Exact Algorithms and Lower Bounds for Forming Coalitions of Constrained Maximum Size

Foivos Fioravantes[1], **Harmender Gahlawat**[2], **Nikolaos Melissinos**[1]

[1] Department of Theoretical Computer Science, Faculty of Information Technology, Czech Technical University in Prague, Prague, Czech Republic

[2]Combinatorial Optimization group of G-SCOP, Grenoble-INP

foivos.fioravantes@fit.cvut.cz, harmender.gahlawat@grenoble-inp.fr, nikolaos.melissinos@fit.cvut.cz

Abstract

Imagine we want to split a group of agents into teams in the most *efficient* way, considering that each agent has their own preferences about their teammates. This scenario is modeled by the extensively studied COALITION FORMATION problem. Here, we study a version of this problem where each team must additionally be of bounded size. We conduct a systematic algorithmic study, providing several intractability results as well as multiple exact algorithms that scale well as the input grows (FPT), which could prove useful in practice.

Our main contribution is an algorithm that deals efficiently with tree-like structures (bounded *treewidth*) for "small" teams. We complement this result by proving that our algorithm is asymptotically optimal. Particularly, there can be no algorithm that vastly outperforms the one we present, under reasonable theoretical assumptions, even when considering star-like structures (bounded *vertex cover number*).

Introduction

Coalition Formation is a central topic in Computational Social Choice and economic game theory (Brandt et al. 2016). The goal is to partition a set of agents into coalitions *optimizing* a *utility* function. One well-studied notion in Coalition Formation is *Hedonic Games* (Dreze and Greenberg 1980), where the utility of an agent depends only on the coalition it is placed in. Due to their general nature, capturing numerous scenarios, hedonic games are intensively studied (Aziz et al. 2019; Barrot et al. 2019; Boehmer and Elkind 2020; Brandt, Bullinger, and Wilczynski 2023; Bullinger and Kober 2021; Fanelli et al. 2021; Igarashi et al. 2019; Ohta et al. 2017; Sliwinski and Zick 2017), and have applications in social network analysis (Olsen 2009), scheduling group activities (Darmann et al. 2018), and allocating tasks to wireless agents (Saad et al. 2010).

Most problems concerning the computational complexity of hedonic games are hard (Peters and Elkind 2015). In fact, even encoding the preferences of agents generally takes exponential space, motivating the study of succinct representations for these preferences. One well-studied such class of games is Additive Separable Hedonic Games (ASHGs for short) (Bogomolnaia and Jackson 2002), where the agents are represented by the vertices of a weighted graph and the weight of each edge represents the *utility* of the agents joined by the edge for each other (see also Weighted Graphical Games model (Deng and Papadimitriou 1994)). Variants where the agent preferences are asymmetric are modeled using directed graphs. Here, the utility of an agent for a group of agents is *additive* in nature. ASHGs are well-studied in the literature (Aloisio, Flammini, and Vinci 2020; Aziz, Brandt, and Seedig 2013; Barrot and Yokoo 2019).

Most literature in the ASHGs considers the agents to be *selfish* in nature. Hence, the metric used to measure efficiency is that of *stability* (Peters and Elkind 2015), including *core stability*, *Nash Stability*, *individual stability*, etc. Semi-altruistic approaches where the agents are concerned about their *relative*'s utility along with theirs are also studied (Monaco, Moscardelli, and Velaj 2021). A standard altruistic approach in computational social choice is that of *utilitarian social welfare*, where the goal is to maximize the total sum of utility of all the agents.

Observe that if all edge weights are positive, then the maximum utilitarian utility is achieved by putting all agents in the same coalition. But there are many practical scenarios, e.g., forming office teams to allocate several projects or allocating cars/buses to people for a trip, where we additionally require that each coalition should be of a bounded size. Coalition formations with constrained coalition size have recently been a focus of attention in ASHGs (Levinger, Azaria, and Hazon 2023) and in Fractional Hedonic Games (Monaco and Moscardelli 2023). Further, coalition formations where each coalition needs to be of a fixed size have also been studied (Bilò, Monaco, and Moscardelli 2022; Cseh, Fleiner, and Harján 2019).

We consider the ASHGs with an additional constraint on the maximum allowed size of a coalition (denoted by $\mathcal{C}$), with the goal to maximize the total sum of utility of all the agents. We formally define the problem, along with other preliminaries, in the Preliminaries section. This game is known to be NP-hard even when $\mathcal{C} = 3$ (Levinger, Azaria, and Hazon 2023). Note also that this reduction is straightforward from the PARTITION INTO TRIANGLES, which is NP-hard even for graphs with $\Delta \leq 4$ (van Rooij, van Kooten Niekerk, and Bodlaender 2013).

We present a comprehensive analysis of the *parameterized complexity* of this problem, considering various structural parameters of the input graph. In parameterized com-

plexity, the goal is to restrict the exponential blow-up of running time to some *parameter* of the input (which is usually much smaller than the input size) rather than the whole input size. Due to its practical efficiency, this paradigm has been used extensively to study problems arising from Computational Social Choice and Artificial Intelligence (Bäckström et al. 2012; Bessiere et al. 2008; Bredereck et al. 2017) (including hedonic games (Ganian et al. 2023; Hanaka and Lampis 2022; Peters 2016b,a; Chen et al. 2023; Li 2021; Hanaka, Ikeyama, and Ono 2023; Hanaka et al. 2019)).

It is worth mentioning that the problem we consider has been studied from an approximation perspective and is shown to have applications in Path Transversals (Lee 2017). Moreover, (Bachrach et al. 2013) considered a Weighted Graphical Game to maximize social welfare and provided constant-factor approximation for restricted families of graphs. Finally, (Flammini et al. 2018) considered the online version of several Weighted Graphical Games (aiming to maximize utilitarian social welfare), in one of which they also consider coalitions of bounded size.

Our Contribution

In this paper we study the parameterized complexity of $\mathcal{C}$-COALITION FORMATION problem, which is a version of the COALITION FORMATION problem with the added constraint that each coalition should be of size at most $\mathcal{C}$. We consider two distinct variants of this problem according to the possibilities for the utilities of the agents. In the *unweighted* version, the utilities of all the pairs of agents are either 0 (there is no edges connecting them) or 1. In the *weighted* version, the utilities of all pairs of agents are given by natural numbers. We will refer to the former as $\mathcal{C}$-CF and the latter as $\mathcal{C}$-CFw , respectively. In both cases, the underlying structure is assumed to be an undirected graph.

We begin by noting an interesting connection to the notion of *Nash-stability*. Consider a solution $\mathcal{P} = \{C_1, ..., C_l\}$. Roughly speaking, $\mathcal{P}$ is Nash-stable if any agent does not have any additional gain by leaving its coalition and joining another that can still accommodate it. In our setting, a solution is Nash-stable if for each agent u, if $u \in C_i$, then for each $j \in [l]$ such that $|C_j| < l$, $\sum_{x \in N_{C_i}} w(ux) \geq \sum_{y \in N_{C_i}} w(uy)$. Observe that an optimal solution for $\mathcal{C}$-CF (similarly, $\mathcal{C}$-CFw) is also *Nash-stable*. This is trivially true because if this condition is not satisfied for some $u \in C_i$, then we can move u to C_j while respecting the maximum size constraint and increase the valuation. Moreover, the notion of utilitarian social welfare captures the notion of Nash stability when the coalitions are required to be of bounded size and the valuations are symmetric. Hence, our positive results stand even when the goal is to obtain a Nash-stable coalition.

