Online Coalition Structure Generation in Graph Games

Michele Flammini
GSSI Institute & Univ. of L'Aquila
L'Aquila, Italy
michele.flammini@univaq.it

Gianpiero Monaco
University of L'Aquila
L'Aquila, Italy
gianpiero.monaco@univaq.it

Luca Moscardelli
University of Chieti-Pescara
Pescara, Italy
luca.moscardelli@unich.it

Mordechai Shalom
Tel-Hai Academic College
Upper Gallilee, Israel
cmshalom@telhai.ac.il

Shmuel Zaks
Dept. of Computer Science, Technion
Haifa, Israel
zaks@cs.technion.ac.il

ABSTRACT

We consider the online version of the coalition structure generation in graph games problem, where agents are vertices in a graph. After each step t, in which the t-th agent appears in an online fashion, agents are partitioned into $c(t)$ coalitions $\mathbf{C}(t) = \{C_1^t, C_2^t, \ldots, C_{c(t)}^t\}$, such that every agent belongs to exactly one coalition C_i^t. When an agent appears, it may either join an existing coalition or form a new one having it as the only agent. The profit of a such a coalition structure $\mathbf{C}(t)$ is the sum of the profits of its coalitions. We consider two cases for the profit of a coalition: (1) the sum of the weights of its edges (which represents the total profit of the agents in the coalition), and (2) the sum of the weights of its edges divided by its size (which represents the average profit of the agents in the coalition). Such coalition structures appear in a variety of application in AI, multi-agent systems, networks, as well as in social networks, data analysis, computational biology, game theory, and scheduling. For each of the profit functions we consider the bounded and unbounded cases depending on whether or not the size of a coalition can exceed a given value α. Furthermore, we consider the case of a limited number of coalitions and various weight functions for the edges, namely the cases of unrestricted, positive and constant weights. We show tight or nearly tight bounds for the competitive ratio in each case.

KEYWORDS

Coalition structure generation; Online algorithms; Multiagent systems

ACM Reference Format:
Michele Flammini, Gianpiero Monaco, Luca Moscardelli, Mordechai Shalom, and Shmuel Zaks. 2018. Online Coalition Structure Generation in Graph Games. In *Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), Stockholm, Sweden, July 10–15, 2018,* IFAAMAS, 9 pages.

1 INTRODUCTION

Coalition structure generation (CSG) or clustering is a major research challenge in AI, multi-agent systems, and networking communities. The problem is partitioning a set of agents into coalitions so that the social welfare is maximized. Specifically, given a set of agents $A = \{1, 2, \ldots, n\}$, and a *value function* $v : 2^A \to \mathbb{R}$ (note that v may map to negative values), that assigns a value to each set of agents (*coalition* or *cluster*) $S \subseteq A$, a coalition structure is a partition of A, into disjoint, exhaustive coalitions. Several papers (see the Related Work section) considered the problem of partitioning agents into disjoint clusters, by a centralized algorithm, so that the overall outcome of the system, that is the sum of all cluster values, is maximized.

CSG models real scenarios. For instance, consider a set of agents who can work in teams. Some agents work well together, while others find it hard to do so. When two agents work well together, a team which contains both of them can achieve better results due to the synergy between them. However, when two agents find it hard to work together, a team that contains both agents has a reduced utility due to their inability to cooperate, and may perform better when one of them is removed. The problem is partitioning agents into teams in order to maximize the total utility.

CSG have been also considered from an algorithmic game theoretic point of view, where agents are supposed to be selfish. Hedonic games, introduced in [16], describe the dependence of a player's utility on the identity of the members of her group. They are games in which players have preferences over the set of all possible player partitions (called clusterings). In particular, the utility of each player only depends on the composition or structure of the cluster she belongs to. Several papers (see the Related Work section) considered different forms of clustering stability like the core, Nash and individual stability.

If the problem is defined by the 2^n distinct coalition values, the mere specification of the input would be intractable. Therefore, researchers have focused on succinct description of the problem (while still allowing it to capture elaborate games). A widely studied setting, introduced in [15], and also studied in other works, see for instance [27], is the one in which the agents are vertices of a graph, and the value of a coalition is the sum of the weights of the edges between coalition members. In the literature, such settings are commonly referred to as weighted graph games [5].

Most of the papers dealing with CSG, assume that all the information on the input is known at the beginning. However, in more realistic scenarios (e.g., hiring employees and assigning them to a team), agents arrive over time.

In this work we study CSG, with agents introduced in an online fashion. When an agent arrives, it knows the weights of edges between itself and all agents that arrived previously. The agent has to decide whether to join it to an existing cluster or to create a new

Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), M. Dastani, G. Sukthankar, E. André, S. Koenig (eds.), July 10–15, 2018, Stockholm, Sweden.

cluster. The decision depends on the cost function associated with the resulting structure. Every setting of CSG in which (i) agents arrive over time and (ii) an irrevocable choice has to be made upon their arrival naturally fits our model. Generally speaking, looking at a social network as a dynamic entity is very natural. For instance, consider (a) social-network games (among the most played in the world) receiving players over time to be assigned to rooms and not allowed to change room before the end of the game; (b) a research institute aiming at assigning researchers (hired over time) to departments: the cost of moving a researcher already inserted in a department could be very high in terms of productivity and of organization and administrative issues; (c) similarly, a company with geographically spread agencies to which hired employees have to be assigned.

We model a network by an undirected weighted graph. The nodes (i.e., agents) of the network appear online, and form clusters. After each step t the network is partitioned into clusters $\mathbf{C}(t) = \{C_1^t, C_2^t, \ldots, C_{c(t)}^t\}$, such that every node belongs to exactly one cluster C_i^t. When a node appears, it can either join existing cluster or form a new one consisting only of the current node. The profit of a clustering $\mathbf{C}(t)$ is the sum of the profits of all of its clusters. We consider two cases for the profit of a cluster: (1) the sum of the weights of its edges, and (2) the sum of the weight of its edges divided by its size. We refer to these two profit functions as *total weight* and *fractional weight*, respectively.

Our Contribution. We consider the online variant of the problem in which nodes are presented one at a time and two different profit measures for a cluster, namely, the total weight, and the fractional weight. In addition, the input contains two numbers $\alpha, k > 0$, that constitute upper bounds for the size of a cluster and for the number of clusters, respectively. Furthermore, we consider different types of weight functions, that is, the cases of unrestricted, positive and constant weights. We show tight or nearly tight bounds for the competitive ratio in each of the cases. Table 1 and Table 2 summarize our results for the total weight and fractional weight measures, respectively. Our main technical results are the $\Omega(\log^2 W)$ lower bound (Thm. 4.6) for the competitive ratio of Maximum Fractional Weight Clustering with positive weights, and the matching upper bound (Thm. 4.4), where W is the maximum absolute value of the edge weights.

Bounds	Weights	Lower Bound	Upper Bound
$\alpha = \infty$	General	$W \cdot (n-2)$ (Thm. 3.1)	$W \cdot (n-1)$ (Thm. 3.2)
$\alpha < \infty$	General	$2W \cdot (\alpha - 1)$ (Thm. 3.5)	$2W \cdot (\alpha - 1)$ (Thm. 3.7)
	Positive	$\frac{W \cdot (\alpha-1)}{o(W)}$ (Thm. 3.8)	$W \cdot \alpha$ (Thm. 3.9)
$k < \infty$	± 1	Unbounded (Thm. 3.3)	

Table 1: The competitive ratio of Maximum Weight Clustering.

