Coalition Formation with Bounded Coalition Size

Chaya Levinger
Ariel University
Ariel, Israel
chayal@ariel.ac.il

Noam Hazon
Ariel University
Ariel, Israel
noamh@ariel.ac.il

Sofia Simola
Technische Universität Wien
Vienna, Austria
sofia.simola@tuwien.ac.at

Amos Azaria
Ariel University
Ariel, Israel
amos.azaria@ariel.ac.il

ABSTRACT

In many situations when people are assigned to coalitions, the utility of each person depends on the friends in her coalition. Additionally, in many situations, the size of each coalition should be bounded. This paper studies such coalition formation scenarios in both weighted and unweighted settings. Since finding a partition that maximizes the utilitarian social welfare is computationally hard, we provide a polynomial-time approximation algorithm. We also investigate the existence and the complexity of finding stable partitions. Namely, we show that the Contractual Strict Core (CSC) is never empty, but the Strict Core (SC) of some games is empty. Finding partitions that are in the CSC is computationally easy, but even deciding whether an SC of a given game exists is NP-hard. The analysis of the core is more involved. In the unweighted setting, we show that when the coalition size is bounded by 3 the core is never empty, and we present a polynomial time algorithm for finding a member of the core. However, for the weighted setting, the core may be empty, and we prove that deciding whether there exists a core is NP-hard.

KEYWORDS

Coalition formation; Additively separable hedonic games; Stability

ACM Reference Format:
Chaya Levinger, Noam Hazon, Sofia Simola, and Amos Azaria. 2024. Coalition Formation with Bounded Coalition Size. In *Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024*, IFAAMAS, 9 pages.

1 INTRODUCTION

Suppose that a group of travelers, located at some origin, would like to reach the same destination, and later return. Each of the travelers has her own vehicle; but each traveler has a preference related to who will be with her in the vehicle. Namely, each traveler would rather share a vehicle with as many as possible of her friends during the ride, and thus the utility of each traveler is the number of friends traveling with her. However, the vehicles have a limited capacity; this capacity can either be a physical constraint of the vehicles, or the maximal number of travelers willing to travel together. How should the travelers be assigned to vehicles in order to maximize the social welfare (the sum of all travelers' utilities)? Can the travelers be organized in such a way that no subgroup of travelers will want to leave their current group and join together? Similar questions arise when assigning students to dormitories, colleagues to office-rooms and workers to project teams. In these settings, it might be that the utility of each person does not depend only on the number of friends, but also on the intensity of friendship.

This set of problems falls within hedonic games [15], in which a set of agents are partitioned into coalitions, and the utility for each agent depends only on the coalition that she is a member of. Additively Separable Hedonic Games (ASHGs) [10] are a special type of hedonic games, in which each agent has a value for any other agent, and the utility she assigns to a coalition is the sum of the values she assigns to its members. In ASHGs there is usually no restriction on the number of agents that are allowed to belong to a coalition. However, in our group of travelers example, the vehicles have physical capacity, and thus, there is an upper bound on the size of each coalition. Despite this restriction being natural, it is scarcely studied in the domain of hedonic games.

In this paper, we study hedonic games with bounded coalition size. Specifically, we concentrate on symmetric ASHGs, in which the value an agent assigns to another agent is non-negative and it is equal to the value that the other agent assigns to her; we refer to these settings as the weighted settings. We also study simple symmetric ASHGs, in which the value an agent assigns to other agents is either 0 or 1; we refer to these settings as the unweighted settings. These models capture many situations, such as social and friendship relations. We begin by studying the problem of finding a partition that maximizes the utilitarian social welfare. Since this problem is computationally hard for any coalition size bound $k > 2$, even in the unweighted setting, we provide a polynomial-time approximation algorithm. We prove that in the unweighted setting, the algorithm has an approximation ratio of $\frac{1}{k-1}$. In the weighted setting, the algorithm has an approximation ratio of $\frac{1}{k}$ when k is odd, and $\frac{1}{k-1}$ when k is even.

We then study stability aspects of the problem. That is, we investigate the existence and the complexity of finding stable partitions. Namely, we show that the Contractual Strict Core (CSC) is never empty, but the Strict Core (SC) of some games is empty. Finding partitions that are in the CSC is computationally easy, but even deciding whether an SC of a given game exists is hard. The analysis of the core is more involved. In the unweighted setting, we show

Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand.

that for k=3 the core is never empty, and we present a polynomial time algorithm for finding a member of the core. For $k > 3$, it is unclear whether the core can be empty, and how to find a partition in the core. Indeed, we show in simulation over 100 million games that a simple heuristic always finds a partition that is in the core. For the weighted setting, the core may be empty even when $k = 3$, and we prove that for any $k \geq 3$, deciding whether there exists a partition in the core is NP-hard.

To summarize, the contribution of this work is a systematic study of additively separable hedonic games with bounded coalition size. Namely, we provide an approximation algorithm for maximizing the utilitarian social welfare and study the computational aspects of several stability concepts.

2 RELATED WORK

Dreze and Greenberg [15] initiated the study of hedonic games, in which the utility for each agent depends only on the coalition that she is a member of. Stability concepts of hedonic games were further analyzed in [6] and [11]. For more details, see the survey of Aziz et al. [2]. A special case is Additively Separable Hedonic Games (ASHGs) [10], in which each agent has a value for any other agent, and the utility she assigns to a coalition is the sum of the values she assigns to its members. The computational aspects of ASHGs are analyzed in [1, 3, 5, 8, 14, 23, 25]. None of these works imposed any restriction on the size of the coalitions.