It follows from our previous discussion on the hardness of the $\mathcal{C}$-CF problem that it is para-NP-hard parameterized by $\mathcal{C} + \Delta$. Thus, we consider other structural parameters of the input graph. We initiate our study with *treewidth*, denoted by tw, which measures how tree-like a graph is. Treewidth is a natural parameter of choice when considering problems admitting a graph structure, specifically in problems related to AI, because many real world networks exhibit bounded treewidth (Maniu, Senellart, and Jog 2019) and tree decompositions of "almost optimal" treewidth are easy to compute in FPT time (Korhonen and Lokshtanov 2023). We begin with showing that $\mathcal{C}$-CFw is FPT when parameterized by $tw+\mathcal{C}$ by application of a bottom-up dynamic programming.

Theorem 1. *$\mathcal{C}$-CFw can be solved in time* $(\text{tw} \cdot \mathcal{C})^{\mathcal{O}(\text{tw})} n^{\mathcal{O}(1)}$.

The complexity in the above algorithm has an exponential dependency on $\mathcal{C}$. It is natural to wonder whether there can be an efficient algorithm that avoids this, i.e., if $\mathcal{C}$-CF is FPT parameterized by tw. We answer this question negatively in the following theorem by establishing that $\mathcal{C}$-CF is W[1]-hard even for treedepth, a more restrictive parameter.

Theorem 2. *$\mathcal{C}$-CF is W[1]-hard parameterized by the treedepth of the input graph.*

Nevertheless, we do achieve such an algorithm (without exponential dependence on $\mathcal{C}$) by considering a slightly more restrictive parameter, *vertex cover number*, denoted by vc, which measures how star-like the input graph is.

Theorem 3. *$\mathcal{C}$-CFw can be solved in time* $\text{vc}^{\mathcal{O}(\text{vc})} n^{\mathcal{O}(1)}$.

Next, we prove that both of the above algorithms are asymptotically optimal, *i.e.*, we do not expect a drastic improvement in their running times.

Theorem 4. *There is no algorithm that solves $\mathcal{C}$-CF in time* $(\mathcal{C} \cdot \text{vc})^{o(\text{vc}+\mathcal{C})} n^{\mathcal{O}(1)}$*, unless the ETH fails.*

We then slightly shift our approach and attack this problem using the toolkit of *kernelization*. Intuitively, our goal is to "peel off" the useless parts of the input (in polynomial time) and solve the problem for the "small" part of the input remaining, known as the *kernel*. Due to its profound impact, kernelization was termed "the lost continent of polynomial time" (Fellows 2006). It is very useful in practical applications as it has shown tremendous speedups in practice (Gao 2009; Guo and Niedermeier 2007).

Theorem 5. *$\mathcal{C}$-CF admits a kernel with* $\mathcal{O}(\text{vc}^2 \cdot \mathcal{C})$ *vertices.*

We complement the above result by proving that, unfortunately, there can be no such kernel for the weighted version, even if we consider $\text{vc} + \mathcal{C}$ as a parameter.

Theorem 6. *$\mathcal{C}$-CFw parameterized by* $\text{vc}+\mathcal{C}$ *does not admit a polynomial kernel, unless polynomial hierarchy collapses.*

We close our study by considering additional structural parameters for the unweighted case. We provide the formal definition of these parameters in the Preliminaries Section.

Theorem 7. *$\mathcal{C}$-CF is FPT parameterized by the vertex integrity of the input graph.*

Theorem 8. *$\mathcal{C}$-CF is W[1]-hard when parameterized by the twin-cover number of G.*

Choosing to focus on these two parameters is not arbitrary. Let G be a graph with vertex integrity vi, twin-cover number twc, vertex cover number vc and clique number ω. Then, $\text{vi} \leq \text{twc} + \omega$ and $\text{twc} \leq \text{twc} + \omega$. Finally, $\text{twc}+\omega \leq f(\text{vc})$, for some computable function f. Thus, our Theorems 7 and 8 provide a clear dichotomy of the tractability of $\mathcal{C}$-CF when considering these parameters.

Preliminaries

We follow standard graph-theoretic notation (Diestel 2012). For $\ell \in \mathbb{N}$, let $[\ell] = \{1, \ldots, \ell\}$. Formally, the input of the $\mathcal{C}$-CFw (resp., $\mathcal{C}$-CF) consists of a graph $G = (V, E)$ and an edge-weight function $w : E \to \mathbb{N}$ (resp., $w : E \to \{1\}$). Additionally, we are given a *capacity* $\mathcal{C} \in \mathbb{N}$ as part of the input. Our goal is to find a *$\mathcal{C}$-partition* of V, *i.e.*, a partition $\mathcal{P} = \{C_1, \ldots, C_p\}$ of V such that $|C_i| \leq \mathcal{C}$ for each $i \in [p]$. Let $E(\mathcal{P})$ be the set of edges of the partition $\mathcal{P}$, *i.e.*, $E(\mathcal{P}) = \bigcup_{i=1}^{p} E(G[C_i])$. The *value* of a $\mathcal{C}$-partition $\mathcal{P}$ is: $v(\mathcal{P}) = \sum_{i=1}^{p} \sum_{e \in E(C_i)} w(e)$. We are interested in computing an *optimal* $\mathcal{C}$-partition, *i.e.*, a $\mathcal{C}$-partition of maximum value. Accordingly, we have a decision version of $\mathcal{C}$-CFw and $\mathcal{C}$-CF where, given $t \in \mathbb{N}$, the goal is to decide whether there exists a $\mathcal{C}$-partition with value at least t.

Parameterized Complexity. *Parameterized complexity* is a computational paradigm that extends classical measures of time complexity. The goal is to examine the computational complexity of problems with respect to an additional measure, referred to as the parameter. Formally, a parameterized problem is a set of instances $(x, k) \in \Sigma^* \times \mathbb{N}$, where k is called the *parameter* of the instance. A parameterized problem is *Fixed-Parameter Tractable* (FPT) if it can be solved in $f(k)|x|^{\mathcal{O}(1)}$ time for an arbitrary computable function $f : \mathbb{N} \to \mathbb{N}$. According to standard complexity-theoretic assumptions, a problem is not in FPT if it is shown to be W[1]-hard. This is achieved through a *parameterized reduction* from another W[1]-hard problem, a reduction, achieved in FPT time, that also guarantees that the size of the considered parameter is preserved.

A *kernelization algorithm* is a polynomial-time algorithm that takes as input an instance (I, k) of a problem and outputs an *equivalent instance* (I', k') of the same problem such that the size of (I', k') is bounded by some computable function $f(k)$. The problem is said to admit an $f(k)$ sized kernel, and if $f(k)$ is polynomial, then the problem is said to admit a polynomial kernel. It is known that a problem is FPT if and only if it admits a kernel.

Finally, the *lower bounds* we present are based on the so-called EXPONENTIAL TIME HYPOTHESIS (ETH for short) (Impagliazzo and Paturi 2001), a weaker version of which states that 3-SAT cannot be solved in time $2^{o(n+m)}$, for n and m being the number of variables and clauses of the input formula, respectively.

We refer the interested reader to classical monographs (Cygan et al. 2015; Niedermeier 2006; Flum and Grohe 2006; Downey and Fellows 2013; Fomin et al. 2019) for a more comprehensive introduction to this topic.

Structural Parameters. Let $G = (V, E)$ be a graph. A set $U \subseteq V$ is a *vertex cover* of G if for every edge $e \in E$ it holds that $U \cap e \neq \emptyset$. The *vertex cover number* of G, denoted $\mathrm{vc}(G)$, is the minimum size of a vertex cover of G.

A *tree-decomposition* of G is a pair $(T, \mathcal{B})$, where T is a tree, $\mathcal{B}$ is a family of sets assigning to each node t of T its *bag* $B_t \subseteq V$, and the following conditions hold: for every edge $uv \in E(G)$, there is a node $t \in V(T)$ such that $u, v \in B_t$, and for every vertex $v \in V$, the set of nodes t with $v \in B_t$ induces a connected subtree of T.

