Gamma)$ is associated with a distributor and has capacity $\kappa(e) \in \mathbb{R}_+$, and a set of suppliers $\mathcal{S}$, where each supplier $i \in \mathcal{S}$ is described by a pair of nodes $(s^i, t^i) \in V(\Gamma) \times V(\Gamma)$, a demand $d^i \in \mathbb{R}_+$ and a per-unit price π^i. The commodity owned by i can be transferred from s^i to t^i via a path in Γ; thus, each task is associated with a supplier i and a set of distributors that corresponds to a simple s^i–t^i path in Γ. The contribution of a supplier to a partial coalition is the amount of her goods that she sends via the respective path; the contribution of a distributor is the capacity he allocates to the respective good. The value of a partial coalition is the minimum contribution of its members times the per-unit price of the respective good. We note that under this description the number of possible tasks may be exponential in the number of agents. However, as discussed by Markakis and Saberi [8], there exist other, more succinct, ways of describing the problem that result in the same solution, and to which our techniques can be applied. We chose to focus on the description above, as it highlights the fact that multicommodity flow games are linear bottleneck games.

Linear bottleneck games can also model network routing: again, we have a directed graph $\Gamma = (V(\Gamma), E(\Gamma))$ and a set of supplier agents $\mathcal{S}$, where each supplier i is associated with a pair of source-sink destinations $(s^i, t^i) \in V(\Gamma) \times V(\Gamma)$ and per-unit payoff π^i. However, now the distributor agents are associated with vertices, with each vertex $v \in V(\Gamma)$ having a processing power $\kappa(v)$, and each task is associated with a supplier i and a set of nodes on a simple s^i–t^i path in Γ; moreover, the demand of each supplier is unlimited, with the only constraint on the value of a partial coalition stemming from limitations on the processing power.

Finally, linear bottleneck games provide a simple model of collaborative production. Consider a bipartite graph with parts A and B, where each agent $i \in A \cup B$ is associated with a quantity $\omega^i \in \mathbb{R}_+$ and each edge $(a, b) \in A \times B$ is associated with a value $\pi^{a,b} \in \mathbb{R}_+$. Intuitively, there are two types of ingredients (say, dairy and fruit), each agent in A possesses a certain amount of the first ingredient, each agent in B possesses a certain amount of the second ingredient, and one can combine one unit of the first ingredient and one unit of the second ingredient to make a good that can be sold in the market (say, yogurt or ice cream). Moreover, for each pair of agents $(a, b) \in A \times B$ there is a price $\pi^{a,b}$ such that one unit of good produced by combining their ingredients can be sold for $\pi^{a,b}$ (in our example, this price depends on a and b, because some milk producers provide organic milk, while others do not, and fruit growers specialize in different fruits). Note that tasks here correspond to the edges of the graph.

7.2. Computing stable outcomes in LBGs

We start by presenting some simple observations on the structure of optimal coalition structures in LBGs.

Lemma 7.2. *For each linear bottleneck game $\mathcal{G} = \langle N, \omega, \mathbf{T}\rangle$, there is an optimal coalition structure CS such that*

(a) *for all $\mathbf{c} \in CS$ we have $c^i\omega^i = c^j\omega^j$ for all $i, j \in supp(\mathbf{c})$.*
(b) *$w_i(CS) = 1$ for all $i \in N$.*
(c) *For each $T_j \in \mathbf{T}$ the set of players A_j forms at most one coalition in CS.*

Proof. To see that condition (a) holds, observe that if $c^i\omega^i > c^j\omega^j$ for some partial coalition $\mathbf{c}$ and some player $i \in supp(\mathbf{c})$, we can obtain a coalition structure with the same or higher value by withdrawing $c^i\omega^i - c^j\omega^j$ units of player i's weight from $\mathbf{c}$ and reallocating them to the task that i can perform on her own. Similarly, condition (b) is satisfied because any agent can allocate unused resources to working on her own. Finally, condition (c) is satisfied as we can merge two coalitions with the same support without lowering the value of a coalition structure. □

Lemma 7.2 implies that an optimal coalition structure can be described by a list $\nu_1, \ldots, \nu_m$, indicating how much weight is allocated to each task.