Indeed, there are few papers that impose a restriction on the size of the coalitions. Wright and Vorobeychik [26] study a model of ASHG where there is an upper bound on the size of each coalition. Within their model, they propose a strategyproof mechanism that achieves good and fair experimental performance, despite not having a theoretical guarantee. Flammini et al. [17] study the online partition problem. Similar to our work, they also consider the scenario that the coalitions are bounded by some number. They consider two cases for the value of a coalition, the sum of the weights of its edges, which is similar to our work, and the sum of the weights of its edges divided by its size. However, in both cases they only consider the online version, i.e., the agents arrive sequentially and must be assigned to a coalition as they arrive. This assignment cannot be adapted later on, and must remain. They show that a simple greedy algorithm achieves an approximation ratio of $\frac{1}{k}$ when the value of the coalition is the sum of the weights. Cseh et al. [12] require the partition to be composed of exactly k coalitions, and also assume a predefined set of size constraints. Each coalition is required to exactly match its predefined size. They study the complexity of finding a Pareto optimal partition, as well as the complexity of deciding whether a given partition is Pareto optimal. Bilò et al. [9] consider the same settings as Cseh et al. Since classical stability notions are infeasible in their setting, they study three different types of swap stability, and analyze the existence, complexity, and efficiency of stable outcomes. Note that almost all other works analyzing ASHGs assume that an agent may assign a negative value to another agent. Otherwise, since they do not impose any restrictions on the coalition size, the game becomes trivial, as the grand coalition is always an optimal solution. We found two exceptions that restrict the value each agent assigns to other agents to be either 0 or 1. Namely, Sless et al. [24] study the setting in which the agents must be partitioned into exactly k coalitions, without any restriction on each coalition's size. Li et al. [21] study the setting in which the agents must be partitioned into exactly k coalitions that are almost equal in their size.

3 PRELIMINARIES

Let $V = \{v_1, ..., v_n\}$ be a set of agents, and let $G(V, E)$ be a weighted undirected graph representing the social relations between the agents. For every edge $e \in E$, the weight of the edge, $w(e)$, is positive. In the unweighted setting, all weights are set to 1. A k-bounded coalition is a coalition of size at most k. A k-bounded partition P is a partition of the agents into disjoint k-bounded coalitions. Given a coalition $S \in P$, and $v \in S$, let $N(v, S)$ be the set of immediate neighbors of $v \in V$ in S, i.e., $N(v, S) = \{u \in S : (v, u) \in E\}$. Let $W(v, S)$ be the sum of weights of immediate neighbors of $v \in V$ in S, i.e., $W(v, S) = \sum_{u \in N(v,S)} w((v, u))$. Note that in the unweighted setting, $W(v, S) = |N(v, S)|$. An *additively separable hedonic game with bounded coalition size* is a tuple (G, k), where for every k-bounded partition P, coalition $S \in P$, and $v \in S$, the agent v gets utility $W(v, S)$. We denote the utility of v given a k-bounded partition P, by $u(v, P)$. Given a tuple (G, k), the goal is to find a k-bounded partition P that satisfies efficiency or stability properties. We consider the following efficiency or stability concepts:

- The utilitarian social welfare of a partition P, denoted $u(P)$, is the sum of the utilities of the agents. That is, $u(P) = \sum_{v \in V} u(v, P)$. A *MaxUtil* k-bounded partition P is a partition with maximum $u(P)$.
- A k-bounded coalition S is said to *strongly block* a k-bounded partition P if for every $v \in S$, $W(v, S) > u(v, P)$. A k-bounded partition P is in the *Core* if it does not have any strongly blocking k-bounded coalitions.
- A k-bounded coalition S is said to *weakly block* a k-bounded partition P if for every $v \in S$, $W(v, S) \geq u(v, P)$, and there exists some $v \in S$ such that $W(v, S) > u(v, P)$. A k-bounded partition P is in the *Strict Core (SC)* if it does not have any weakly blocking k-bounded coalitions.
- Given a partition P and a set S, let P^{-S} be the partition when S breaks off. That is, $P^{-S} = \{S\} \cup \bigcup_{C \in P} \{C \setminus S\}$. A k-bounded partition P is in the *Contractual Strict Core (CSC)* if for any weakly blocking k-bounded coalition S, there exists at least one agent v such that $u(v, P^{-S}) < u(v, P)$.

4 EFFICIENCY

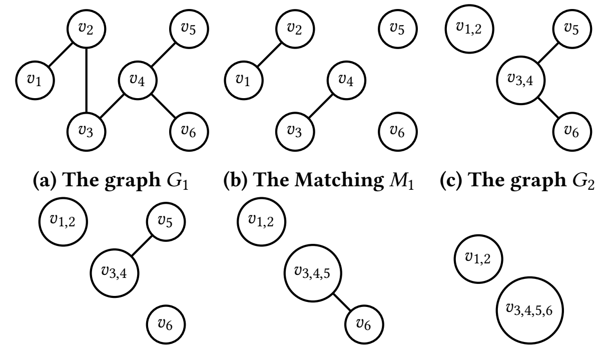
We begin with the elementary concept of efficiency, which is to maximize the utilitarian social welfare.