The *width* of a tree-decomposition $(T, \mathcal{B})$ is $\max_{t \in V(T)} |B_t| - 1$, and the treewidth $\mathrm{tw}(G)$ of a graph G is the minimum width of a tree-decomposition of G. It is well known that computing a tree-decomposition of minimum width is fixed-parameter tractable when parameterized by the treewidth (Kloks 1994; Bodlaender 1996), and even more efficient algorithms exist for obtaining near-optimal tree-decompositions (Korhonen and Lokshtanov 2023).

A tree-decomposition $(T, \mathcal{B})$ is *nice* if every node $t \in V(T)$ is exactly of one of the following four types: (i) Leaf: t is a leaf of T and $|B_t| = 0$. (ii) Introduce: t has a unique child c and there exists $v \in V$ such that $B_t = B_c \cup \{v\}$. (iii) Forget: t has a unique child c and there exists $v \in V$ such that $B_c = B_t \cup \{v\}$. (iv) Join: t has exactly two children c_1, c_2 and $B_t = B_{c_1} = B_{c_2}$. Every graph G admits a nice tree-decomposition of width $\mathrm{tw}(G)$ (Bodlaender 1998).

The *tree-depth* of G can be defined recursively: if $|V| = 1$ then G has tree-depth 1. Then, G has tree-depth k if there exists a vertex $v \in V$ such that every connected component of $G[V \setminus \{v\}]$ has tree-depth at most $k - 1$. The graph G has *vertex integrity* k if there exists a set $U \subseteq V$ such that $|U| = k' \leq k$ and all connected components of $G[V \setminus U]$ are of order at most $k - k'$. We can find such a set in FPT-time parameterized by k (Drange, Dregi, and van 't Hof 2016). A set S is a *twin-cover* (Ganian 2011) of G if V can be partitioned into the sets $S, V_1, \ldots, V_p$, such that for every $i \in [p]$, all the vertices of V_i are twins. The size of a minimum twin-cover of G is the *twin-cover number* of G.

Let A and B be two parameters of the same graph. We will write $A \leq_f B$ to denote that the parameter A is upperly bounded by a function of parameter B. Let G be a graph with treewidth tw, vertex cover number vc, tree-depth td, twin-cover number twc and vertex integrity vi. We have that that twc $\leq_f$ vc. Moreover, tw $\leq_f$ td $\leq_f$ vi $\leq_f$ vc, but twc is incomparable to tw.

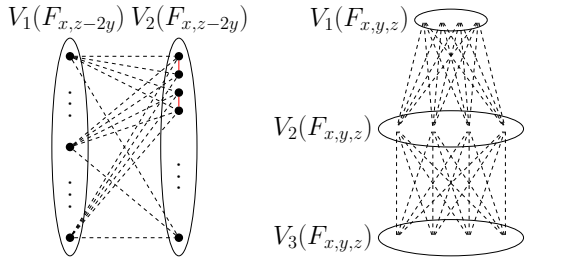
Trees Versus Stars

Before we proceed to the main theorem of this section, allow us to briefly comment upon the MAX UTILITARIAN ASHG (MU for short). Simply put, MU is a restriction of $\mathcal{C}$-CFw where $\mathcal{C} = n$ (i.e., the size of the coalitions is unbounded). The authors of (Hanaka et al. 2019) provide an FPT algorithm for MU parameterized by the treewidth of the input graph. We stress however that, as MU is a special case of $\mathcal{C}$-CFw, the above FPT algorithm cannot be used to deal with our problem in the general case.

Theorem 1. *$\mathcal{C}$-CFw can be solved in time* $(\mathrm{tw} \cdot \mathcal{C})^{\mathcal{O}(\mathrm{tw})} n^{\mathcal{O}(1)}$.

Sketch of proof. We perform dynamic programming on the nodes of a *nice tree decomposition* $\mathcal{T}$ of the graph G rooted at a node r. Such a decomposition can be computes as discussed in the Preliminaries. For a node t of $\mathcal{T}$, we denote by B_t the bag of this node and by $B_t^{\downarrow}$ the set of vertices of the graph that appears in the bags of the nodes of the subtree with t as a root. Observe that $B_t \subseteq B_t^{\downarrow}$.

For all nodes t of $\mathcal{T}$ we create all the $\mathcal{C}$-partitions of $G[B_t^{\downarrow}]$ that are needed in order to find an optimal $\mathcal{C}$-partition; this is



(α) the gadget $F_{x,z-2y}$ (β) the gadget $F_{x,y,z}$

Figure 1: The gadgets used in the proof of Theorem 2

achieved by storing only $(\text{tw} \cdot \mathcal{C})^{\mathcal{O}(\text{tw})}$ $\mathcal{C}$-partitions for each bag. We first define types of $\mathcal{C}$-partitions of $G[B_t^{\downarrow}]$ based on their intersection with B_t and the size of their sets. In particular, we define a *coloring function* $Col : B_t \to [tw+1]$ and let S be a table of size $|tw+1|$ such that $0 \leq S[i] \leq \mathcal{C}$ for all $i \in [\text{tw}+1]$. We will say that a $\mathcal{C}$-partition $\mathcal{P} = \{C_1, \ldots, C_p\}$ is of *type* $(Col, S)_t$ if: (i) $\mathcal{P}$ is a $\mathcal{C}$-partition of $G[B_t^{\downarrow}]$, (ii) for any $i \leq \text{tw}+1$ and $u \in B_t$, $Col(u) = i$ if and only if $u \in C_i \cap B_t$, and (iii) $S[i] = |C_i|$ for all $i \in [tw+1]$.

Intuitively, for any $\mathcal{C}$-partition $\mathcal{P}$ of type $(Col, S)_t$, the function Col describes the way that $\mathcal{P}$ partitions the set B_t. Also, the table S gives us the sizes of the sets of $\mathcal{P}$ that intersect with B_t. Finally, for any node t, a $\mathcal{C}$-partition of type $(Col, S)_t$ is *important* if it has a value greater or equal to the value of any other $\mathcal{C}$-partition of the same type. Notice that any optimal $\mathcal{C}$-partition of the given graph is also an important $\mathcal{C}$-partition of r. Therefore, to compute an optimal $\mathcal{C}$-partition of G, it suffices to find an important $\mathcal{C}$-partition of maximum value among all important $\mathcal{C}$-partitions of r.

We now present the information we keep for each node. Let t be a node of $\mathcal{T}$, $Col : B_t \to [tw+1]$ be a function and S be a table of size $|tw+1|$ such that $0 \leq S[i] \leq \mathcal{C}$ for all $i \in [\text{tw}+1]$. For each important $\mathcal{C}$-partition of type $(Col, S)_t$, we store a tuple $(Col, S, W, \mathcal{P})$ for t, where $\mathcal{P}$ is an important $\mathcal{C}$-partition of type $(Col, S)_t$ and W is its value. Observe that W is the value of a partition of the whole subgraph induced by the vertices belonging to $B_t^{\downarrow}$.

We now explain how to deal with each kind of node of $\mathcal{T}$.

Leaf Nodes. Since the leaf nodes contain no vertices, we do not need to keep any non-trivial coloring. Also, all the positions of the tables S are set to 0. We keep a $\mathcal{C}$-partition $\mathcal{P} = \{C_1, \ldots, C_{\text{tw}+1}\}$ where $C_i = \emptyset$ for all $i \in [\text{tw}+1]$.

Introduce Nodes. Let t be an introduce node with c being its child node and u be the newly introduced vertex. For each tuple $(Col, S, W, \mathcal{P})$ of c, we create at most tw $+ 1$ tuples for t. For each color $i \in [\text{tw}+1]$ we consider two cases: either $0 \leq S[i] < \mathcal{C}$ or $S[i] = \mathcal{C}$. If $0 \leq S[i] < \mathcal{C}$, then we set $Col(u) = i$, increase $S[i]$ by one, extend the $\mathcal{C}$-partition $\mathcal{P}$ by adding u into the set C_i and increase W by $\sum_{uv \in E, v \in C_i} w(uv)$. If $S[i] = \mathcal{C}$ then we cannot color u with the color i as the corresponding set is already of size $\mathcal{C}$.