Related Work. [21] proposed one of the most efficient centralized algorithms for the CSG problem, which returns an optimal solution in time $O(3^n)$. Anytime algorithms which can return a solution at anytime during the running time, with the property that the quality of this solution improves monotonically as the computation time increases, have been developed (e.g., [22, 24, 25]) for the coalition structure generation problem.

Bounds	Weights	Lower Bound	Upper Bound
$\alpha = \infty$	General	$4W$ (Thm. 4.1)	$4W$ [4]
	Unweighted	4 (Thm. 4.3)	4 [4]
	Positive	$\Omega(\log^2 W)$ (Thm. 4.6)	$O(\log^2 W)$ (Thm. 4.4)
$\alpha < \infty$	General	$\Omega(W)$ (Thm. 4.9)	$4W$ [4]
	Unweighted	$4(1 - 1/\alpha)$ (Thm. 4.7)	$4(1 - 1/\alpha)$ (Thm. 4.8)
$k < \infty$	± 1	Unbounded (Thm. 4.10)	
	Positive	$\frac{n}{2}$ (Thm. 4.11)	$\frac{n}{2}$ (Observation 1)

Table 2: The competitive ratio of Maximum Fractional Weight Clustering.

All the algorithms that solve the CSG problem optimally have a worst-case time complexity exponential in n, therefore several heuristics have been proposed. For instance, in [26], the authors propose a greedy algorithm which restricts the search space by imposing constraints on the size of the coalition.

A widely studied setting commonly called weighted graph games, introduced in [15] and studied in other works (e.g. [27]), is the one in which the agents are vertices of a graph, and the value of a coalition is the sum of the weights of the edges present between coalition members. In [5], it is showed that finding the optimal coalition structure is hard even for planar graphs. They also provide constant factor approximation algorithms for minor-free graphs (that include the family of planar graphs) and bounded degree graphs.

Most of the literature on multi-agent coalition formation has focused on settings where the value of a coalition does not depend on players who are not part of the coalition. However, in [22], authors consider the coalition structure generation problem for games with externalities. Moreover, [29] studies cooperative games in a setting in which clusters do not constitute a partition of the agents, but may also overlap. A survey of different approaches for the coalition structure generation problem has been presented in [23].

Our problem is also closely related to game theoretic works. Hedonic games have been first formalized in [16], where they are analyzed under a cooperative perspective. Additively separable hedonic games (ASHGs) constitute a natural and succinctly representable class of hedonic games. In this setting, each agent has a value for any other agent, and the utility of a coalition to a particular agent is simply the sum of the values she assigns to the members of her coalition. Additive separability satisfies a number of desirable axiomatic properties [3]. Properties guaranteeing the existence of core allocations for games with additively separable utility have been studied in [9], while [12] also considers other forms of clustering stability like Nash and individual stability. [6], [3] and [19] deal with computational complexity issues related to hedonic games, also considering additively separable utilities.

Fractional hedonic games have been introduced in [2] from the cooperative perspective. They are similar to additively separable hedonic games with the difference that the utility of an agent is divided by the number of agents of the coalition. Fractional hedonic games model natural behavioral dynamics in social environments. In [14], the computational complexity of deciding whether a core and individual stable partition exists in a given fractional hedonic game is studied. [10, 11] deal with Nash stable outcomes in fractional hedonic games, while in [18] the authors consider Nash and core stable outcomes for modified fractional hedonic games, where, slightly differently than fractional hedonic games, the utility of an agent is divided by the size of the coalition she belongs minus 1. [4] considers the computational complexity of computing welfare maximizing partitions (not necessarily Nash stable) for fractional hedonic games. In [20], several classes of hedonic games and fractional hedonic games are considered. Simple sufficient conditions on expressivity for the hardness of the problem of checking whether a given game admits a stable outcome are identified.

Finally, strategyproof mechanisms for additively separable and fractional hedonic games have been proposed in [17, 28].

We note that our profit functions are equivalent to the ones of the corresponding hedonic games and fractional hedonic games, being just scaled by a constant factor of 2.

A related (but different) problem in the online setting was initiated in [1]. They studied the problem of balanced repartitioning: given an online sequence of pairs of nodes to be interconnected, the objective is to dynamically partition the nodes into clusters of similar size, at a minimum cost. Partitioning can be updated dynamically, by migrating nodes between clusters at a given cost per migration. Thus, the three main differences between that model and ours are that we do not require equal size clusters, we consider different value functions, and clusters in our model cannot be reconfigured.

Paper organization. In Section 2 we present definitions and notation used throughout the paper, and also the problems' statement. In Sections 3 and 4 we analyze the total weight measure and the fractional weight measure, respectively. Section 5 contains concluding remarks. Due to space limitations, some proofs are only sketched or omitted.

2 PRELIMINARIES

For an integer $k > 0$, we denote by $[k]$ the set $\{1, \ldots, k\}$.

Through this work G is an undirected edge-weighted graph (V, E, w) on n vertices having no loop, with $w : E \longrightarrow \mathbb{R}$. We denote by uv and $w_{u,v}$, the edge $\{u, v\} \in E$ and its weight $w(\{u, v\})$, respectively. We assume that $|w_{u,v}| \geq 1$, for every $uv \in E$. We denote by $W = \max_{uv \in E} |w_{u,v}|$ the maximum absolute value of the edge weights. We say that G is *unweighted* if $w_{u,v} = 1$ for any $uv \in E$. We denote by $G^+ = (V, E^+, w^+)$ the subgraph of G consisting of its positive-weighted edges. Given a set of edges $F \subseteq E$, we denote by $w(F) = \sum_{uv \in F} w_{u,v}$, the total weight of edges in F. We denote by $G[S]$, the subgraph of G induced by a subset S of its vertices, i.e., $G[S] = (S, E_S, w_S)$, where $E_S = \{uv \in E : u, v \in S\}$ and w_S is the restriction of w to E_S. We denote by $\delta_S(v)$, the set of edges incident to v and S, i.e., $\delta_S(v) = \{uv \in E : u \in S\}$, and by $N_S(v)$ (resp. $N_S[v]$) the open (resp. closed) neighborhood of v in S. A *clique* (resp. *independent set*) of G is a set of pairwise adjacent (resp. non-adjacent) vertices of G.

A *clustering* $\mathbf{C}$ of G is a partition of V into *clusters* $C_1, C_2, \ldots, C_c$, for some positive integer c. We use the term cluster for both C_i and the weighted graph $G[C_i]$. Two clusters C_i and C_j are *adjacent* if there exist $v_i \in C_i$ and $v_j \in C_j$ with $v_i v_j \in E$. For a vertex $v \in V$, we denote by $\mathbf{C}(v)$ the unique cluster $C_i \in \mathbf{C}$ such that $v \in C_i$. For two positive integers α and k, we say that a clustering $\mathbf{C}$ is (α, k)-bounded if $|\mathbf{C}| \leq k$ and $|C_i| \leq \alpha$, for every $C_i \in \mathbf{C}$. We suppose that $\alpha \geq 2$ and $k \geq 2$.

We denote by $w(C_i)$ the total weight of the edges of $G[C_i]$. The *fractional weight* of a cluster C_i is $w_F(C_i) = \frac{w(C_i)}{|C_i|}$. Clearly, when C_i is an independent set, and in particular a single vertex, we have $w_F(C_i) = w(C_i) = 0$. We refer to the unique vertex of a singleton cluster of $\mathbf{C}$ as an *isolated vertex* of $\mathbf{C}$. When G is unweighted we have $w_F(C_i) = \frac{|C_i|-1}{2}$ whenever C_i is a clique, and $w_F(C_i) = 1 - \frac{1}{|C_i|}$ whenever $G[C_i]$ is a tree.