Definition 4.1 (MaxUtil problem). *Given a coalition size limit k and a graph G, find a MaxUtil k-bounded partition.*

Clearly, the decision variant of the MaxUtil problem is to decide whether there exists a k-bounded partition with a utilitarian social welfare of at least v.

Algorithm 1: Match and Merge (MnM)

1 **Input**: A graph $G(V,E)$, and a limit k.
  Result: A k-bounded partition P of V.
2 $G_1(V_1,E_1) \leftarrow G(V,E)$
3 **for** $l \leftarrow 1$ *to* $k-1$ **do**
4   $M_l \leftarrow$ maximum (weight) matching in G_l
5   $G_{l+1} = (V_{l+1}, E_{l+1}) \leftarrow$ an empty graph
6   $V_{l+1} \leftarrow V_l$
7   **for** every $(v_{i_1,\dots,i_l}, v_j) \in M_l$ **do**
8   add vertex $v_{i_1,\dots,i_l,j}$ to V_{l+1}
9   remove $v_{i_1,\dots,i_l}, v_j$ from V_{l+1}
10   **for** every $v_{i_1,\dots,i_{l+1}} \in V_{l+1}$ **do**
11   **for** every $v_q \in V_{l+1}$ **do**
12   **if** $(v_{i_1,\dots,i_l}, v_q) \in E_l$ or $(v_{i_{l+1}}, v_q) \in E_l$ **then**
13   add $(v_{i_1,\dots,i_{l+1}}, v_q)$ to E_{l+1}
14 $P \leftarrow$ an empty partition
15 **for** every $v_{i_1,\dots,i_j} \in G_k$ **do**
16   add the set $\{v_{i_1}, \dots, v_{i_j}\}$ to P
17 **return** P



(a) The graph G_1 **(b) The Matching M_1** **(c) The graph G_2**
(d) The Matching M_2 **(e) The graph G_3** **(f) The graph G_4**

Figure 1: An example for Algorithm 1 where $k = 4$.

4.1 Approximation of the MaxUtil Problem

The MaxUtil problem when $k = 2$ is equivalent to the maximum (weight) matching problem, and thus it can be computed in polynomial time [16]. However, our problem becomes intractable when $k \geq 3$ even in the unweighted setting. Indeed, the decision variant of the MaxUtil problem in the unweighted setting is equivalent to the graph partitioning problem as defined by Hyafil and Rivest [19], which they show to be NP-Complete. Therefore, we provide the Match and Merge (MnM) algorithm (Algorithm 1), which is a polynomial-time approximation algorithm for any $k \geq 3$. The algorithm consists of $k-1$ rounds. Each round is composed of a matching phase followed by a merging phase. Specifically, in round l MnM computes a maximum (weight) matching, $M_l \subseteq E_l$, for G_l (where $G_1 = G$). In the merging phase, MnM creates a graph G_{l+1} that includes a unified node for each pair of matched nodes. The graph G_{l+1} also includes all unmatched nodes, along with their edges to the unified nodes (lines 10-13). Clearly, each node in V_l is composed of up-to l nodes from V_1. Finally, MnM returns the k-bounded partition, P, of all the matched sets. For example, given the graph G_1 in Figure 1a and $k = 4$, the algorithm finds a maximum matching $M_1 = \{(v_1, v_2), (v_3, v_4)\}$ shown in Figure 1b. It then creates the graph G_2, as shown in Figure 1c, and finds a maximum matching for it, $M_2 = \{(v_{3,4}, v_5)\}$ shown in Figure 1d. It then creates the graph G_3, as shown in Figure 1e, and finds a maximum matching for it, $M_3 = \{(v_{3,4,5}, v_6)\}$. Finally, MnM created the graph G_4, as shown in Figure 1f, and returns the 4-bounded partition $P = \{v_1, v_2\}, \{v_3, v_4, v_5, v_6\}$. We note that by the algorithm construction, a unified node $v_{i_1,\dots,i_l}$, is created by merging nodes v_{i_1} and v_{i_2}, and then by merging v_{i_1,i_2} and v_{i_3}, and so on.

4.2 Approximation Ratio for Unweighted Setting

We first show that MnM provides an approximation ratio of $\frac{1}{k-1}$ for the MaxUtil problem in the unweighted setting. For that end, we first prove the following lemma related to the possible edges in every G_l, $l > 1$. Note that the indexes follow the order in which the nodes join the matched node.

Lemma 1. *Given $\hat{v} = v_{i_1,\dots,i_l} \in V_l$, if there exist $v_i, v_j \in V_l$, $v_i \neq v_j$ such that $(v_i, v_{i_n}), (v_j, v_{i_m}) \in E$ for some $1 \leq n \leq m \leq l$, then $n = m$.*

Proof. Observe that for every $v_i, v_j \in V_l$ where $l > 1$, $(v_i, v_j) \notin E$, since M_1 is a maximum matching in G_1. Assume by contradiction and without loss of generality that $n < m$. If $n = 1$ and $m = 2$, then the path $v_i \rightarrow v_{i_n} \rightarrow v_{i_m} \rightarrow v_j$ is an M_1-augmenting path in G_1 ([16]), contrary to the fact that M_1 is a maximum matching in G_1. Therefore, $m \geq 3$.

Now, since v_{i_m} joined the merged node only after the first merge stage, it must be a singleton node in G_2 (as well as v_j). In addition, since $(v_j, v_{i_m}) \in E$, they should have been matched at the very first stage.