Forget Nodes. Let t be an forget node, with c being its child node and u be the newly introduced vertex. For each tuple $(Col, S, W, \mathcal{P})$ of c we create one tuple $(Col', S', W', \mathcal{P}')$ for t. Let $Col(u) = i$. We consider two cases: either $C_i \cap B_t = \emptyset$ or not. In the former, we have that the color i does not appear on any vertex of $B_c \setminus \{u\} = B_t$. Therefore, we are free to reuse this color. To do so, we set $S'[i] = 0$, and we modify $\mathcal{P}$. In particular, if $\mathcal{P} = \{C_1, \ldots, C_k\}$, we create a new $\mathcal{C}$-partition $\mathcal{P}' = \{C'_1 \ldots, C'_{k+1}\}$ where $C'_j = C_j$ for all $j \in [k] \setminus \{i\}$, $C'_i = \emptyset$ and $C'_{k+1} = C_i$. Also, we define Col' as the restriction of the function Col to the set B_t. Finally, $W' = W$. In the latter case, it suffices to restrict Col to the set B_t. We keep all the other information the same.

Join Nodes. Let t be a join node, with c_1 and c_2 being its children nodes. For any pair of tuples $(Col_1, S_1, W_1, \mathcal{P}_1)$ and $(Col_2, S_2, W_2, \mathcal{P}_2)$ of c_1 and c_2 respectively, we will create a tuple $(Col, S, W, \mathcal{P})$ for t if: (i) $Col_1(u) = Col_2(u)$ for all $u \in B_t$ and (ii) $S_1[i] + S_2[i] - |C_i \cap B_t| \leq \mathcal{C}$ for all $i \in [\text{tw}+1]$, where C_i is the i^{th} set of $\mathcal{P}_1$. The choice of $\mathcal{P}_1$ here is arbitrary because of the first condition. Indeed, the first condition guarantees that $\mathcal{P}_1$ and $\mathcal{P}_2$ "agree" on the vertices of B_t. That is, the vertices of B_t are partitioned in the same sets according to $\mathcal{P}_1$ and $\mathcal{P}_2$. The second condition guarantees that the sets created for $\mathcal{P}$ are of size at most $\mathcal{C}$. The tuple $(Col, S, W, \mathcal{P})$ is created as follows. We set: (i) $Col(u) = Col_1(u)$ for all $u \in B_t$, (ii) $S[i] = S_1[i] + S_2[i] - |C_i \cap B_t|$ for all $i \in [\text{tw}+1]$, and (iii) $W = W_1 + W_2 - \sum_{uv \in E(G[B_t]), Col(u) = Col(v)} w(uv)$. Once more, C_i is chosen w.l.o.g. to be the i^{th} set of $\mathcal{P}_1$. Finally, let $\mathcal{P}_1 = \{C_1^1, \ldots, C_p^1\}$ and $\mathcal{P}_2 = \{C_1^2, \ldots, C_{p'}^2\}$; we create the $\mathcal{C}$-partition $\mathcal{P} = \{C_1 \ldots, C_{p+p'-\text{tw}-1}\}$ as follows. For any $i \in [\text{tw}+1]$, set $C_i = C_i^1 \cup C_i^2$. For any $i \in [p] \setminus [\text{tw}+1]$, set $C_i = C_i^1$. Last, for any $i \in [p'] \setminus [\text{tw}+1]$, set $C_{p+i} = C_i^2$. This finishes the description of our algorithm. It remains to compute its running time.

First, we calculate the number of different types of $\mathcal{C}$-partitions for a node t. We have at most $(\text{tw}+1)^{\text{tw}+1}$ different functions Col and $(\mathcal{C}+1)^{\text{tw}+1}$ different tables S. Thus, we have $(\text{tw} \cdot \mathcal{C})^{\mathcal{O}(\text{tw})}$ different types for each node. Since we keep only one tuple per type, we are storing $(\text{tw} \cdot \mathcal{C})^{\mathcal{O}(\text{tw})}$ tuples for each node of $\mathcal{T}$. Moreover, for the leaf nodes, we need to create just one tuple. For the introduce and forget nodes, we need to consider each tuple of their children once. Therefore, we can compute all tuples for these nodes in time $(\text{tw} \cdot \mathcal{C})^{\mathcal{O}(\text{tw})}$. For the join nodes, in the worst case, we need to consider all pairs of tuples of their children that share the same coloring function. This still does not result in more than $(\text{tw} \cdot \mathcal{C})^{\mathcal{O}(\text{tw})}$ combinations. Finally, all the other calculations remain polynomial to the number of vertices. ◇

Theorem 2. *$\mathcal{C}$-CF is W[1]-hard parameterized by the tree-depth of the input graph.*

Sketch of proof. We present a reduction from the GENERAL FACTORS (GF for short) problem. In this problem, we are given a graph $H = (V, E)$ and a list function $L : V \to \mathcal{P}(\{0, \ldots, \Delta(H)\})$ that specifies the available degrees for each vertex $u \in V$. The question is whether there exists a set $S \subseteq E$ such that $d_{H-S}(u) \in L(u)$ for all $u \in V$.

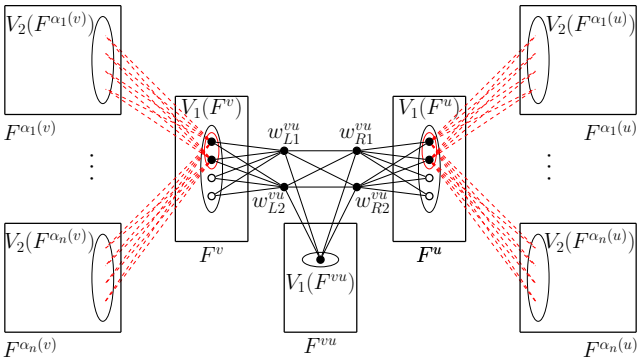


Figure 2: The graph G constructed in the proof of Theorem 2

It is known that GF in connected bipartite graphs is $\mathsf{W}[1]$-hard, even when parameterized by the size of the smallest bipartition (Gutin et al. 2012). Let (H, L) be an instance of GF where $H = (V_L, V_R, E)$ is a bipartite graph ($V(H) = V_L \cup V_R$ and $E(H) = E$) and $L : V_L \cup V_R \to \mathcal{P}([|V(H)|])$ gives the list of degrees for each vertex.

We now describe the two different gadget graphs $F_{x,z-2y}$ and $F_{x,y,z}$ that will be used for the construction that follows (illustrated in Figure 1). $F_{x,z-2y}$ is constructed as follows: (i) We create two independent sets U and V of size x and $z-2y$, respectively, (ii) we add all edges between vertices of U and V, and (iii) we add edges between vertices of V so that the graph induced by V contains a matching wit $2xy$ edges. Hereafter, for any gadget $F_{x,z-2y} = F$ we will refer to U as $V_1(F)$ and to V as $V_2(F)$.

The construction of $F_{x,y,z}$ is as follows: (i) We create three independent sets U, V and W of size x, $\mathcal{C} - y$ and z, respectively, and (ii) we add all edges between vertices of U and V and all edges between vertices of V and W. Hereafter, for any gadget $F_{x,y,z} = F$ we will refer to U as $V_1(F)$, to V as $V_2(F)$ and to W as $V_3(F)$.