The weight of a clustering $\mathbf{C}$ is $w(\mathbf{C}) = \sum_{C_i \in \mathbf{C}} w(C_i)$, and its fractional weight is $w_F(\mathbf{C}) = \sum_{C_i \in \mathbf{C}} w_F(C_i)$. We name the clustering $\{V\}$ as the GrandCoalition.

Let Π be a maximization problem with objective function f, and I an instance of Π. We denote by $OPT_\Pi(I)$ an arbitrary optimal solution of I. Given an algorithm $\mathcal{A}$ for Π, we denote by $\mathcal{A}(I)$ a solution returned by $\mathcal{A}$ on input I. A feasible solution S of an instance I is a ρ-approximation if $f(S) \geq \frac{f(OPT_\Pi(I))}{\rho}$. An algorithm $\mathcal{A}$ is a ρ-approximation algorithm for Π if every solution $\mathcal{A}(I)$ is a ρ-approximation for every instance I of Π.

An instance of an *online* optimization problem Π is a sequence $I = \sigma_1, \sigma_2, \ldots$. An online algorithm has to produce partial output for every σ_i without the knowledge of the future entries, i.e. $\sigma_{i+1}, \sigma_{i+2}, \ldots$. Furthermore, the output produced by the algorithm at step i cannot be modified at later steps. An online algorithm $\mathcal{A}$ is c-*competitive* for Π if there exists some $b \geq 0$ such that $f(\mathcal{A}(I)) \geq \frac{f(OPT_\Pi(I))}{c} - b$ for every instance I. If $b = 0$ then $\mathcal{A}$ is *strictly* c-competitive. The (strict) competitive ratio of $\mathcal{A}$ is the smallest c such that $\mathcal{A}$ is c-competitive [13]. In this *Extended Abstract* we consider only the strict competitive ratio in our lower bounds. When no ambiguity arises we omit the subscript Π, the instance I and the objective function f. In such cases OPT stands for $OPT_\Pi(I)$ and also for $f(OPT_\Pi(I))$. Similarly, $\mathcal{A}$ may stand for either $\mathcal{A}(I)$ or for $f(\mathcal{A}(I))$ besides being the name of an algorithm.

We consider the following two optimization problems under the online setting in which the vertices of G are presented one at a time in the order $v_1, v_2, \ldots, v_n$ and one has to decide on the cluster $C(v_i)$ of every v_i upon its arrival.

MaxWC(Maximum Weight Clustering)
Input A weighted graph $G = (W, E, w)$. Two positive integers α and k.
Output (α, k)-bounded clustering $\mathbf{C}$.
Objective Maximize $w(\mathbf{C})$.

MaxFWC(Maximum Fractional Weight Clustering)
Input A weighted graph $G = (W, E, w)$. Two positive integers α and k.
Output An (α, k)-bounded clustering **C**.
Objective Maximize $w_F(\mathbf{C})$.

3 MAXIMUM WEIGHT CLUSTERING

3.1 Unbounded Cluster Size

Note that when the size of a cluster is unbounded the case of non-negative weights is trivial, since GrandCoalition is optimal in this case. Therefore, in this section we consider only instances containing both positive and negative edges. We first consider the case where the number of clusters is unbounded, and subsequently the bounded one.

3.1.1 Unbounded Number of Clusters.

Theorem 3.1. *The strict competitive ratio of every deterministic online algorithm for MaxWC is at least $W \cdot (n-2)$.*

Proof. Let $\mathcal{A}$ be a strictly c-competitive deterministic online algorithm for MaxWC. Consider the online input that is supplied to $\mathcal{A}$ by the following adversary. The adversary releases two adjacent vertices v_1 and v_2. If $\mathcal{A}$ does not put both vertices in the same cluster the adversary stops. In this case $OPT = 1$ and $\mathcal{A} = 0$, thus the strict competitive ratio of $\mathcal{A}$ is unbounded. Therefore, $\mathcal{A}$ puts v_1 and v_2 in the same cluster, say C_1. At this point the weight of the solution is 1. The adversary releases x additional vertices each of which is adjacent only to v_1 and v_2 with edges of weight W and $-W$, respectively. The weight of the clustering of $\mathcal{A}$ remains 1, since every vertex will add zero to $f(\mathcal{A})$ regardless whether the vertex joins cluster C_1, joins any other cluster or forms a new cluster. Consider the clustering $\mathbf{C} = \{v_1, V \setminus \{v_1\}\}$. We have $OPT \geq w(\mathbf{C}) = x \cdot W$. Therefore the competitive ratio of $\mathcal{A}$ is at least: $\frac{OPT}{\mathcal{A}} \geq x \cdot W = W \cdot (n-2)$. □

We now consider the following greedy algorithm. Upon presentation of a vertex v_i, algorithm Greedy adds it to the cluster C_j that brings the maximum positive increase in the weight of the current clustering. If no cluster brings a positive increase in the weight, Greedy creates a new cluster $\{v_i\}$.

Theorem 3.2. *Greedy is strictly $(W \cdot (n-1))$-competitive.*

Proof. First, we show that every cluster returned by Greedy is connected in G^+. A newly created cluster that consists of a single vertex is trivially connected. On the other hand, whenever a vertex v_i is added to an existing cluster C_j, since the weight of the cluster increases, there is at least one positive-weighted edge in $\delta_{C_j}(v_i)$. Therefore, C_j remains connected in G^+. Let c_i be the number of clusters of Greedy with i vertices. since the smallest weight is 1, and a connected graph on i vertices has at least $i-1$ edges we have that Greedy $\geq \sum_{i=1}^{n}(i-1)c_i$.

Let I be the set of isolated vertices of Greedy. It is easy to see that I is an independent set of G^+. Indeed, otherwise there are two vertices $v_i, v_j \in I$ with $j > i$, adjacent in G^+. In this case v_j would be added to cluster of v_i by Greedy. The second observation is that for every vertex $v \in I$ and every cluster C the number of positive-weighted edges between v and C is at most $|C|-1$. Indeed, otherwise v is adjacent to every vertex of C in G^+. If v arrives after C is created then v is added to C by Greedy. Otherwise, v arrives before the first vertex u of C in which case u should be added to the cluster $\{v\}$ by Greedy. Clearly, the number of non-isolated vertices of Greedy is $n - |I| = \sum_{i=2}^{n} i \cdot c_i$. We have that $2 \cdot \text{Greedy} \geq 2\sum_{i=1}^{n}(i-1)c_i \geq \sum_{i=2}^{n} i \cdot c_i = n - |I|$. Since I is an independent set of G^+ and every vertex of I is adjacent to at most $i-1$ vertices of every cluster with i vertices, we get:

$$\begin{aligned} OPT &\leq W \cdot \left|E^+\right| \\ &\leq W \cdot \left(\binom{n-|I|}{2} + |I| \cdot \sum_{i=2}^{n}(i-1)c_i\right) \\ &\leq W \cdot \binom{n-|I|}{2} + W \cdot |I| \cdot \text{Greedy}. \end{aligned}$$

Therefore, $\frac{OPT}{W \cdot \text{Greedy}} \leq |I| + \frac{(n-|I|)(n-|I|-1)}{2 \cdot \text{Greedy}} \leq |I| + (n - |I| - 1) = n - 1$. □

3.1.2 Bounded Number of Clusters.

In this section we present two impossibility results for the case where the number of clusters is bounded by some $k \geq 2$, the case of $k = 1$ being trivial. In the following result, the adversary releases an independent set of at most $k+1$ vertices until two vertices v_i, v_j are put together, and then one vertex only adjacent to v_i and v_j, with edges of weights 1 and -1 respectively.