□

We now present a hypothetical procedure (Procedure 2) that is provided with a solution to the MaxUtil problem, which is a k-bounded partition (of G) Opt, a graph G_l (as defined in Algorithm 1), and a corresponding round index l. Without loss of generality, we assume that every set $S \in Opt$ is a connected component. Let $O = \{v_o \mid \{v_o\} \in Opt \text{ and } v_o \in V_2\}$. That is, $|O|$ is the number of singletons in the partition Opt that are also not matched in M_1. We show that Procedure 2 finds a matching, and we provide a lower bound on the size of this matching (the number of edges in it). We further show that MnM is guaranteed to perform at least as well as this procedure, which, as we show, results in an approximation ratio of $\frac{1}{k-1}$ for every $k \geq 3$.

Lemma 2. *Procedure 2 finds a matching, R_l, in the graph G_l, such that $|R_l| \geq (|V| - 2|M_1| - \sum_{i=2}^{l-1} |M_i| - |O|)/(k-1)$, where $l > 1$.*

Proof. We first show that Procedure 2 finds a matching, R_l, in the graph G_l. At each iteration of the loop in line 5, we add an edge between a single node, v_q, and a unified node, $v_{i_1,\dots,i_l}$. We consider each single node only once. Therefore, it is not possible to add a single node twice to R_l. Similarly, each time a unified node is added to R_l, every single node $v_n \neq v_q$ such that v_{i_m} and v_n belong to the

Procedure 2: Find matching

1 **Input**: A k-bounded partition (of G) Opt, and a graph $G_l = (V_l, E_l)$.
Result: A matching in G_l.
2 $R_l \leftarrow$ an empty matching
3 **for** each $v_i \in V_l$ such that $\{v_i\} \in Opt$ **do**
4   remove v_i from V_l
5 **for** each $v_q \in V_l$ **do**
6   let $\hat{v}$ be a vertex $v_{i_1,\ldots,i_l}$ such that $(v_q, \hat{v}) \in E_l$ and for a $1 \leq j \leq l$, v_q and v_{i_j} belong to the same set in Opt
7   **for** each $v_n \neq v_q$ **do**
8   **if** $(v_n, \hat{v}) \in E_l$ and exists $1 \leq m \leq l$, s.t. v_{i_m} and v_n belong to the same set in Opt **then**
9   remove v_n from V_l
10   add $(v_q, \hat{v})$ to R_l
11 **return** R_l

same set in Opt, for some $1 \leq m \leq l$, is removed from V_l. Therefore, a unified node is not added more than once. That is, R_l is a matching in G_l.

We now show a lower bound on the size of $|R_l|$. Let $V_l' = \{v_i | v_i \in V_l\}$, i.e., the set of all the single nodes in G_l. In line 9 we remove nodes only when $m = j$ (according to Lemma 1). Given $\hat{v} = v_{i_1,\ldots,i_l}$, there are at most $k-1$ different nodes, $v_{j_1}, \ldots, v_{j_{k-1}}$ that are in the same set with $\hat{v}$ in Opt. Therefore, in each iteration of the loop in line 5, we remove at most $k-2$ single nodes in line 9 while adding one edge to R_l in line 10. Thus, at least $\frac{1}{k-1}$ of the single nodes in V_l (who are not in O) are matched to a unified node. Therefore, $|R_l| \geq \frac{|V_l'|-|O|}{k-1}$. Now, $|V_2'| = |V_1| - 2|M_1|$. In addition, at each iteration $l > i > 1$, $|M_i|$ single nodes are each added to a unified node. Therefore, $|V_l'| = |V_1| - 2|M_1| - \sum_{i=2}^{l-1}|M_i|$. In addition, $V = V_1$. Overall, $|R_l| \geq (|V| - 2|M_1| - \sum_{i=2}^{l-1}|M_i| - |O|)/(k-1)$. □

Theorem 3. *Algorithm 1 provides a solution for the MaxUtil problem with an approximation ratio of $\frac{1}{k-1}$ for every $k \geq 3$, in the unweighted setting.*

Proof. Let P be the k-bounded partition returned by Algorithm 1. Clearly, $u(P) \geq 2 \cdot \sum_{i=1}^{k-1}|M_i|$. For every $l \geq 1$, M_l is a maximum matching and thus $|M_l| \geq |R_l|$. In addition, according to Lemma 2, $|R_l| \geq \frac{|V|-2|M_1|-\sum_{i=2}^{l-1}|M_i|-|O|}{k-1}$. Therefore,

$$u(P) \geq 2 \cdot \sum_{i=1}^{k-1}|M_i| = 2|M_1| + 2 \cdot \sum_{i=2}^{k-1}|M_i|.$$

$$\sum_{i=2}^{k-1}|M_i| = |M_2| + |M_3| + \ldots + |M_{k-1}| \geq$$

$$|M_2| + |M_3| + \ldots + |M_{k-2}| + |R_{k-1}| \geq |M_2| + |M_3| + \ldots + |M_{k-2}| + \frac{|V| - |O| - 2|M_1| - |M_2| - \ldots - |M_{k-2}|}{k-1} =$$

$$\frac{|V|-|O|-2|M_1|}{k-1} + \frac{k-2}{k-1}\sum_{i=2}^{k-2}|M_i| \geq (1+\frac{k-2}{k-1}) \cdot \frac{|V|-|O|-2|M_1|}{k-1} + (\frac{k-2}{k-1})^2\sum_{i=2}^{k-3}|M_i| \geq \ldots \geq$$

$$(1 + \frac{k-2}{k-1} + (\frac{k-2}{k-1})^2 + \ldots + (\frac{k-2}{k-1})^{k-3}) \cdot \frac{|V|-|O|-2|M_1|}{k-1} + (\frac{k-2}{k-1})^{k-2}\sum_{i=2}^{k-1-(k-2)}|M_i| = \sum_{i=0}^{k-3}\Big((\frac{k-2}{k-1})^i \cdot \frac{|V|-|O|-2|M_1|}{k-1}\Big).$$