We can now write a linear program that finds an optimal coalition structure for an LBG $\mathcal{G} = \langle N, \omega, \mathbf{T}\rangle$:

$$\begin{aligned} \text{max:} \quad & \sum_{j=1}^{m} \nu_j \pi_j & (7)\\ \text{s.t.} \quad & \sum_{j: i \in A_j} \nu_j \le \omega^i \quad \forall i \in N \\ & \nu_j \ge 0 \qquad \forall j \in [m] \end{aligned}$$

The dual of LP (7) is

$$\begin{aligned} \text{min:} \quad & \sum_{i=1}^{n} \omega^i \gamma^i & (8)\\ \text{s.t.} \quad & \sum_{i \in A_j} \gamma^i \ge \pi_j \quad \forall j \in [m] \\ & \gamma^i \ge 0 \qquad \forall i \in N \end{aligned}$$

Let $\widehat{\nu}_1, \ldots, \widehat{\nu}_m$ and $\widehat{\gamma}^1, \ldots, \widehat{\gamma}^n$ be optimal solutions to (7) and (8), respectively. Let CS be the coalition structure that corresponds to $\widehat{\nu}_1, \ldots, \widehat{\nu}_m$. We construct a payoff vector $\mathbf{x}$ for CS as follows: for every $j = 1, \ldots, m$ we set $x^i_j = \widehat{\gamma}^i\widehat{\nu}_j$ if $i \in A_j$, and $x^i_j = 0$ otherwise. In words, each player i has some "bargaining power" $\widehat{\gamma}^i$, and is paid for each task she works on in proportion to her bargaining power. Note that both CS and $\mathbf{x}$ can be computed efficiently from the description of the game. We will now show that $\mathbf{x}$ is an imputation for CS, and, moreover, $(CS, \mathbf{x})$ is in the optimistic core.

Theorem 7.3. *Given a linear bottleneck game $\mathcal{G} = \langle N, \omega, \mathbf{T}\rangle$, let CS and $\mathbf{x}$ be the coalition structure and the payoff vector constructed above. Then $\mathbf{x} \in I(CS)$ and $(CS, \mathbf{x})$ is in the optimistic core of $\mathcal{G}$.*

Proof. We will first argue that $\mathbf{x} \in I(CS)$. To see that $\mathbf{x}$ satisfies coalitional efficiency, note that the sum of payoffs from task T_j is

$$\sum_{i \in A_j} x^i_j = \sum_{i \in A_j} \widehat{\gamma}^i\widehat{\nu}_j = \widehat{\nu}_j \sum_{i \in A_j} \widehat{\gamma}_i.$$

As $\widehat{\gamma}^1, \ldots, \widehat{\gamma}^n$ is an optimal solution to (8), by complementary slackness we have either $\sum_{i \in A_j} \widehat{\gamma}^i = \pi_j$ or $\widehat{\nu}_j = 0$. Thus, for every task T_j with $\widehat{\nu}_j > 0$, its total payoff $\widehat{\nu}_j\pi_j$ is shared by players in A_j only.

We now show that the outcome $(CS, \mathbf{x})$ is in the optimistic core. We can assume without loss of generality that CS allocates non-zero weight to the first k tasks $T_1, \ldots, T_k$, $k \le m$, and no weight to the remaining tasks. Consider a deviation from $(CS, \mathbf{x})$ by a set S. This deviation can be described by the list of tasks that S abandons completely, and the amount of weight that players in S withdraw from all other tasks. Assume without loss of generality that the tasks that S abandons completely are $T_{\ell+1}, \ldots, T_k$ (this list includes all tasks T_j with $A_j \subseteq S$), and for each $j = 1, \ldots, \ell$ each member of $A_j \cap S$ withdraws z_j units of weight from T_j. Observe that, because of the properties of linear bottleneck games, we can restrict

ourselves to such 'uniform' deviations without loss of generality. When players in S deviate, they lose their payoff from $T_{\ell+1}, \ldots, T_k$, and their payoff from $T_1, \ldots, T_\ell$ is reduced by $\sum_{j=1}^{\ell} z_j \pi_j$.

For each $i \in S$, set

$$\mu^i = \sum_{\ell < j \leq k, i \in A_j} \widehat{\nu}_j, \qquad Z^i = \sum_{j \leq \ell, i \in A_j} z_j;$$

μ^i is the total amount of weight that i withdraws from tasks $T_{\ell+1}, \ldots, T_k$, while Z^i is the total amount of weight that i withdraws from $T_1, \ldots, T_\ell$. The profit that S obtains from optimally using the withdrawn resources is given by the following linear program:

$$\begin{aligned} \text{max:} \quad & \sum_{A_j \subseteq S} \nu_j \pi_j && (9) \\ \text{s.t.} \quad & \sum_{j: i \in A_j, A_j \subseteq S} \nu_j \leq \mu^i + Z^i \quad \forall i \in S \\ & \nu_j \geq 0 \qquad\qquad \forall A_j \subseteq S \end{aligned}$$