We are now ready to describe the construction of our reduction, illustrated in Figure 2. First, for each vertex $v \in V(H)$, we create a copy of $F_{4,y,z}$; we denote this gadget as F^v. We also fix a set $U(F^v) \subset V_1(F^v)$ such that $|U(F^v))| = 2$. Now, for any vertex $v \in V(H)$ and integer $\alpha \in L(v)$, we create a copy of $F_{x,y-2\alpha}$ and add all edges between $V_2(F_{x-2\alpha})$ and $U(F^v)$. Finally, for each edge $uv \in E(H)$, where $u \in V_L$ and $v \in V_R$, we create a copy, F^{uv} of the $F_{1,y,z}$ gadget and a set of vertices $V_{uv} = \{w^{uv}_{L1}, w^{uv}_{L2}, w^{uv}_{R1}, w^{uv}_{R2}\}$. We add all the edges between $V_1(F^{uv})$ and V_{uv}, all the edges between $V_1(F^u)$ and $\{w^e_{L1}, w^e_{L2}\}$, all the edges between $V_1(F^v)$ and $\{w^{uv}_{R1}, w^{uv}_{R2}\}$ and the edges w^{uv}_{Li}, w^{uv}_{Rj} for all $i, j \in [2]$ (*i.e.*, the V_{uv} induces a $K_{2,2}$). Let G be the constructed graph.

The reduction works for a carefully chosen value for $\mathcal{C}$. Also, each gadget that is added has the number of its vertices carefully tweaked through changing the values of the x, y and z. We proceed by showing that in any optimal $\mathcal{C}$-partition of G, for every gadget, its vertices belong in the same set of the partition. Moreover, each gadget F^v will be in the same set as exactly one of the F^as. Then, we are left with the vertices of V^{uv}, which will serve as translators between the two problems. Intuitively, if an edge uv does not appear in the set S of the solution of the GF problem, then any optimal $\mathcal{C}$-partition $\mathcal{P}$ of G will be such that the vertices of V_{uv} will be in the same set as the vertices of F^{uv}. In particular, every $\mathcal{C}$-partition $\mathcal{P}$ of G that has a value exceeding a threshold, will be such that the vertices of V_{uv} will be split to the different sets that contain the vertices of F^v and F^u if and only if the edge uv belongs in the set S. ◇

Theorem 3. *$\mathcal{C}$-CFw can be solved in time* $\text{vc}^{\mathcal{O}(\text{vc})}n^{\mathcal{O}(1)}$.

Proof. Let U be a vertex cover of G of size vc and let I be the independent set $V \setminus U$. If such a vertex cover is not provided as input, we can compute one in time $2^{\text{vc}}n^{\mathcal{O}(1)}$ time (Cygan et al. 2015). First, observe that there can be at most vc many coalitions in G which can have a positive contribution (since the contribution comes from edges and each edge in G is incident to some vertex in U). Next, we guess $\mathcal{P}' = \{C_1, \ldots, C_p\}$ (here, $p \leq$ vc), the non-empty intersection of the sets of an optimal $\mathcal{C}$-partition of G with U; let $W = v(\mathcal{P}')$. We can enumerate all $\text{vc}^{\mathcal{O}(\text{vc})}$ partitions of U in $\text{vc}^{\mathcal{O}(\text{vc})}$ time.

Next, for each $\mathcal{P}'$ we do the following (in $n^{\mathcal{O}(1)}$ time). We create a new graph G' as follows. First, we create the vertex sets S_i, where $|S_i| = \mathcal{C} - C_i$ for each $i \in [p]$. Then, we add all the edges between the vertices of $x \in I$ and S_i if $v \in N(C_i)$, for every $i \in [p]$. Formally, $S_i = \{u^i_1, \ldots, u^i_{S_i}\}$ for $i \in [p]$, $V(G') = \bigcup_{i\in[p]} S_i$, and $E(G') = \{u^i_j x \mid x \in N(C_i) \cap I, i \in [p], j \in [S_i]\}$. Finally, for every edge xy, with $x \in S_i$ and $y \in I$, we set the weight $w(xy) = \sum_{u \in C_i \wedge uy \in E} w(uy)$, i.e., to be equal with to much W would increase if y was added to C_i. Now, observe that in order to compute an optimal $\mathcal{C}$-partition of G whose intersection with U is $\mathcal{P}'$, it suffices to find a maximum weighted matching of G', which can be done in polynomial time (Edmonds 1965). Since we do this operation for each possible intersection of the $\mathcal{C}$-partition with U, all of which can be enumerated in $\text{vc}^{\mathcal{O}(\text{vc})}$ time, we can compute an optimal $\mathcal{C}$-partition of G in time $\text{vc}^{\mathcal{O}(\text{vc})}n^{\mathcal{O}(1)}$. □

Parameterized complexity offers a rich toolkit for establishing conditional lower bounds on the running time of possible FPT algorithms. We use the (arguably) most used tool in this regard, ETH, to argue that our algorithms from Theorems 1 and 3 are asymptotically optimal. In particular, our algorithms from Theorem 3 is optimal asymptotically assuming ETH.

Theorem 4. *There is no algorithm that solves $\mathcal{C}$-CF in time $(\mathcal{C} \cdot \text{vc})^{o(\text{vc}+\mathcal{C})}n^{\mathcal{O}(1)}$, unless the ETH fails.*

Sketch of proof. We present a reduction from R3-SAT, a restricted version of 3-SAT. The input consists of a 3-SAT formula ϕ defined on a set of variables X and a set of clauses C. Additionally, each variable appears at most four times in C and X is partitioned into $X_1 \cup X_2 \cup X_3$, such that every clause includes at most one variable from each one of the sets X_1, X_2 and X_3. The question is whether there exists a satisfying truth assignment of ϕ. We prove that the R3-SAT

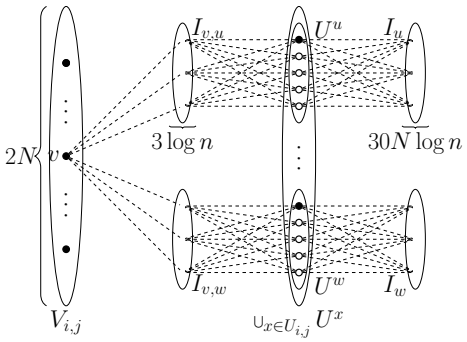


Figure 3: The gadget $G_{i,j}$ constructed in Theorem 4

problem is NP-hard. Also, under the ETH, there is no algorithm that solves this problem in time $2^{o(n+m)}$, where n is the number of variables and m is the number of clauses.

The construction. Let (X, C) be an instance of the R3-SAT problem, and $X = X_1 \cup X_2 \cup X_3$. We partition each variable set X_i into $k = \log n$ sets $X_{i,1} \ldots, X_{i,k}$, with $|X_{i,j}| \leq \lceil n/\log n \rceil$ for every $j \in [k]$. For each set $X_{i,j}$, we construct a variable gadget $G_{i,j}$ (see Figure 3):

- Create a *variable* vertex set $V_{i,j}$ with $2N = 2\lceil n/\log^2 n \rceil$ vertices. Each vertex v in $V_{i,j}$ represents a set $X(v)$ of at most $\frac{\log n}{2}$ variables. Notice that $X(v) \subseteq X_{i,j} \subseteq X_i$ for all $v \in V_{i,j}$.
- Then create the set of *assignment vertices* $U_{i,j} = \{u_\ell \mid \ell \in [\sqrt{n}]\}$. For each vertex $v \in V_{i,j}$ and each assignment over $X(v)$, we want a vertex of $U_{i,j}$ to represent this assignment. Since $|X(v)| \leq \frac{\log n}{2}$, there are at most $2^{\frac{\log n}{2}}$ different assignments over $X(v)$. Thus, we can select the variables of $U_{i,j}$ to represent the assignments over $X(v)$ so that each assignment is represented by at least one vertex and no vertex represents more than one assignment.
- We then create four copies $u^1, \ldots, u^4$ of each vertex $u \in U_{i,j}$. For each assignment vertex u, let U^u be the set $\{u, u^1, \ldots, u^4\}$. For each set U^u, we add an independent set I_u of size $30N \log n$. Then, for each vertex $v \in I_u$, we add all the edges between v and the vertices of U^u.
- Finally, for each pair $(v, u) \in V_{i,j} \times U_{i,j}$, we create an independent set $I_{v,u}$ of $3 \log n$ vertices and, for all $w \in I_{v,u}$ and $x \in U^u \cup \{v\}$, we add the edge wx.