That is,

$$u(P) \geq 2|M_1| + 2 \cdot \sum_{i=0}^{k-3}\Big((\frac{k-2}{k-1})^i \cdot \frac{|V|-|O|-2|M_1|}{k-1}\Big) =$$

$$2|M_1| + 2 \cdot \frac{|V|-|O|-2|M_1|}{k-1} \cdot \frac{(\frac{k-2}{k-1})^{(k-2)} - 1}{\frac{k-2}{k-1} - 1} =$$

$$2|M_1| + 2(|V|-|O|-2|M_1|) \cdot \frac{(\frac{k-2}{k-1})^{(k-2)} - 1}{(k-1)(\frac{k-2}{k-1} - 1)} =$$

$$2|M_1| - 2(|V|-|O|-2|M_1|)((\frac{k-2}{k-1})^{(k-2)} - 1) =$$

$$2(|V|-|O|)(1 - (\frac{k-2}{k-1})^{(k-2)}) - 2|M_1|(1 - 2 \cdot (\frac{k-2}{k-1})^{(k-2)}).$$

Next, we show that $(1-2\cdot(\frac{k-2}{k-1})^{(k-2)}) \geq 0$. Let $f(k) = (\frac{k-2}{k-1})^{k-2}$, for $k \geq 3$. Thus, $f'(k) = \frac{(k-2)^{k-2}\left(\ln(\frac{k-2}{k-1})(k-1)+1\right)}{(k-1)^{k-1}}$.

Now, $\frac{(k-2)^{k-2}}{(k-1)^{k-1}} > 0$. In addition, it is known that $\ln(x) \leq x - 1$ [22], and thus $\ln(\frac{k-2}{k-1})(k-1)+1 \leq -\frac{1}{k-1}(k-1)+1 = 0$. Therefore, for all $k \geq 3$, $f'(k) \leq 0$ and $f(k) \leq f(3) = \frac{1}{2}$.

Overall, since $|M_1| \leq \frac{|V|-|O|}{2}$,

$$u(P) \geq 2(|V|-|O|)(1-(\frac{k-2}{k-1})^{(k-2)}) - 2\cdot\frac{|V|-|O|}{2}(1-2\cdot(\frac{k-2}{k-1})^{(k-2)}) = |V| - |O|.$$

Now, since in Opt there are at least $|O|$ singletons, then $u(Opt)$ is at most $(|V| - |O|) \cdot (k-1)$, which occurs when all nodes are partitioned into cliques of size k (except those in O). That is,

$$u(P) \geq \frac{u(Opt)}{k-1}.$$

□

Since finding a maximum matching in a graph can be computed in $O(|E|\sqrt{|V|})$, Algorithm 1 runs in $O(kn^{2.5})$ time.

4.3 Approximation Ratio for Weighted Setting

We now show that in the weighted setting MnM provides an approximation ratio of $\frac{1}{k}$ for the MaxUtil problem with an odd k and $\frac{1}{k-1}$ for the problem with an even k. Specifically, we show that the first step of the algorithm, which finds a maximum weight matching, provides such an approximation ratio.

Theorem 4. *Algorithm 1 provides a solution for the MaxUtil problem in the weighted setting with an approximation ratio of $\frac{1}{k}$ for an odd k and an approximation ratio of $\frac{1}{k-1}$ for an even k.*

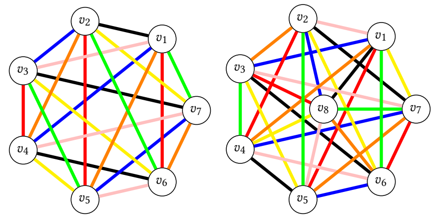


Figure 2: Examples for edge coloring in graphs with an odd and an even number of vertices.

Proof. let $Opt = \{S_1, S_2, ...\}$ be an optimal k-bounded partition, and let M_i be a maximum weight matching for coalition S_i. Due to Behzad et al. [7], for a graph with n vertices, there exists a proper edge coloring using n colors for an odd n and $n-1$ colors for an even n (examples for these coloring for $n = 7$ and for $n = 8$ are shown in Figure 2). Clearly, each color defines a matching in the graph. Thus, each coalition S_i can be partitioned into $|S_i|$ disjoint matchings, if $|S_i|$ is odd, and $|S_i| - 1$ disjoint matchings otherwise. We note that since the matchings are disjoint and cover the entire graph, the sum of the weights of all matchings equals the sum of the weights of all edges in the graph induced by S_i. In addition, for each i, $|S_i| \leq k$. Let M_{S_i} be a maximum weight matching for S_i, imposed by one of the colors. Now, since a maximum is at least as great as the average, and since M_i is a maximum weight matching for coalition S_i, $\sum_{e \in M_i} w(e) \geq \sum_{e \in M_{S_i}} w(e) \geq \frac{\sum_{v \in S_i} W(v, S_i)}{k}$ for an odd k and $\sum_{e \in M_i} w(e) \geq \sum_{e \in M_{S_i}} w(e) \geq \frac{\sum_{v \in S_i} W(v, S_i)}{k-1}$ for an even k. Let M be the maximum weight matching for G found by Algorithm 1 in its first step. Clearly, $\sum_{e \in M} w(e) \geq \sum_i \sum_{e \in M_i} w(e)$. In addition, for the k-bounded partition P that MnM returns, $u(P) \geq \sum_{e \in M} w(e)$. Therefore, in the weighted setting, Algorithm 1 provides a solution for the MaxUtil problem with an approximation ratio of $\frac{1}{k}$ for an odd k and an approximation ratio of $\frac{1}{k-1}$ for an even k. □