The dual of LP (9) is

$$\begin{aligned} \text{min:} \quad & \sum_{i \in S} \gamma^i (\mu^i + Z^i) && (10) \\ \text{s.t.} \quad & \sum_{i \in A_j} \gamma^i \geq \pi_j \quad \forall A_j \subseteq S \\ & \gamma^i \geq 0 \qquad \forall i \in S \end{aligned}$$

Let α be the value of LP (9) (and hence also of LP (10)). Note that the total profit that S gets by deviating equals $\alpha - \sum_{j=1}^{\ell} z_j \pi_j$, (recall that $\sum_{j=1}^{\ell} z_j \pi_j$ is the total marginal loss that S incurs by partially deviating from $T_1, \ldots, T_\ell$). For every optimal solution to (8), its restriction to $i \in S$ is a feasible solution to (10): the constraints in (8) are more restrictive than those in (10). Thus, we obtain

$$\alpha \leq \sum_{i \in S} \widehat{\gamma}^i (\mu^i + Z^i).$$

Now, $\sum_{i \in S} \widehat{\gamma}^i \mu^i$ is exactly the payoff that S was getting from $T_{\ell+1}, \ldots, T_k$ under $(CS, \mathbf{x})$. Further,

$$\begin{aligned} \sum_{i \in S} \widehat{\gamma}^i Z^i &= \sum_{j=1}^{\ell} \sum_{i \in S \cap A_j} \widehat{\gamma}^i z_j \\ &= \sum_{j=1}^{\ell} z_j \left(\sum_{i \in S \cap A_j} \widehat{\gamma}^i \right) \leq \sum_{j=1}^{\ell} z_j \left(\sum_{i \in A_j} \widehat{\gamma}^i \right) = \sum_{j=1}^{\ell} z_j \pi_j, \end{aligned}$$

where the last equality holds because of the complementary slackness; as previously mentioned, the latter expression is the marginal loss that S suffers for withdrawing resources from $T_1, \ldots, T_\ell$. Thus, the total payoff that S gets from deviating is at most $\sum_{i \in S} \widehat{\gamma}^i \mu^i$. Further,

$$\begin{aligned} \sum_{i \in S} \widehat{\gamma}^i \mu^i &= \sum_{i \in S} \widehat{\gamma}^i \sum_{\ell < j \leq k, i \in A_j} \widehat{\nu}_j \\ &\leq \sum_{i \in S} \widehat{\gamma}^i \sum_{j: i \in A_j} \widehat{\nu}_j \\ &= \sum_{i \in S} p^i(CS, \mathbf{x}) = p^S(CS, \mathbf{x}). \end{aligned}$$

To conclude, the total payoff that S receives by deviating under the optimistic arbitration function does not exceed its payoff in $(CS, \mathbf{x})$. As this holds for every deviation and every $S \subseteq N$, it follows that $(CS, \mathbf{x})$ is in the optimistic core. □

An optimal solution of a linear program and its dual can be found in polynomial time. Hence, we obtain the following immediate corollary.

Corollary 7.4. *For linear bottleneck games, the problems* $\mathcal{A}_o$-OptVal, $\mathcal{A}_o$-ArbVal, *and* $\mathcal{A}_o$-CheckCore *can be decided in polynomial time.*

Note also that, since the refined core, the sensitive core and the conservative core all contain the optimistic core, we can conclude that linear bottleneck games admit stable solutions with respect to the conservative, sensitive and refined arbitration functions.

8. Conclusions

We have explored two methods of designing efficient algorithms for finding optimal coalition structures and stable outcomes in OCF games. The first method is to consider a discretized version of OCF games and impose constraints on agent communication; the second method is to place no constraints on agent communication, but focus on a specific family of OCF games. Our results for discrete OCF games demonstrate that when the communication network is treelike, many stability-related computational problems become easier. However, this constraint alone is insufficient for tractability, which can be seen as evidence that discrete OCF games form a rich and complex class of games. For Linear Bottleneck Games, the OCF model, and, in particular, the notion of the optimistic core, enable us to formalize the intuition that such games admit outcomes that are strongly resistant to deviation.

Our positive results for discrete OCF games rely on the notion of a local arbitration function, which essentially requires agents to be myopic in their behavior. In particular, none of our efficient algorithms can be used to find an outcome in the sensitive core. It would be interesting to identify a class of games where outcomes in the sensitive core can be computed efficiently, as well as to computationally separate the refined core and the sensitive core.

Acknowledgements

This research was supported by National Research Foundation Singapore Research Fellowship 2009-08 (Elkind), European Research Council Starting Grant ACCORD under Grant Agreement 639945 (Elkind) and a SINGA A*STAR scholarship (Zick).