Theorem 3.3. *No deterministic algorithm is strictly competitive for MaxWC for any $k \geq 2$, even when $W = 1$.*

Proof. Suppose that there is c-competitive algorithm $\mathcal{A}$ for MaxWC. The adversary releases an independent set of at most $k+1$ vertices until $\mathcal{A}$ puts two vertices v_i, v_j in the same cluster. Then it releases a vertex adjacent to only v_i and v_j with edges of weights 1 and -1 respectively. Then, regardless of the decisions of $\mathcal{A}$, we have $\mathcal{A} = 0$. Moreover, one can form a cluster consisting of v_i and the last vertex. Therefore $\mathcal{A} = 0 < \frac{OPT}{c}$, a contradiction. □

The next result is obtained by exploiting a polynomial reduction from the k-colorability problem, in which given an unweighted and undirected graph G' and k colors, the answer is yes if and only if it is possible to find a mapping of all vertices of G' to colors $\{1, \ldots, k\}$ such that for any edge of G' the colors associated to its endpoints are different.

Theorem 3.4. *The offline variant of the problem MaxWC is inapproximable for any $k \geq 3$, unless $P = NP$.*

Proof. Given an instance G' of k-colorability, we construct the following edge-weighted graph G. G is complete graph on the same vertex set as G'. The weight of an edge e of G is 1 if e is a non-edge of G' and $-|E(G')|$ otherwise. If $k = n$ the instance is clearly a YES instance. Therefore, we assume $k < n$. To conclude the proof we show that G' is k-colorable if and only if $OPT > 0$.

Suppose that G' is k-colorable. Then its vertex set can be partitioned into $k' \leq k$ independent sets that induces a clustering $\mathbf{C}$ with k' clusters. The weight of an independent set I of G is $w(I) = \binom{|I|}{2} \geq \frac{|I|-1}{2}$. Therefore, $OPT \geq w(\mathbf{C}) \geq \frac{n-k'}{2} > 0$. Conversely, suppose that $OPT > 0$. Then, there is a clustering $\mathbf{C}$ of G with $w(\mathbf{C}) > 0$ and $|\mathbf{C}| \leq k$. We claim that every cluster of $\mathbf{C}$ is an independent set of G'. Suppose that **C** contains a cluster C that is not an independent set. Then $G[C]$ contains an edge of weight $-|E(G')|$. Since $w(E^+) = |E(G')|$ we conclude that $w(\mathbf{C}) \leq 0$. □

3.2 Bounded Cluster Size

When both the size of a cluster and the number of clusters is bounded, the size of the instance becomes bounded in which case every algorithm is 1-competitive. Therefore, in this section we assume that the number of clusters is unbounded. Since in this case GrandCoalition is not necessarily a feasible solution, the case of positive weights is not trivial. We analyze the cases of general weights and positive weights in two different sections.

In this section we consider the variant of Greedy that does not consider clusters of size α as possible clusters for the presented vertex.

3.2.1 General Weights. We show that Greedy is an optimal deterministic online algorithm for MaxWC. We start with the lower bound.

Theorem 3.5. *The strict competitive ratio of every deterministic online algorithm for MaxWC is at least* $2W \cdot (\alpha - 1)$.

We proceed with the analysis of Greedy. We denote by c_i the number of clusters with exactly i vertices in a given clustering returned by Greedy. The following is a technical lemma.

Lemma 3.6.
(1) *The set of isolated vertices of Greedy is an independent set of* G^+.
(2) *If all the weights are positive,* G^+ *contains an independent set that intersects every component of Greedy with less than* α *vertices.*
(3) *Greedy* $\geq \sum_{i=1}^{\alpha}(i-1)c_i$.

Proof.
(1) Suppose that Greedy contains two isolated vertices $\{v_i\}$ and $\{v_j\}$ with $w_{v_i,v_j} > 0$, and without loss of generality $i < j$. Then, when v_j is presented to Greedy it would be added to $\{v_i\}$ contradicting the fact that v_j is an isolated vertex of Greedy.
(2) Consider the set I consisting of the first vertex of every cluster of Greedy with less than α vertices. Clearly, I intersects every cluster of size less than α. It remains to show that I is an independent set of G^+. Suppose, for a contradiction, that there are two vertices $v_i, v_j \in I$ with $i < j$ and $w_{v_i,v_j} > 0$. When v_j is presented to Greedy the option of adding v_j to the cluster of v_i brings an increase of at least $w_{v_i,v_j} > 0$ since all the edges have positive weights. This contradicts the fact that v_j is the first vertex of its cluster.
(3) Whenever a vertex is added to an existing cluster it increases the weight of the clustering by at least 1.

□

Theorem 3.7. *Greedy is a strictly* $(2W \cdot (\alpha - 1))$*-competitive deterministic online algorithm for MaxWC.*

Proof. Let I be the set of isolated vertices of Greedy. Clearly, $n - |I| = \sum_{i=2}^{\alpha} i \cdot c_i$. Combining with Lemma 3.6 (3.) we have $2 \cdot \text{Greedy} \geq n - |I|$. By Lemma 3.6, I constitute an independent set of G^+. Therefore, every edge of G^+ is incident to at least one of the $n - |I|$ other vertices. Every such vertex has degree at most $\alpha - 1$ in every solution. Therefore, $OPT \leq W \cdot (n - |I|)(\alpha - 1)$. Then, the strict competitive ratio of Greedy is at most: $\frac{W \cdot (n-|I|)(\alpha-1)}{\text{Greedy}} \leq 2W \cdot (\alpha - 1)$. □

3.2.2 Positive Weights. We observe that the proof of Theorem 3.5 is not valid in this case, since the adversary uses negative edges. In this section we show that the lower bound of Theorem 3.5 does not hold in this case, and that Greedy is almost optimal.

Theorem 3.8. *The strict competitive ratio of every deterministic online algorithm for MaxWC is at least* $\frac{W \cdot (\alpha-1)}{o(W)}$ *even when all weights are positive.*

Proof. Let $\mathcal{A}$ be a strictly c-competitive deterministic online algorithm for MaxWC. Consider the online input that is supplied to $\mathcal{A}$ by the following adversary. The adversary releases a sufficiently big independent set of vertices until $\mathcal{A}$ forms either i) a cluster of size α, or ii) $\alpha - 1$ clusters. In case i), let C_1 be the cluster of α vertices formed by $\mathcal{A}$. Then, the adversary releases another vertex u incident only to some node $v \in C_1$. We have that $OPT = 1$ and $\mathcal{A} = 0$, because $\mathcal{A}$ cannot add v to C_1, thus the strict competitive ratio of $\mathcal{A}$ is unbounded. In case ii), there are $\alpha - 1$ clusters $C_1, \ldots, C_{\alpha-1}$ each of which is an independent set of at most $\alpha - 1$ vertices. Let v_i be an arbitrary vertex of C_i, for every $i \in [\alpha - 1]$. The adversary releases additional vertices $u_1, u_2, \ldots$ until $\mathcal{A}$ creates a new cluster (which must happen at some step $j \leq (\alpha-1)^2+1$). Every vertex u_j is adjacent to the vertices $v_1, \ldots, v_{\alpha-1}$, and $w_{u_j,v_1} = \cdots = w_{u_j,v_{\alpha-1}}$. Moreover, $w_{u_j,v_1} \gg w_{u_{j-1},v_1}$. At this point, by setting $W = w_{u_j,v_1}$, we have that $OPT \geq W \cdot (\alpha - 1)$ and $\mathcal{A} = o(W)$. Therefore, the competitive ratio of $\mathcal{A}$ is $\frac{OPT}{\mathcal{A}} \geq \frac{W \cdot (\alpha-1)}{o(W)}$. □

The proof of the following theorem exploits Lemma 3.6, and it is a bit more involved than the proof of Theorem 3.7.