We refer the reader to the full version of the paper [20] for results related to the tightness of the approximation ratio of MnM.

5 STABILITY

When considering a stability concept c, we analyze the following two problems: (i) Existence: determine whether for any (G, k) there exists a partition that satisfies c, and (ii) Finding: given (G, k), decide if there exists a partition that satisfies c and if so, find such a partition.

5.1 Core

We begin with the unweighted setting. We show that for $k = 3$ the core is never empty, and we present Algorithm 3, a polynomial time algorithm that finds a 3-bounded partition P in the core. The algorithm begins with all agents in singletons and iteratively considers for each 3-bounded coalition whether it strongly blocks the current partition.

Theorem 5. *In the unweighted setting, there always exists a 3-bounded partition in the core, and it can be found in polynomial time.*

Algorithm 3: Finding a 3-bounded partition in the core

```
1  Input: A graph G(V, E).
   Result: A 3-bounded partition P of V in the core.
2  P ← {{v} for every v ∈ V}
3  V′ ← V
4  outerLoop:
5  for S ⊂ V′, such that |S| = 2 OR |S| = 3 do
6      if ∀v ∈ S, W(v, S) > u(v, P) then
7          P ← P^{−S}
8          if S is clique of size 3 then
9              V′ ← V′ \ S
10         goto outerLoop
11 return P
```

Proof. Consider Algorithm 3. Note that for every 3-bounded partition P, if $S \in P$ is a clique of size 3 then every $v \in S$ cannot belong to any strongly blocking coalition. Therefore, Algorithm 3 removes such vertices from V' (in line 9). Clearly, if Algorithm 3 terminates, the 3-bounded partition P is in the core. We now show that Algorithm 3 must always terminate, and it runs in polynomial time. The algorithm initiates a new iteration (line 4) whenever the if statement in line 6 is true, which can happen when the blocking coalition S, is one of the following:

- Only singletons (i.e., two or three singletons). Then, $u(P)$ increases by at least 2.
- One agent from a coalition in which she has one neighbor, and two singleton agents. Then, $u(P)$ increases by at least 2.
- Two agents, each from a coalition with a single neighbor, and one singleton agent. Then, S must be a clique of size 3, which increases $u(P)$ by 2.
- Three agents, each from a coalition with a single neighbor. Then, S must also be a clique of size 3; however, $u(P)$ remains the same.

Overall, either $u(P)$ has increased by at least 2 or S is a clique of size 3 and thus its vertices are removed from further consideration (in line 9). Since $u(P)$ is bounded by $2|E|$ and the number of vertices is finite, the algorithm must terminate after at most $|E| + |V|/3$ iterations. □

For $k > 3$ it is unclear whether the core can be empty, and how to find a partition in the core. Indeed, we show in simulation that a simple heuristic always finds a partition that is in the core. Our heuristic function works as follows:

(1) Start with a k-bounded partition P, where all the agents are singletons.
(2) Iterate randomly over all the k-bounded coalitions until a coalition S is found, which strongly blocks the partition P.
(3) Update P to be P^{-S}, and return to step (2).

The heuristic terminates when either there is no strongly blocking coalition for the partition P (i.e., P is in the core), or when in 100 consecutive iterations the algorithm only visits partitions that it has already seen before. In the latter case, we restart the search from the very beginning.

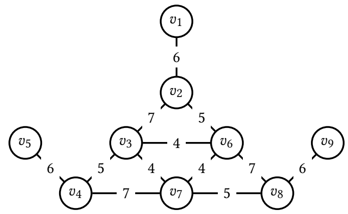


Figure 3: An example of a weighted graph in which the core is empty, for $k = 3$.

We test our heuristic function for $k = 5$ over more than 100 million random graphs of different types: (i) random graphs of size 30 with probability of 0.5 for rewiring each edge, (ii) random trees of size 30, and (iii) random connected Watts–Strogatz small-world graphs of size 30, where each node is joined with its 5 nearest neighbors in a ring topology and with a probability of 0.5 for rewiring each edge.

Our heuristic always found a k-bounded partition that is in the core. Moreover, we had to restart the heuristic in only 33 instances, and then a k-bounded partition in the core was found.

We continue with the analysis of the weighted setting. We first show that the core may be empty. Specifically, Figure 3 provides an example of such a graph for $k = 3$. We use a computer program that iterates over all possible 3-bounded partitions, for verifying that each such 3-bounded partition has a strongly blocking 3-bounded coalition.

Next, we show that the existence problem for the core in the weighted setting is NP-hard, for any $k \geq 3$. Formally,

Definition 5.1 (Core existence problem). *Given a coalition size limit k and a graph G, decide whether a k-bounded partition in the core exists.*

We reduce from the following problem, which was shown to be strongly NP-complete by [13].