This concludes the construction of $G_{i,j}$, which corresponds to the set $X_{i,j}$.

Let V_C be the set of clause vertices, which contains a vertex v_c for each $c \in C$. We add the vertices of V_C to the graph we are constructing. The edges incident to the vertices of V_C are added as follows. Let $c \in C$, l be a literal that appears in c, x be the variable that appears in l and v the variable vertex such that $x \in X(v)$. Add the edge $v_c v$. Now, consider the $(i, j) \in [3] \times [\log n]$ such that $v \in V_{i,j}$. For each $u \in U_{i,j}$, add the edge uv_c if and only if l becomes true by the assignment over $X(v)$ represented by u. Let G be the resulting graph.

Finally, set $\mathcal{C} = 42N \log n$, where, $N = \lceil n/\log^2 n \rceil$. This finishes our construction. Observe that the vertex set containing the $V_{i,j}$s, the $U_{i,j}$s and the copies of the vertices in the $U_{i,j}$s, for every (i, j), is a vertex cover of G.

Properties of optimal $\mathcal{C}$-partitions of G and reduction. Any optimal $\mathcal{C}$-partition $\mathcal{P}$ of G has a very particular structure. This is due to the structural characteristics of G as well as the specific value chosen for $\mathcal{C}$. In particular, we know that any set S of $\mathcal{P}$ contains exactly one assignment vertex u. Moreover, if S contains a clause vertex v_c (representing the clause c) and the $v(\mathcal{P})$ is greater than some threshold, then it also contains a vertex v from a variable set, such that the assignment over $X(v)$ corresponding to u satisfies c. $\diamond$

Peeling the Input

Polynomial Kernel for $\mathcal{C}$-CF. To get our kernel, we will use an auxiliary bipartite graph H as follows. Let U be a vertex cover in G and let $I = V(G) \setminus U$. Then, $V(H)$ contains two partitions X and Y such that $V(Y) = I$ and for each $u \in U$, we add $t = \text{vc} \times \mathcal{C} + \mathcal{C}$ many vertices $u_1, \ldots, u_t$. Moreover, if $uv \in E(G)$ such that $u \in U$ and $v \in I$, we add the edge $u_i v$ in H for each $i \in [t]$. Now, we compute a maximum matching $\mathcal{M}$ in H. Let $Y' \subseteq Y$ be the set of vertices that are not matched in $\mathcal{M}$. We have the following reduction rule (RR).

(RR): Delete an arbitrary vertex $w \in Y'$ from I.

Lemma 1. *RR is safe.*

Sketch of proof. In any $\mathcal{C}$-partition $\mathcal{P}$ of G, at most vc $\times \mathcal{C}$ many vertices can participate in sets $C \in \mathcal{P}$ such that $C \cap U \neq \emptyset$. These are the only vertices of I that can contribute in the value of $\mathcal{P}$. Let $G' = G[V(G) \setminus \{w\}]$. Since any $\mathcal{C}$-partition $\mathcal{P}^*$ of G' can be extended to a $\mathcal{C}$-partition of G by adding to it a singleton set $C = \{w\}$, it suffices to show that if $\mathcal{P}$ is an optimal $\mathcal{C}$-partition of G, then there exists a $\mathcal{C}$-partition $\mathcal{P}^*$ such that $\{w\} \in \mathcal{P}^*$ and $v(\mathcal{P}^*) = v(\mathcal{P})$. Indeed, $\mathcal{P}^* \setminus \{w\}$ is a $\mathcal{C}$-partition of G' and $v(\mathcal{P}^*) = v(\mathcal{P}^* \setminus \{w\}) = v(\mathcal{P})$; thus $\mathcal{P}^* \setminus \{w\}$ is a $\mathcal{C}$-partition of G' (due to optimality of $\mathcal{P}$). It remains to prove that such a $\mathcal{C}$-partition exists. If $\{w\}$ is a singleton in $\mathcal{P}$ then $\mathcal{P}^* = \mathcal{P}$. Therefore, we assume that w participates in some set $C \neq \{w\}$ of $\mathcal{P}$. Let $x \in U \cap C$ such that $xw \in E(G)$. Then, $x_1, \ldots, x_t$ are matched to $t = \text{vc} \times \mathcal{C} + \mathcal{C}$ many vertices of I by the maximum matching in H (as $w \notin Y$); let S_x be the set of these vertices. Note that at least $\mathcal{C}$ of the vertices in S_x are not contributing in the value of $\mathcal{P}$. We create a new set C' by moving $\mathcal{C} - 1$ of these vertices (which contain vertices that are connected to x) into C' and move x from C to C'. Then, $v(\mathcal{P}') \geq v(\mathcal{P})$ as we remove at most $\mathcal{C} - 1$ edges incident on x in C and add exactly $\mathcal{C} - 1$ edges incident on x in C'. We repeat this step until w is no longer connected to any vertex in C, and at this point, we can delete w from C (since its contribution is 0) and add it as a singleton. $\diamond$

Once we cannot apply RR anymore, $|V(G)| = \mathcal{O}(\text{vc}^2\mathcal{C})$, completing the proof of Theorem 5.

Incompressibility for $\mathcal{C}$-CFw. In this section, we establish that, unless the polynomial hierarchy collapses, $\mathcal{C}$-CFw parameterized by vc does not admit a polynomial kernel. To this end, we provide a *polynomial parameter transformation*

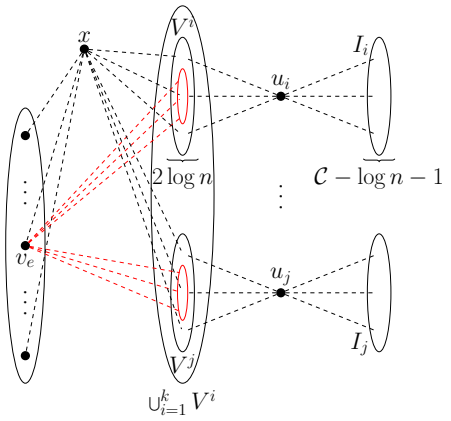


Figure 4: The graph G constructed in the proof of Theorem 6

from k-MULTICOLORED CLIQUE (k-MC for short) parameterized by $k + \log n$.

Theorem 6. *$\mathcal{C}$-CFw parameterized by* vc+$\mathcal{C}$ *does not admit a polynomial kernel, unless polynomial hierarchy collapses.*

Sketch of proof. The reduction is from k-MC, where given a graph $H = (V, E)$ and a partition $(V_1, \ldots, V_k)$ of V into k independent sets, the question is whether there exists a set $S \subseteq V$ such that $G'[S]$ is a clique. It is known that k-MC does not admit a kernel of size $poly(k + \log n)$, unless the Polynomial Hierarchy collapses (Hermelin et al. 2015).

We construct an instance of $\mathcal{C}$-CF as follows, for $\mathcal{C} = \binom{k}{2} + k \log n + 1$ (see Figure 4). For each set V_i, we first create a clique of $2 \log n$ vertices V^i. Then, for each $i \in [k]$, we create a vertex u_i and an independent set I_i of size $\mathcal{C} - \log n - 1$. Finally, we add all edges between u_i and $V^i \cup I_i$. We then relate the vertices of $V_i = \{v_1, \ldots, v_n\}$, for each $i \in [k]$, to a subset of V^i. For each $j \in [n]$, we assign to v_j the string (of length $\log n$) that corresponds to the binary representation of j; let $s(v)$ be the string assigned to a vertex v of H. Also, for each $i \in [k]$ and $v \in V_i$, we define the set $S(v) \subseteq V^i$ as follows: for each $\ell \in [\log n]$, if the ℓ-th letter of $s(v)$ is 0 then we add u^i_ℓ to $S(v)$. Otherwise, we add v^i_ℓ to $S(v)$. We continue by creating one vertex v_e for each edge $e \in E(H)$. Let V_e be the set of these "edge" vertices. For each edge $uv = e \in E(H)$, we add the edges $v_e w$, where $w \in S(u) \cup S(v)$. Finally, we add a vertex x and all the edges between x and any vertex in $V_e \cup \bigcup_{i \in [k]} V^i$. The weights of all the edges are chosen in a specific manner; due to lack of space, we will just mention the weights later in the sketch, where they become relevant. Let G be the resulting graphs.