References

[1] G. Chalkiadakis, E. Elkind, E. Markakis, M. Polukarov, N. Jennings, Cooperative games with overlapping coalitions, J. Artif. Intell. Res. 39 (2010) 179–216.
[2] B. Peleg, P. Sudhölter, Introduction to the Theory of Cooperative Games, second edn., Theory and Decision Library, Series C: Game Theory, Mathematical Programming and Operations Research, vol. 34, Springer, Berlin, 2007.
[3] Y. Zick, E. Markakis, E. Elkind, Arbitration and stability in cooperative games with overlapping coalitions, J. Artif. Intell. Res. 50 (2014) 847–884.
[4] G. Chalkiadakis, E. Elkind, M. Wooldridge, Computational Aspects of Cooperative Game Theory, Morgan and Claypool, 2011.
[5] R.B. Myerson, Graphs and cooperation in games, Math. Oper. Res. 2 (3) (1977) 225–229.
[6] G. Demange, On group stability in hierarchies and networks, J. Polit. Econ. 112 (4) (2004) 754–778.
[7] V. Vazirani, Approximation Algorithms, Springer Verlag, 2001.
[8] E. Markakis, A. Saberi, On the core of the multicommodity flow game, Decis. Support Syst. 39 (1) (2005) 3–10.
[9] T. Sandholm, K. Larson, M. Andersson, O. Shehory, F. Tohmé, Coalition structure generation with worst case guarantees, Artif. Intell. 111 (1) (1999) 209–238.
[10] K. Larson, T. Sandholm, Anytime coalition structure generation: an average case study, J. Exp. Theor. Artif. Intell. 12 (1) (2000) 23–42.
[11] E. Elkind, T. Rahwan, N.R. Jennings, Computational coalition formation, in: G. Weiss (Ed.), Multiagent Systems, 2nd edn., MIT Press, 2013, pp. 329–380.
[12] T. Rahwan, T.P. Michalak, M. Wooldridge, N.R. Jennings, Coalition structure generation: a survey, Artif. Intell. 229 (2015) 139–174.
[13] O. Shehory, S. Kraus, Formation of overlapping coalitions for precedence-ordered task-execution among autonomous agents, in: Proceedings of the 2nd International Conference on Multi-Agent Systems, ICMAS-96, 1996, pp. 330–337.
[14] C.F. Lin, S.L. Hu, Multi-task overlapping coalition parallel formation algorithm, in: Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS-07, 2007, p. 211.
[15] G. Zhang, J. Jiang, Z. Su, M. Qi, H. Fang, Searching for overlapping coalitions in multiple virtual organizations, Inf. Sci. 180 (2010) 3140–3156.
[16] I. Mann, L. Shapley, Values of Large Games IV: Evaluating the Electoral College by Montecarlo Techniques, Tech. Rep., The RAND Corporation, 1960.
[17] I. Mann, L.S. Shapley, Values of Large Games VI: Evaluating the Electoral College Exactly, Tech. Rep., The RAND Corporation, 1962.
[18] X. Deng, C. Papadimitriou, On the complexity of cooperative solution concepts, Math. Oper. Res. 19 (2) (1994) 257–266.
[19] X. Deng, T. Ibaraki, H. Nagamochi, Algorithmic aspects of the core of combinatorial optimization games, Math. Oper. Res. 24 (3) (1999) 751–766.
[20] S. Ieong, Y. Shoham, Marginal contribution nets: a compact representation scheme for coalitional games, in: Proceedings of the 6th ACM conference on electronic commerce, EC-05, ACM, 2005, pp. 193–202.
[21] T. Matsui, Y. Matsui, A survey of algorithms for calculating power indices of weighted majority games, J. Oper. Res. Soc. Jpn. 43 (2000) 71–86.
[22] E. Elkind, L. Goldberg, P. Goldberg, M. Wooldridge, On the computational complexity of weighted voting games, Ann. Math. Artif. Intell. 56 (2) (2009) 109–131.
[23] G. Greco, E. Malizia, L. Palopoli, F. Scarcello, On the complexity of core, kernel, and bargaining set, Artif. Intell. 175 (12–13) (2011) 1877–1910.
[24] Z. Zhang, L. Song, Z. Han, W. Saad, Z. Lu, Overlapping coalition formation games for cooperative interference management in small cell networks, in: Wireless Communications and Networking Conference, WCNC-13, IEEE, 2013, pp. 643–648.
[25] M. Ackerman, S. Brânzei, The authorship dilemma: alphabetical or contribution? in: Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems, AAMAS-14, 2014, pp. 1487–1488.
[26] R. Brafman, C. Domshlak, Y. Engel, M. Tennenholtz, Transferable utility planning games, in: Proceedings of the 24th AAAI Conference on Artificial Intelligence, AAAI-10, 2010, pp. 709–714.
[27] G. Chalkiadakis, G. Greco, E. Markakis, Characteristic function games with restricted agent interactions: core-stability and coalition structures, Artif. Intell. 232 (2016) 76–113.
[28] Y. Bachrach, E. Elkind, R. Meir, D. Pasechnik, M. Zuckerman, J. Rothe, J. Rosenschein, The cost of stability in coalitional games, in: Proceedings of the 2nd International Symposium on Algorithmic Game Theory, SAGT-09, 2009, pp. 122–134.