Definition 5.2 (3-Dimensional Stable Roommates with Metric Preferences (Metric-3DSR)). *Let A be A set of agents such that $|A| = 3n, n \in \mathbb{N}$, equipped with a metric distance function d. Given an agent $i \in A$ and two triples S_1 and S_2, such that $i \in S_1, S_2$, agent i is said to strictly prefer a triple S_1 to S_2 if $\sum_{j \in S_1 \setminus \{i\}} d(i, j) < \sum_{j \in S_2 \setminus \{i\}} d(i, j)$. A partition of A into triples is said to be core-stable if there is no triple of agents T in which each of the agents strictly prefers T to her triple in the partition. The Metric-3DSR problem asks whether there exists a core-stable partition of A into triples.*

Theorem 6. *In the weighted setting, the Core existence problem is strongly NP-hard, for every $k \geq 3$.*

Proof. Throughout this proof, for a partition P and element i, let $P(i)$ denote the coalition of i in P. Let (A, d) be an instance of Metric-3DSR. We construct an instance $(G = (V, E), k)$ of the Core existence problem. Let $M := \max_{i,j \in A} d(i, j) + 1$. We create a set of agent vertices $U := \{u_i \mid i \in A\}$. For every pair $\{i, j\} \in A$, create an edge $\{u_i, u_j\}$ with weight $2M - d(i, j)$. Intuitively, these weights enforce that every agent prefers being in a triple to being in a pair or alone. Additionally, restricted to triples, these weights give raise to preferences that are identical to those of (A, d).

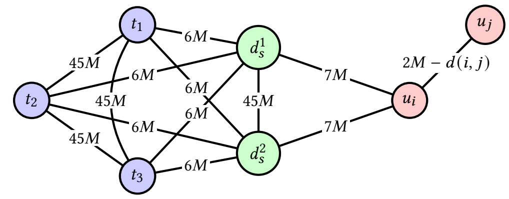


Figure 4: An illustration of the construction in Theorem 6 for $k = 5$, showing the edges between T, D_s for some $s \in \{1, \dots, n+1\}$ and u_i for some $i \in A$. Additionally, we show the edge between u_i and an arbitrary u_j, $j \in A$.

If $k = 3$, then $V := U$. If $k \geq 4$, we additionally create $(n+1)(k-3)$ many dummy vertices $D := \{d_s^t \mid s \in \{1, \dots, n+1\}, t \in \{1, \dots, k-3\}\}$. The dummy vertices enforce that every agent vertex prefers to be in a triple containing two other agent vertices and $k-3$ dummies. For every $s \in \{1, \dots, n+1\}$, the agents $D_s := \{d_s^t \mid t \in \{1, \dots, k-3\}\}$ form a clique with edges of weight $(k-2)15M$. Thus the dummies want to always be with their clique. There are no other edges between dummy agents. For every $i \in A, s \in \{1, \dots, n+1\}, t \in \{1, \dots, k-3\}$, we create an edge $\{u_i, d_s^t\}$ with weight $7M$. We also create additional vertices $T := \{t_1, t_2, t_3\}$. There is an edge of weight $(k-2)15M$ between every pair of vertices in T. For every $d_s^t, s \in \{1, \dots, n+1\}, t \in \{1, \dots, k-3\}, j \in \{1, 2, 3\}$, we add an edge $\{d_s^t, t_j\}$ with weight $6M$. We set $V := U \cup D \cup T$. An illustration of our construction is depicted in Figure 4. We begin by showing the following properties of our construction.

Claim 7. *The following properties hold:*

1. *If $k \geq 4$, then for every $s \in \{1, \dots, n+1\}, t \in \{1, \dots, k-3\}$, the vertex d_s^t strictly prefers a coalition containing D_s to one that does not.*
2. *If $k \geq 4$, then for every $j \in \{1, 2, 3\}$, the vertex t_j strictly prefers a coalition containing T to one that does not. Moreover, for every $s \in \{1, \dots, n+1\}$, there is no coalition that t_j strictly prefers over $D_s \cup T$.*
3. *Let $U^3, U^2 \subseteq U$ be arbitrary subsets such that $|U^3| = 3$ and $|U^2| = 2$. If $k \geq 4$, then for every $s \in \{1, \dots, n+1\}, t \in \{1, \dots, k-3\}$, the vertex d_s^t strictly prefers $D_s \cup U^3$ to $D_s \cup T$, and $D_s \cup T$ to $D_s \cup U^2$.*
4. *If $k \geq 4$, then for every $i \in A$, the vertex u_i strictly prefers a coalition S to S' if $D_s \subseteq S$ for some $s \in \{1, \dots, n+1\}$ and $D \cap S' = \emptyset$.*
5. *If $k \geq 4$, then for every $i \in A, \{j, \ell\}, \{j', \ell'\} \subseteq A \setminus \{i\}, s, s' \in \{1, \dots, n+1\}$ the vertex u_i (strictly) prefers the coalition $D_s \cup \{u_i, u_j, u_\ell\}$ to $D_{s'} \cup \{u_i, u_{j'}, u_{\ell'}\}$ if and only if i (strictly) prefers $\{i, j, \ell\}$ to $\{i, j', \ell'\}$.*
   If $k = 3$, then for every $i \in A, \{j, \ell\}, \{j', \ell'\} \subseteq A \setminus \{i\}$, the vertex u_i (strictly) prefers the coalition $\{u_i, u_j, u_\ell\}$ to $\{u_i, u_{j'}, u_{\ell'}\}$ if and only if i (strictly) prefers $\{i, j, \ell\}$ to $\{i, j', \ell'\}$.
6. *If $k \geq 4$, then for every $i \in A, j, \ell, j' \in A \setminus \{i\}, j \neq \ell, s, s' \in \{1, \dots, n+1\}$, the vertex u_i strictly prefers the $D_s \cup \{u_i, u_j, u_\ell\}$ to $D_{s'} \cup \{u_i, u_{j'}\}$ and to $D_{s'} \cup \{u_i\}$.*
   If $k = 3$, then for every $i \in A, j, \ell, j' \in A \setminus \{i\}, j \neq \ell$, the vertex u_i strictly prefers the coalition $\{u_i, u_j, u_\ell\}$ to $\{u_i, u_{j'}\}$ and to $\{u_i\}$.