Theorem 3.9. *The strict competitive ratio of Greedy is* $(\alpha \cdot W)$ *when all the weights are positive.*

Proof. By Lemma 3.6, G^+ contains an independent set I of size $\sum_{i=1}^{\alpha-1} c_i$. Clearly, $n = \sum_{i=1}^{\alpha} i \cdot c_i$, thus $n - |I| = \sum_{i=1}^{\alpha-1}(i-1)c_i + \alpha c_\alpha$. Every edge of G^+ is incident to at least one of the remaining $n - |I|$ vertices of Greedy. Every such vertex has degree at most $\alpha - 1$ in every solution. Therefore,

$$\begin{aligned} OPT &\leq W \cdot (\alpha - 1)(n - |I|) \\ &= W \cdot (\alpha - 1)\left(\sum_{i=1}^{\alpha-1}(i-1)c_i + \alpha c_\alpha\right) \\ &\leq W \cdot (\alpha - 1)(\text{Greedy} + c_\alpha). \end{aligned}$$

Clearly, Greedy $\geq (\alpha - 1)c_\alpha$. Then

$$OPT \leq W \cdot (\alpha - 1)\left(\text{Greedy} + \frac{\text{Greedy}}{\alpha - 1}\right) =$$

$$\alpha \cdot W \cdot \text{Greedy}.$$

We now show an example showing that the competitive ratio of Greedy is at least $\alpha \cdot W$. Let G be the following graph on α^2 vertices. $v_1, v_2, \cdots, v_\alpha$ is a path each edge of which has weight 1. Greedy will put all these vertices in one cluster C_1 with $w(C_1) = \alpha - 1$. The rest of the input is an independent set I. Since C_1 has already α vertices, every other vertex will be isolated in Greedy. Therefore, we have GREEDY $= \alpha - 1$. The vertices of I are grouped into α groups of $\alpha - 1$ vertices and every vertex v of group i is adjacent to v_i with an edge of weight W. A possible solution consists of α stars each of which is centered at one of the vertices $v_1, \ldots, v_\alpha$ and has $\alpha - 1$ leaves from I. Therefore, $OPT \leq W \cdot \alpha(\alpha - 1) = W \cdot \alpha \cdot$ GREEDY. □

4 MAXIMUM FRACTIONAL WEIGHT CLUSTERING

In this section our objective is to maximize the fractional weight of the clustering. We note that, as opposed to problem MAXWC, for non-negative weights, GRANDCOALITION is not necessarily an optimal solution even when cluster size is unbounded. We start with a general lower bound and then analyze the cases of unbounded and bounded cluster sizes separately.

THEOREM 4.1. *No deterministic online algorithm for MAXFWC is strictly* $(4W \cdot (1 - 1/\alpha))$*-competitive.*

4.1 Unbounded Cluster Size

4.1.1 General Weights. In this section we analyze the MAXFWC problem when the cluster size is unbounded. For this case the following result is known.

THEOREM 4.2. *[Theorem 5 in: [4]] Any maximal matching is a* $4W$*-approximation.*

A maximal matching is a clustering in which every cluster is connected and consists of at most two vertices. Moreover, for any pair of non-matched vertices, they are not connected by a positive weight edge. A maximal matching can be computed online by the following algorithm that we name as MAXIMALMATCHING. Whenever a vertex v_i is presented it is added to an existing cluster of size one that is adjacent to v_i by means of a positive weight edge. If no such cluster exists, a new cluster $\{v_i\}$ is created.

We note that MAXIMALMATCHING is an optimal algorithm for this case because Theorem 4.1 implies a matching lower bound of $4W$. We observe that the adversary in the proof of Theorem 4.1 uses edges with negative weights. Therefore, it makes sense to consider the case of positive weights. We start with the analysis of unweighted graphs and in which then proceed with the case of general positive weights.

4.1.2 Unweighted Graphs. Since MAXIMALMATCHING is $4W$ competitive in general, it is clearly 4-competitive for unweighted graphs. It is possible to show that this is the best possible for deterministic algorithms in this case.

THEOREM 4.3. *There is no* $(4 - \epsilon)$*-competitive deterministic online algorithm for MAXFWC even in unweighted graphs, for any* $\epsilon > 0$.

PROOF. Consult Figure 1 for the following discussion. Let $\mathcal{A}$ be a $(4 - \epsilon)$-competitive deterministic online algorithm for some $\epsilon > 0$. The adversary first releases two adjacent vertices v_1, v_2 that must be taken to the same cluster, say C_1, by $\mathcal{A}$, since otherwise $\mathcal{A}$ is not competitive. Then the adversary releases a sequence of vertices $v_3, v_4, \ldots$ where the vertices with odd (resp. even) indices are adjacent only to v_1 (resp. v_2). This sequence ends when one of the vertices is added to C_1 by $\mathcal{A}$. We show that this must eventually happen. Indeed, suppose that none of the vertices $v_3, \ldots, v_{2i}$ are added to C_1. Then, $\mathcal{A} = w_F(C_1) = 1/2$ and $OPT \geq 2(i-1)/i$ which is the fractional weight of the solution that divides V into two clusters, namely the cluster of odd vertices and the cluster of even vertices, each of which is a star on i vertices. Then $OPT/\mathcal{A} \geq 4(1 - 1/i)$, and for sufficiently big i we have $OPT/\mathcal{A} \geq 4 - \epsilon$, a contradiction.

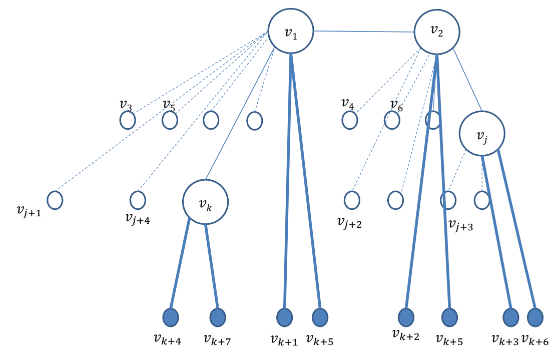

Figure 1: Lower bound of $4 - \epsilon$ for the unweighted case. The edges drawn by thin solid lines are taken by $\mathcal{A}$ and the thick solid edges form four stars that is used to prove the lower bound if the adversary reaches the last stage.

Let v_j be the first vertex (after v_1 and v_2) added to C_1 by $\mathcal{A}$. At his stage the adversary releases a sequence of vertices $v_{j+1}, v_{j+2}, \ldots$ where v_{j+1} is adjacent to v_1, v_{j+2} is adjacent to v_2, v_{j+3} is adjacent to v_j, v_{j+4} is adjacent to v_1, and so on. This continues until one of these vertices is added to C_1 by $\mathcal{A}$. If this does not happen during a sequence of 24 vertices, we have $\mathcal{A} = 2/3$, and $OPT \geq 3 \cdot 8/9 = 8/3$ since there is a solution consisting of three clusters each of which is a star on 9 vertices. Then $OPT/\mathcal{A} \geq 4 > 4 - \epsilon$, a contradiction. Therefore, this sequence ends after at most 24 additional vertices are released.