The rest of the proof consists in showing that there is a clique of order k in H if and only if there exists a $\mathcal{C}$-partition of G with $v(\mathcal{P})$ above some threshold. Intuitively, the weights of the edges incident to the u_is are chosen so that every set of $\mathcal{P}$ contains at most one of the u_is. Also, if a set of $\mathcal{P}$ contains a u_i, then it also contains $\log n$ vertices from V^i; the remaining $\log n$ vertices of V^i are the encoding of a vertex $v \in V_i$. We then build an equivalence between locating the vertices in the V_is that can form a clique in H and having a set in $\mathcal{P}$ that contains x, $\ell = \binom{k}{2}$ "edge" vertices $e_{u_1,v_1}, \ldots, e_{u_\ell,v_\ell}$ and the $k \log n$ vertices contained in $S(u_1), \ldots, S(u_\ell)$ and $S(v_1), \ldots, S(v_\ell)$ that correspond to the encodings of the vertices $u_1, \ldots, u_\ell$ and $v_1, \ldots, v_\ell$.

Finally, in order for our claim to hold, we also need to observe that $U = \{x\} \cup \bigcup_{i \in [k]} (V^i \cup \{u_i\})$ is a vertex cover of G and that $|U| + \mathcal{C} \in \mathcal{O}(k \log n)$. ◇

Rounding up the Structural Parameters

Theorem 7. *$\mathcal{C}$-CF is FPT parameterized by the vertex integrity of the input graph.*

Sketch of proof. Once we have a U according to the definition of the vertex integrity, we can guess $\mathcal{P}'$, the intersection of the sets of an optimal $\mathcal{C}$-partition of G with U. Let $k = |U|$. Then, we organize the connected components of $G[V \setminus U]$ according to their structural characteristics; the number of different such types is bounded by a function of k. We reduce the problem of extending $\mathcal{P}'$ into an optimal $\mathcal{C}$-partition of G to solving an ILP with a number of variables that is bounded by a function of k. A solution of this ILP can be obtained in FPT time, parameterized by k (by running for example, the Lenstra algorithm (Lenstra Jr. 1983)). ◇

Theorem 8. *$\mathcal{C}$-CF is W[1]-hard when parameterized by the twin-cover number of G.*

Sketch of proof. The reduction is from the UNARY BIN PACKING (UBP) problem. UBP takes as input a set of items $A = \{a_1, \ldots, a_n\}$, a *size function* $s : A \to \mathbb{N}$ which returns the size of each item in unary encoding, and two integers B and k. The question is whether the items of A can fit into k bins, so that every bin contains items of total size exactly B, and every item is assigned to exactly one bin. This problem is W[1]-hard parameterized by k (Jansen et al. 2013).

Let (A, s, B, k), where $A = \{a_1, \ldots, a_n\}$, be an instance of UBP. We construct an instance of $\mathcal{C}$-CF as follows: for each $j \in [n]$, construct the clique K^j of order $s(a_j)$. Then, for each $i \in [k]$, add one vertex b_i and all the edges between b_i and all the vertices of the cliques K^j, for all $j \in [n]$. Let G be the resulting graph, and set $\mathcal{C} = B + 1$. Observe that the twin-cover number of G is at most k, as the set $\{b_1, \ldots, b_k\}$ is a twin-cover of G. The rest of the proof consists in showing that any optimal partition $\mathcal{P}$ of $(G, \mathcal{C})$ has value $v(\mathcal{P}) = \sum_{j=1}^{n} \frac{s(a_j)(s(a_j)-1)}{2} + kB$ if and only if (A, s, B, k) is a yes-instance of UBP. ◇

Conclusion

In this paper, we studied $\mathcal{C}$-CFw, considering both its weighted and unweighted versions, through the lens of parameterized complexity. The main takeaway message is that the problems behave relatively well in regards to many widely used parameters, despite the multiple intractability results that we provided. Moreover, our intractability results provide motivation towards a more heuristic-oriented approach. Finally, we wonder about the existence of an FPT algorithm for $\mathcal{C}$-CFw parameterized by other interesting parameters that are left untouched by our work, such as the vertex integrity, the neighborhood diversity and the feedback vertex set of the input graph.

Acknowledgments

FF and NM acknowledge the support by the European Union under the project Robotics and advanced industrial production (reg. no. CZ.02.01.01/00/22_008/0004590) and the CTU Global postdoc fellowship program.

The full version of our paper is available here (Fioravantes, Gahlawat, and Melissinos 2024).

References

Aloisio, A.; Flammini, M.; and Vinci, C. 2020. The impact of selfishness in hypergraph hedonic games. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Aziz, H.; Brandl, F.; Brandt, F.; Harrenstein, P.; Olsen, M.; and Peters, D. 2019. Fractional hedonic games. *ACM Transactions on Economics and Computation (TEAC)*, 7(2): 1–29.

Aziz, H.; Brandt, F.; and Seedig, H. G. 2013. Computing desirable partitions in additively separable hedonic games. *Artificial Intelligence*, 195: 316–334.

Bachrach, Y.; Kohli, P.; Kolmogorov, V.; and Zadimoghaddam, M. 2013. Optimal coalition structure generation in cooperative graph games. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Bäckström, C.; Chen, Y.; Jonsson, P.; Ordyniak, S.; and Szeider, S. 2012. The complexity of planning revisited—a parameterized analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Barrot, N.; Ota, K.; Sakurai, Y.; and Yokoo, M. 2019. Unknown agents in friends oriented hedonic games: Stability and complexity. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Barrot, N.; and Yokoo, M. 2019. Stable and Envy-free Partitions in Hedonic Games. In *IJCAI*.

Bessiere, C.; Hebrard, E.; Hnich, B.; Kiziltan, Z.; Quimper, C. G.; and Walsh, T. 2008. The parameterized complexity of global constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Bilò, V.; Monaco, G.; and Moscardelli, L. 2022. Hedonic games with fixed-size coalitions. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Bodlaender, H. L. 1996. A Linear-Time Algorithm for Finding Tree-Decompositions of Small Treewidth. *SIAM J. Comput.*, 25(6): 1305–1317.

Bodlaender, H. L. 1998. A partial k-arboretum of graphs with bounded treewidth. *Theoretical Computer Science*, 209(1): 1–45.

Boehmer, N.; and Elkind, E. 2020. Individual-based stability in hedonic diversity games. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Bogomolnaia, A.; and Jackson, M. O. 2002. The stability of hedonic coalition structures. *Games and Economic Behavior*, 38(2): 201–230.

Brandt, F.; Bullinger, M.; and Wilczynski, A. 2023. Reaching individually stable coalition structures. *ACM Transactions on Economics and Computation*, 11(1-2): 1–65.

Brandt, F.; Conitzer, V.; Endriss, U.; Lang, J.; and Procaccia, A. D. 2016. *Handbook of computational social choice*. Cambridge University Press.

Bredereck, R.; Chen, J.; Niedermeier, R.; and Walsh, T. 2017. Parliamentary voting procedures: Agenda control, manipulation, and uncertainty. *Journal of Artificial Intelligence Research*, 59: 133–173.

Bullinger, M.; and Kober, S. 2021. Loyalty in Cardinal Hedonic Games. In *IJCAI*.