[29] R. Meir, Y. Zick, E. Elkind, J.S. Rosenschein, Bounding the cost of stability in games over interaction networks, in: Proceedings of the 27th AAAI Conference on Artificial Intelligence, AAAI-13, 2013, pp. 690–696.
[30] N. Bousquet, Z. Li, A. Vetta, Coalition games on interaction graphs: a horticultural perspective, in: Proceedings of the 16th ACM Conference on Economics and Computation, EC-15, 2015, pp. 95–112.
[31] A. Igarashi, E. Elkind, Hedonic games with graph-restricted communication, in: Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems, AAMAS-16, 2016, pp. 242–250.
[32] A. Igarashi, D. Peters, E. Elkind, Group activity selection on social networks, in: Proceedings of the 31st AAAI Conference on Artificial Intelligence, AAAI-17, 2017, pp. 565–571.
[33] A. Igarashi, R. Bredereck, E. Elkind, On parameterized complexity of group activity selection problems on social networks, in: Proceedings of the 16th Conference on Autonomous Agents and Multiagent Systems, AAMAS-17, 2017, pp. 1575–1577.
[34] S. Gupta, S. Roy, S. Saurabh, M. Zehavi, Group activity selection on graphs: parameterized analysis, in: Proceedings of the 10th International Symposium on Algorithmic Game Theory, SAGT-17, 2017, pp. 106–118.
[35] R. Aumann, J. Drèze, Cooperative games with coalition structures, Int. J. Game Theory 3 (1974) 217–237.
[36] T. Shrot, Y. Aumann, S. Kraus, On agent types in coalition formation problems, in: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems, AAMAS-10, 2010, pp. 757–764.
[37] K. Aadithya, T. Michalak, N. Jennings, Representation of Coalitional Games with Algebraic Decision Diagrams, Tech. Rep. UCB/EECS-2011-8, UC Berkeley, 2011.
[38] S. Ueda, M. Kitaki, A. Iwasaki, M. Yokoo, Concise characteristic function representations in coalitional games based on agent types, in: Proceedings of the 22nd International Joint Conference on Artificial Intelligence, IJCAI-11, 2011, pp. 393–399.
[39] G. Greco, E. Malizia, F. Scarcello, L. Palopoli, Hard and easy k-typed compact coalitional games: the knowledge of player types marks the boundary, in: Proceedings of the 20th European Conference on Artificial Intelligence, ECAI-12, 2012, pp. 372–377.
[40] M.R. Garey, D.S. Johnson, Computers and Intractibility, W.H. Freeman and Company, 1979.
[41] O. Shehory, S. Kraus, Task allocation via coalition formation among autonomous agents, in: Proceedings of the 14th International Joint Conference on Artificial Intelligence, IJCAI-95, 1995, pp. 655–661.
[42] J. Edmonds, Paths, trees, and flowers, Can. J. Math. 17 (1965) 449–467.
[43] A. Schrijver, Theory of Linear and Integer Programming, Wiley, Chichester, 1986.
[44] Y. Zick, G. Chalkiadakis, E. Elkind, Overlapping coalition formation games: charting the tractability frontier, in: Proceedings of the 11th International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS-12, 2012, pp. 787–794.
[45] N. Robertson, P. Seymour, Graph minors, III: planar tree-width, J. Comb. Theory, Ser. A 36 (1) (1984) 49–64.
[46] B. Courcelle, The monadic second-order logic of graphs, I: recognizable sets of finite graphs, Inf. Comput. 85 (1990) 12–75.
[47] H.L. Bodlaender, A linear-time algorithm for finding tree-decompositions of small treewidth, SIAM J. Sci. Comput. 25 (6) (1996) 1305–1317.
[48] K. Jain, M. Mahdian, Cost sharing, in: N. Nisan, T. Rougarden, E. Tardas, V. Vazirani (Eds.), Algorithmic Game Theory, Cambridge University Press, 2007, pp. 383–408, Chap. 15.