At this stage a vertex v_k is added to C_1. Now the adversary releases a final sequence of i' vertices $v_{k+1}, v_{k+2}, \cdots$ where each vertex is adjacent exactly one of the four vertices v_1, v_2, v_j, v_k in a round robin manner and then stops. In this case the best option for $\mathcal{A}$ is to add every new vertex to C_1, since this is the only option that increases the fractional weight of the solution which consists of a single tree whose fractional weight is at most one and isolated vertices. Therefore, $\mathcal{A} < 1$. On the other hand an optimal solution consists of four stars centered at v_1, v_2, v_j and v_k for a total fractional weight of $4(1 - 1/i')$ which is bigger than $4 - \epsilon$ for a sufficiently big i'. □

4.1.3 Positive Weights. In the previous sections we have shown that MAXIMALMATCHING is an optimal algorithm for the general case and also for the unweighted case. In this section we show that quite surprisingly this is not the case for positive weights. We present an $O(\log^2 W)$-competitive algorithm and also a matching lower bound of $\Omega(\log^2 W)$.

Our algorithm partitions the edges into classes according to their weights. We denote the *class* of an edge e by $class(e)$ and it is equal to the smallest integer i such that $w(e) < 2^i$. We note that

$class(e) > 0$. The class of a cluster C_i is denoted by $class(C_i)$ and is equal to the class of its heaviest edge. Upon presentation of a vertex v_i, Algorithm Classify considers the edges incident to v_i in descending order and adds v_i to a cluster whose class is lower than the edge under consideration. If this is not possible, it creates a new cluster (See Algorithm 1).

Algorithm 1 Classify

```
Initialization:
C ← ∅.

When vertex v_i arrives:
for all edge e = v_i v_j in descending order of w(e) do
    if class(e) > class(C(v_j)) then
        Add v_i to the cluster C(v_j)
        return
Create a new cluster {v_i}.
```

Theorem 4.4. *Classify is strictly* $(O(\min\{n, 1 + \log W\})^2)$*-competitive.*

Proof. Consider an optimal clustering OPT, and a clustering $\mathbf{C} = \{C_1, C_2, \ldots, C_c\}$ returned by Classify. Denote by OPT_{EXT} (resp. OPT_{INT}) the set of edges whose endpoints fall within a same cluster of OPT, i.e., contribute to $w_F(OPT)$, but within different clusters (resp. a same cluster) of $\mathbf{C}$. We denote by OPT_{EXT} and OPT_{INT} also the contribution of these edges to $w_F(OPT)$. Clearly, $OPT = OPT_{EXT} + OPT_{INT}$. In the sequel we upper bound each of these values.

Upper bounding OPT_{EXT}: We exploit the following property: For every edge $vv' \in OPT_{EXT}$ there exists a cluster $\mathbf{C}(vv') \in \{\mathbf{C}(v), \mathbf{C}(v')\}$ such that $class(\mathbf{C}(vv')) \geq class(vv')$.

In fact, let $vv' \in OPT_{EXT}$, $C = \mathbf{C}(v)$ and $C' = \mathbf{C}(v')$. Assume without loss of generality that v' appears before v in the input. If v is the first vertex of C, since v is not added to C' by Classify we conclude that $class(C') \geq class(vv')$ and we are done. Otherwise, v is not the first vertex of C thus there exists an edge e incident to v that caused Classify to add v to C. If $w(e) \geq w_{v,v'}$ we have $class(C) \geq class(e) \geq class(vv')$ and we are done. Otherwise, $w_{v,v'} > w(e)$ thus vv' was considered before e by Classify and v was not added to C'. Therefore, $class(C') \geq class(vv')$.

Consider a cluster $C_j \in \mathbf{C}$, and let OPT_{EXT,C_j} be the set of edges $e \in OPT_{EXT}$ such that $\mathbf{C}(e) = C_j$. Since C_j contains an edge of weight at least $2^{class(C_j)-1}$, the contribution of the edges in C_j to the Classify is

$$w_F(C_j) \geq \frac{2^{class(C_j)-1}}{|C_j|}. \tag{1}$$

Consider a vertex $v \in C_j$, the set $OPT_{EXT,v}$ of edges of OPT_{EXT,C_j} incident to v, and let $a = |OPT_{EXT,v}|$. Let also h be the class of the heaviest edge of $OPT_{EXT,v}$. Clearly, the edges of $OPT_{EXT,v}$ are in the same cluster of OPT, that contains at least $a+1$ vertices. Therefore, the contribution of the edges in $OPT_{EXT,v}$ to OPT is less than $\frac{a \cdot 2^h}{a+1} < 2^h \leq 2^{class(C_j)}$. Summing up for all vertices $v \in C_j$ and using (1) we get

$$\begin{aligned} OPT_{EXT,C_j} &< |C_j| \cdot 2^{class(C_j)} \leq 2|C_j|^2 w_F(C_j) \\ &\leq 2\left(\max_{j\in[c]} |C_j|\right)^2 w_F(C_j). \end{aligned}$$

Finally, we sum up over all clusters C_j, and obtain

$$OPT_{EXT} < 2\left(\max_j |C_j|\right)^2 \cdot \text{Classify}.$$

Upper bounding OPT_{INT}: The contribution to OPT of the edges of OPT_{INT} that fall within some cluster C_j is at most half of the sum of weights of all edges of C_j, since every edge has to be in a cluster of at least two vertices. Therefore,

$$\begin{aligned} OPT_{IN,C_j} &\leq \sum_{e\in OPT_{INT,C_j}} \frac{w(e)}{2} \leq \sum_{e\in E(C_j)} \frac{w(e)}{2} \\ &\leq \frac{|C_j|}{2} \cdot w_F(C_j) \leq \frac{\max_{j\in[c]} |C_j|}{2} \cdot w_F(C_j), \end{aligned}$$

and summing up over all clusters we get:

$$OPT_{INT} \leq \frac{\max_{j\in[c]}|C_j|}{2} \cdot \text{Classify} \leq \frac{OPT_{EXT}}{4}.$$

Upper bounding OPT: Now we note that $|C_j| \leq class(C_j)$ by the way vertices are added to C_j by Classify. Therefore,

$$\max_{j\in[c]} |C_j| \leq \max_{j\in[c]} class(C_j) = \lceil \log W \rceil \leq 1 + \log W.$$

Clearly, $\max_{j\in[c]} |C_j| \leq n$. We conclude that

$$\begin{aligned} OPT &= OPT_{EXT} + OPT_{INT} \leq \frac{5}{4} OPT_{EXT} \\ &< \frac{5}{2}\left(\max_j |C_j|\right)^2 \cdot \text{Classify} \\ &\leq \frac{5}{2}\left(\min\{n, 1+\log W\}\right)^2 \cdot \text{Classify}. \end{aligned}$$

□

Theorem 4.6 provides a matching lower bound. In order to prove it, we need the following technical lemma whose proof will appear in the full version of this paper.

Lemma 4.5. *Given any integer k, there exists $h \geq k$ such that, for any sequence of non-negative integers $y_1, y_2, \ldots, y_k, \ldots, y_h$ with $y_1 = 1$ and $y_i \leq 2^{i-1}$ for any $i = 1, \ldots, h$,*

$$\frac{\sigma_h^2}{\alpha_h} \geq \frac{h^2}{2^{10}},$$

where $\sigma_h = \sum_{i=1}^h y_i$ and $\alpha_h = \sum_{i=1}^h \frac{y_i}{2^{h-i}}$, i.e., $\alpha_h = y_h + \frac{y_{h-1}}{2} + \frac{y_{h-2}}{4} + \ldots + \frac{y_1}{2^{h-1}}$.