Chen, J.; Csáji, G.; Roy, S.; and Simola, S. 2023. Hedonic Games With Friends, Enemies, and Neutrals: Resolving Open Questions and Fine-Grained Complexity. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2023, London, United Kingdom, 29 May 2023 - 2 June 2023*.

Cseh, A.; Fleiner, T.; and Harján, P. 2019. Pareto Optimal Coalitions of Fixed Size. *Journal of Mechanism and Institution Design*, 4: 1.

Cygan, M.; Fomin, F. V.; Kowalik, L.; Lokshtanov, D.; Marx, D.; Pilipczuk, M.; Pilipczuk, M.; and Saurabh, S. 2015. *Parameterized Algorithms*. Springer. ISBN 978-3-319-21274-6.

Darmann, A.; Elkind, E.; Kurz, S.; Lang, J.; Schauer, J.; and Woeginger, G. 2018. Group activity selection problem with approval preferences. *International Journal of Game Theory*, 47: 767–796.

Deng, X.; and Papadimitriou, C. H. 1994. On the complexity of cooperative solution concepts. *Mathematics of operations research*, 19(2): 257–266.

Diestel, R. 2012. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer.

Downey, R. G.; and Fellows, M. R. 2013. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer. ISBN 978-1-4471-5558-4.

Drange, P. G.; Dregi, M. S.; and van 't Hof, P. 2016. On the Computational Complexity of Vertex Integrity and Component Order Connectivity. *Algorithmica*, 76(4): 1181–1202.

Dreze, J. H.; and Greenberg, J. 1980. Hedonic coalitions: Optimality and stability. *Econometrica: Journal of the Econometric Society*, 987–1003.

Edmonds, J. 1965. Paths, Trees, and Flowers. *Canadian Journal of Mathematics*.

Fanelli, A.; Monaco, G.; Moscardelli, L.; et al. 2021. Relaxed core stability in fractional hedonic games. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*.

Fellows, M. R. 2006. The Lost Continent of Polynomial Time: Preprocessing and Kernelization. In *IWPEC'06*. ISBN 3540390987.

Fioravantes, F.; Gahlawat, H.; and Melissinos, N. 2024. A parameterized point of view on forming small coalitions. hal-04562753.

Flammini, M.; Monaco, G.; Moscardelli, L.; Shalom, M.; and Zaks, S. 2018. Online coalition structure generation in graph games. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*.

Flum, J.; and Grohe, M. 2006. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer. ISBN 978-3-540-29952-3.

Fomin, F. V.; Lokshtanov, D.; Saurabh, S.; and Zehavi, M. 2019. *Kernelization: Theory of Parameterized Preprocessing*. Cambridge University Press.

Ganian, R. 2011. Twin-Cover: Beyond Vertex Cover in Parameterized Algorithmics. In *Parameterized and Exact Computation - 6th International Symposium, IPEC 2011*.

Ganian, R.; Hamm, T.; Knop, D.; Schierreich, Š.; and Suchỳ, O. 2023. Hedonic diversity games: A complexity picture with more than two colors. *Artificial Intelligence*, 325: 104017.

Gao, Y. 2009. Data reductions, fixed parameter tractability, and random weighted d-CNF satisfiability. *Artificial Intelligence*, 173(14): 1343–1366.

Guo, J.; and Niedermeier, R. 2007. Invitation to data reduction and problem kernelization. *ACM SIGACT News*, 38(1): 31–45.

Gutin, G. Z.; Kim, E. J.; Soleimanfallah, A.; Szeider, S.; and Yeo, A. 2012. Parameterized Complexity Results for General Factors in Bipartite Graphs with an Application to Constraint Programming. *Algorithmica*, 64(1): 112–125.

Hanaka, T.; Ikeyama, A.; and Ono, H. 2023. Maximizing Utilitarian and Egalitarian Welfare of Fractional Hedonic Games on Tree-Like Graphs. In *Combinatorial Optimization and Applications - 17th International Conference, COCOA 2023, Hawaii, HI, USA, December 15-17, 2023, Proceedings, Part I*.

Hanaka, T.; Kiya, H.; Maei, Y.; and Ono, H. 2019. Computational Complexity of Hedonic Games on Sparse Graphs. In *PRIMA 2019: Principles and Practice of Multi-Agent Systems - 22nd International Conference, Turin, Italy, October 28-31, 2019, Proceedings*.

Hanaka, T.; and Lampis, M. 2022. Hedonic Games and Treewidth Revisited. In *30th Annual European Symposium on Algorithms, ESA 2022*.

Hermelin, D.; Kratsch, S.; Soltys, K.; Wahlström, M.; and Wu, X. 2015. A Completeness Theory for Polynomial (Turing) Kernelization. *Algorithmica*, 71(3): 702–730.

Igarashi, A.; Ota, K.; Sakurai, Y.; and Yokoo, M. 2019. Robustness against agent failure in hedonic games. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*.

Impagliazzo, R.; and Paturi, R. 2001. On the Complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2): 367–375.

Jansen, K.; Kratsch, S.; Marx, D.; and Schlotter, I. 2013. Bin packing with fixed number of bins revisited. *Journal of Computer and System Sciences*, 79(1): 39–49.

Kloks, T. 1994. *Treewidth, Computations and Approximations*, volume 842 of *Lecture Notes in Computer Science*. Springer. ISBN 3-540-58356-4.

Korhonen, T.; and Lokshtanov, D. 2023. An Improved Parameterized Algorithm for Treewidth. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023*.

Lee, E. 2017. Partitioning a graph into small pieces with applications to path transversal. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*.

Lenstra Jr., H. W. 1983. Integer Programming with a Fixed Number of Variables. *Mathematics of Operations Research*, 8(4): 538–548.

Levinger, C.; Azaria, A.; and Hazon, N. 2023. Social Aware Coalition Formation with Bounded Coalition Size. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*.

Li, F. 2021. Fractional Hedonic Games With a Limited Number of Coalitions. In *Proceedings of the 22nd Italian Conference on Theoretical Computer Science, Bologna, Italy, September 13-15, 2021*.

Maniu, S.; Senellart, P.; and Jog, S. 2019. An experimental study of the treewidth of real-world graph data. In *ICDT 2019–22nd International Conference on Database Theory*.

Monaco, G.; and Moscardelli, L. 2023. Nash Stability in Fractional Hedonic Games with Bounded Size Coalitions. In *International Conference on Web and Internet Economics*.

Monaco, G.; Moscardelli, L.; and Velaj, Y. 2021. Additively Separable Hedonic Games with Social Context. *Games*, 12(3): 71.

Niedermeier, R. 2006. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press. ISBN 9780198566076.

Ohta, K.; Barrot, N.; Ismaili, A.; Sakurai, Y.; and Yokoo, M. 2017. Core Stability in Hedonic Games among Friends and Enemies: Impact of Neutrals. In *IJCAI*.

Olsen, M. 2009. Nash stability in additively separable hedonic games and community structures. *Theory of Computing Systems*, 45: 917–925.

Peters, D. 2016a. Graphical hedonic games of bounded treewidth. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Peters, D. 2016b. Towards Structural Tractability in Hedonic Games. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Peters, D.; and Elkind, E. 2015. Simple causes of complexity in hedonic games. In *Proceedings of the 24th International Conference on Artificial Intelligence*.

Saad, W.; Han, Z.; Basar, T.; Debbah, M.; and Hjorungnes, A. 2010. Hedonic coalition formation for distributed task allocation among wireless agents. *IEEE Transactions on Mobile Computing*, 10(9): 1327–1344.

Sliwinski, J.; and Zick, Y. 2017. Learning Hedonic Games. In *IJCAI*.

van Rooij, J. M. M.; van Kooten Niekerk, M. E.; and Bodlaender, H. L. 2013. Partition Into Triangles on Bounded Degree Graphs. *Theory Comput. Syst.*, 52(4): 687–718.