Theorem 4.6. *The strict competitive ratio of any deterministic online algorithm for MaxFWC is $\Omega(\log^2 W)$ even when all weights are positive.*

Proof. Let $\mathcal{A}$ be any deterministic online algorithm for MaxFWC. The adversary works in phases $i = 1, 2, \ldots$. In phase 1, she releases a vertex v_1. The adversary is able to maintain the following invariant: at the end of each phase, the solution of $\mathcal{A}$ contains a single component C_1 in $\mathbf{C}$. In fact, let σ_i be the number of vertex in the (unique) connected component built by $\mathcal{A}$ at the end of phase i, and let $v_1, v_2, \ldots, v_{q_i}$ be the nodes belonging to this component. Clearly, $\sigma_1 = 1$. The adversary releases, in phase i (for

$i = 2, 3, \ldots$), σ_{i-1} vertices $u_1, u_2, \ldots, u_{\sigma_{i-1}}$, such that every u_j (for $j = 1, \ldots, \sigma_{i-1}$) is adjacent to v_j by an edge of weight 2^{i-2}. Let y_i the number of nodes that $\mathcal{A}$ adds to the unique component in phase i. It follows that $\sigma_i = \sigma_{i-1} + y_i$ and $y_i \leq \sigma_i \leq 2^{i-1}$. First of all, notice that $\mathcal{A}$ cannot stop adding new nodes to the unique component, because otherwise it would be not competitive, given that the weights of edges are geometrically increasing. Given any integer k', let $k \geq k'$ the first phase after phase k' such that $y_k \geq 1$.

Clearly, $\sigma_i = \sum_{j=1}^{i} y_j$. It can be easily checked that, after phase i, the measure of the solution computed by $\mathcal{A}$ is $w_F(\mathrm{C}_i) = \frac{\sum_{j=1}^{i} y_j 2^{j-2}}{\sigma_i}$. Moreover, another clustering $\mathrm{C}' = (C_1, \ldots, C_{\sigma_{i-1}})$ exists in which each edge added by the adversary in phase i constitutes a separate cluster. This solution has measure $w_F(\mathrm{C}') = \frac{\sigma_{i-1} 2^{i-2}}{2}$. Therefore, the measure of an optimal solution C_i^* is $w_F(\mathrm{C}_i^*) \geq \frac{\sigma_{i-1} 2^{i-2}}{2}$. Given that (for any $i = 1, 2, \ldots$) $\sigma_{i+1} \leq 2\sigma_i$, we obtain $w_F(\mathrm{C}_i^*) \geq \frac{\sigma_i 2^{i-1}}{8}$. Thus, we obtain that the competitive ratio of $\mathcal{A}$ at the end of phase i is at least $\frac{w_F(\mathrm{C}_i^*)}{w_F(\mathrm{C}_i)} = \frac{\sigma_i^2 2^{i-1}}{8 \sum_{j=1}^{i} y_j 2^{j-2}} = \frac{\sigma_i^2 2^{i-1}}{4 \sum_{j=1}^{i} y_j 2^{j-1}} = \frac{\sigma_i^2}{4 \sum_{j=1}^{k} y_j 2^{j-i}} = \frac{\sigma_i^2}{4\alpha_i}$, where $\alpha_i = \sum_{j=1}^{i} \frac{y_j}{2^{i-j}}$. Since any phase i the maximum edge weight is $W = 2^{i-2}$, it holds that $i = \Omega(\log W)$.

By Lemma 4.5, we obtain that, for any integer k, there exists $h \geq k$ such that $\frac{w_F(\mathrm{C}_h^*)}{w_F(\mathrm{C}_h)} \geq \frac{h^2}{2^{12}} = \Omega(\log^2 W)$. Therefore, the claim follows by letting the adversary continue until phase h. □

4.2 Bounded Cluster Size

4.2.1 Unweighted Graphs. For the case of unweighted graphs, we are able to prove that algorithm MaximalMatching provides the best possible competitive ratio of $4\frac{\alpha-1}{\alpha}$.

Theorem 4.7 proves the lower bound. The proof is very similar to the one of Theorem 4.3.

Theorem 4.7. *There is no $(4\frac{\alpha-1}{\alpha} - \epsilon)$-competitive deterministic online algorithm for MaxFWC even in unweighted graphs, for any $\epsilon > 0$.*

The following theorem provides a matching upper bound to Theorem 4.7. Its proof exploits and refines arguments introduced in the proof of Lemma 1 in [11].

Theorem 4.8. *Algorithm MaximalMatching is a $(4\frac{\alpha-1}{\alpha})$-competitive deterministic online algorithm for MaxFWC in unweighted graphs.*

4.2.2 Weighted Graphs. From Theorem 4.2, we know that, for any $\alpha \geq 2$, MaximalMatching is strictly $4W$-competitive for general weights. It is not difficult to show that this bound is asymptotically tight, even when all the weights are positive.

Theorem 4.9. *The strict competitive ratio of any deterministic online algorithm for MaxFWC is $\Omega(W)$ even when all weights are positive.*

Proof. Let $\mathcal{A}$ be a deterministic online algorithm for MaxFWC. Consider the online input that is supplied to $\mathcal{A}$ by the following adversary. The adversary releases a star $v_1, v_2, \ldots$ centered at v_1 until $\mathcal{A}$ creates its second cluster $C_2 = \{v_j\}$. The weights are such that $w_{v_{j+1}, v_1} \gg w_{v_j, v_1}$ for every $j > 1$. By setting $W = w_{v_j, v_1}$, we have that $OPT \geq \frac{W}{2}$ and $\mathcal{A} = o(W)$. Therefore, the competitive ratio of $\mathcal{A}$ is at least $\frac{OPT}{\mathcal{A}} \geq \frac{W/2}{o(W)} = \Omega(W)$. □

4.3 Bounded Number of Clusters

In this section we consider the case where the number of clusters is bounded by some $k \geq 2$, the case of $k = 1$ being trivial.

4.3.1 General Weights. By exploiting a proof similar to the one of Theorem 3.3, it is possible to prove the following result.

Theorem 4.10. *No deterministic algorithm is strictly competitive for MaxFWC for any $k \geq 2$, even when $W = 1$.*

4.3.2 Positive Weights.

Observation 1. *For positive weights, GrandCoalition is $\frac{n}{2}$-competitive. In fact, GrandCoalition $= \frac{\sum_{e \in E(G)} w(e)}{n}$, and $OPT \leq \frac{\sum_{e \in E(G)} w(e)}{2}$, since the weight of any edge is shared by at least its two endpoints.*

Theorem 4.11. *The strict competitive ratio of every deterministic online algorithm for MaxFWC for any $k \geq 2$, is at least $\frac{n}{2} \frac{W}{W + o(W)}$, when all weights are positive.*

Proof. Let $\mathcal{A}$ be a deterministic online algorithm for MaxFWC. Consider the online input that is supplied to $\mathcal{A}$ by the following adversary. The adversary releases a star $v_1, v_2, \ldots$ centered at v_1. The weights are such that $w_{v_{j+1}, v_1} \gg w_{v_j, v_1}$, for every $j > 1$. Let us call C_1 the cluster where algorithm $\mathcal{A}$ puts vertex v_1. If at some step j, $\mathcal{A}$ puts the vertex v_j into a different cluster than C_1, then the adversary stops. In this case, by setting $W = w_{v_j, v_1}$, we have that $OPT \geq \frac{W}{2}$ and $\mathcal{A} \leq \frac{o(W)}{n-1}$, and the theorem holds. However, if algorithm $\mathcal{A}$ puts all the vertices into cluster C_1, the competitive ratio of $\mathcal{A}$ is at least $\frac{OPT}{\mathcal{A}} \geq \frac{n}{2} \frac{W}{W + o(W)}$. □

5 CONCLUSION AND OPEN PROBLEMS

We studied the online version of the online coalition structure generation problem on edge-weighted graphs, considering two different utility functions for the coalition profit.

We point out that our lower bounds hold also for general (i.e. non-strict competitive ratio). All the related proofs will appear in the full version of the paper.

Basic extensions and open problems include the following: (1) more involved profit functions can be considered, depending on specific applications. One might also consider costs associated with nodes, or taking into account the topological properties of the subgraphs induced by the clusters, such as the diameter, the average distance between the nodes, measures depending on the centrality indices in social networks, etc. (recent papers considering such measures are [7, 8]). (2) In this paper we considered the classic online setting where the decision about the cluster where to allocate the next agent is irrevocable. It is worth to extend this research to the case where the clustering can be modified by migrating nodes from cluster to cluster by paying some penalty (as considered in [1]), or the case where there is a bound on the number of possible migrations. (3) Finally, it would be interesting to understand whether randomized algorithms can achieve significantly better performance than deterministic ones.

REFERENCES

- [1] J. Augustine, C. Avin, M. Liaee, G. Pandurangan, and R. Rajaraman. 2016. Information Spreading in Dynamic Networks Under Oblivious Adversaries. In *30th International Symposium on Distributed Computing, (DISC)*. 399–413.
- [2] H. Aziz, F. Brandt, and P. Harrenstein. 2014. Fractional hedonic games. In *13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. 5–12.
- [3] H. Aziz, F. Brandt, and H.G. Seedig. 2011. Stable partitions in additively separable hedonic games. In *Tenth International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. 183–190.
- [4] H. Aziz, S. Gaspers, J. Gudmundsson, J. Mestre, and H. Taubig. 2015. Welfare Maximization in Fractional Hedonic Games. In *25th International Joint Conference on Artificial Intelligence (IJCAI)*. 461–467.
- [5] Y. Bachrach, P. Kohli, V. Kolmogorov, and M. Zadimoghaddam. 2013. Optimal Coalition Structure Generation in Cooperative Graph Games. In *27th AAAI Conference on Artificial Intelligence (AAAI)*. 81–87.
- [6] C. Ballester. 2004. NP-completeness in hedonic games. *Games and Economic Behavior* 49(1) (2004), 1–30.
- [7] A. Balliu, M. Flammini, G. Melideo, and D. Olivetti. 2017. Nash Stability in Social Distance Games. In *Thirty-First AAAI Conference on Artificial Intelligence (AAAI)*. 324–348.
- [8] A. Balliu, M. Flammini, and D. Olivetti. 2017. On Pareto Optimality in Social Distance Games. In *Thirty-First AAAI Conference on Artificial Intelligence (AAAI)*. 349–355.
- [9] S. Banerjee, H. Konishi, and T. Sönmez. 2001. Core in a simple coalition formation game. *Social Choice and Welfare* 18 (2001), 135–153.
- [10] V. Biló, A. Fanelli, M. Flammini, G. Monaco, and L. Moscardelli. 2014. Nash Stability in Fractional Hedonic Games. In *10th Conference on Web and Internet Economics (WINE)*. 486–491.
- [11] V. Biló, A. Fanelli, M. Flammini, G. Monaco, and L. Moscardelli. 2015. On the Price of Stability of Fractional Hedonic Games. In *14th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*. 1239–1247.
- [12] A. Bogomolnaia and M. O. Jackson. 2002. The Stability of Hedonic Coalition Structures. *Games and Economic Behavior* 38(2) (2002), 201–230.
- [13] A. Borodin and R. El-Yaniv. 1998. *Online Computation and Competitive Analysis*. Cambridge University Press.
- [14] F. Brandl, F. Brandt, and M. Strobel. 2015. Fractional hedonic games: individual and group stability. In *14th International Conference on Autonomous Agents and Multiagent Systems, (AAMAS)*. 1219–1227.
- [15] X. Deng and C. H. Papadimitriou. 1994. On the complexity of cooperative solution concepts. *Mathematics of Operations Research* 12 (1994), 257–266.
- [16] J. H. Dréze and J. Greenbergd. 1980. Hedonic coalitions: optimality and stability. *Econometrica* 48(4) (1980), 987–1003.
- [17] Michele Flammini, Gianpiero Monaco, and Qiang Zhang. 2017. Strategyproof Mechanisms for Additively Separable Hedonic Games and Fractional Hedonic Games. In *Proceedings of the 15th Workshop on Approximation and Online Algorithms (WAOA)*.
- [18] Gianpiero Monaco, Luca Moscardelli, and Yllka Velaj. 2018. Stable Outcomes in Modified Fractional Hedonic Games. In *Proceedings of the Seventeenth International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.
- [19] M. Olsen. 2009. Nash stability in additively separable hedonic games and community structures. *Theory of Computing Systems* 45(4) (2009), 917–925.
- [20] D. Peters and E. Elkind. 2015. Simple Causes of Complexity in Hedonic Games. In *24th International Joint Conference on Artificial Intelligence (IJCAI)*. 617–623.
- [21] T. Rahwan and N. R. Jennings. 2008. An improved dynamic programming algorithm for coalition structure generation. In *7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. 1417–1420.
- [22] T. Rahwan, T. P. Michalak, M. Wooldridge, and N. R. Jennings. 2012. Anytime coalition structure generation in multi-agent systems with positive or negative externalities. *Artificial Intelligence* 186 (2012), 95–122.
- [23] T. Rahwan, T. P. Michalak, M. Wooldridge, and N. R. Jennings. 2015. Coalition structure generation: A survey. *Artificial Intelligence* 229 (2015), 139–174.
- [24] T. Rahwan, S. D. Ramchurn, N. R. Jennings, and A. Giovannucci. 2009. An Anytime Algorithm for Optimal Coalition Structure Generation. *Journal of Artificial Intelligence Research* 34 (2009), 521–567.
- [25] T. Sandholm, K.Larson, M. Andersson, O. Shehory, and F. Tohmé. 1999. Coalition structure generation with worst case guarantees. *Artificial Intelligence* 111(1-2) (1999), 209–238.
- [26] O. Shehory and S. Kraus. 1998. Methods for task allocation via agent coalition formation. *Artificial Intelligence* 101(1-2) (1998), 165–200.
- [27] T. Voice, M. Polukarov, and N. R. Jennings. 2012. Coalition Structure Generation over Graphs. *Journal of Artificial Intelligence Research* 45 (2012), 165–196.
- [28] Mason Wright and Yevgeniy Vorobeychik. 2015. Mechanism Design for Team Formation. In *Proceedings of the Twenty-Ninth Conference on Artificial Intelligence (AAAI)*. 1050–1056.
- [29] Y. Zick, E. Markakis, and E. Elkind. 2014. Arbitration and Stability in Cooperative Games with Overlapping Coalitions. *Journal of Artificial Intelligence Research* 50 (2014), 847–884.