(7) *If $k \geq 4$, then for every $\{x_1, x_2, x_3\}, \{x_4, x_5\}, \{x_6\} \subseteq U \cup T, s \in \{1, \ldots, n+1\}, t \in \{1, \ldots, k-3\}$, the vertex d_s^t strictly prefers $D_s \cup \{x_1, x_2, x_3\}$ to $D_s \cup \{x_4, x_5\}$ to $D_s \cup \{x_6\}$ to D_s.*

Due to space constraints, the proof is deferred to the full version of the paper [20].

If (A, d) admits a core-stable partition, then $(G = (V, E), k)$ admits a core-stable partition. Let $P = \{S_1, \ldots, S_n\}$ be a core stable partition of A. We construct a partition P' of V as $P' := \{\{u_i \mid i \in S_j\} \cup D_j \mid j \in \{1, \ldots, n\}\} \cup \{D_{n+1} \cup T\}$ if $k \geq 4$ and $P' := \{\{u_i \mid i \in S_j \mid j \in \{1, \ldots, n\}\}$ if $k = 3$. Assume that P' is not core-stable in (G, k). Then exists $S \subseteq V, |S| \leq k$ that blocks P'.

We start by showing that if $k \geq 4$, any blocking coalition S can have at most 3 elements from U, and the rest of the elements must be from D.

First note that for every $d_s^t, s \in \{1, \ldots, n\}, t \in \{1, \ldots, k-3\}$, we have that $u(d_s^t, P') = (k-4)(k-2)15M + 3 \cdot 7M = (k-4)(k-2)15M + 21M$. For every $d_{n+1}^t, t \in \{1, \ldots, k-3\}$, we have that $u(d_{n+1}^t, P') = (k-4)(k-2)15M + 3 \cdot 6M = (k-4)(k-2)15M + 18M$. By Claim 7(2), no vertex in T wants to deviate.

If $d_s^t \in S$, where $s \in \{1, \ldots, n+1\}, t \in \{1, \ldots, k-3\}$, then $D_s \subseteq S$. Assume, towards a contradiction, that there is some dummy agent $d_s^t \in S, s \in \{1, \ldots, n+1\}, t \in \{1, \ldots, k-3\}$ such that $D_s \not\subseteq S$. By Claim 7(1), the vertex d_s^t prefers $D_s \subseteq P'(d_s^t)$ to S, a contradiction to S blocking.

Moreover, if $D_s \subseteq S$, then $D_{s'} \not\subseteq S$ for any $s' \in \{1, \ldots, n+1\} \setminus \{s\}$. Assume, towards a contradiction, that $D_s \cup D_{s'} \subseteq S$ for some $s, s' \in \{1, \ldots, n+1\}, s \neq s'$. Note that this is only possible if $2(k-3) \leq k \iff k \leq 6$. Then for every $d_s^t, t \in \{1, \ldots, k-3\}$, $u(d_s^t, S) \leq (k-4)(k-2)15M + (6-k)6M$. Since $6 - k \leq 3$, it follows that $u(d_s^t, P'^{-S}) \leq u(d_i^j, P')$, a contradiction. Therefore, if S contains some element in $d_s^t \in D$, then $D \cap S = D_s$.

If $D_s \not\subseteq (D \cap S)$ for any $s \in \{1, \ldots, n+1\}$, then no agent in $U \cap S$ wants to deviate by Claim 7(4), a contradiction. Thus there must be some $s \in \{1, \ldots, n+1\}$ such that $D_s \subseteq S$. If $|S \cap U| > 3$, then $|S \cap D| < k-3$, and thus $D_s \not\subseteq S$ for any $s \in \{1, \ldots, n+1\}$, contradicting the previous statement.

Now we continue in the generic case where $k \geq 3$. If $|S \cap U| < 3$, then by Claim 7(6), every $u_i \in S, i \in A$ prefers $P'(u_i)$ to S, a contradiction to S blocking. Thus it must be that $|S \cap U| = 3$. By Claim 7(5), every $u_i \in S, i \in A$ strictly prefers $P'(u_i)$ to S if and only if i prefers $\hat{S} := \{i \in A \mid u_i \in S \cap U\}$ to $P(i)$. But then $\hat{S}$ blocks P, a contradiction.

If (G, k) admits a core-stable partition, then (A, d) admits a core-stable partition. Let P' be a core stable partition of V.

We first show that if $k \geq 4$, then every coalition in P' contains a clique of the vertices in D. Assume, towards a contradiction, that there is $d_s^t, s \in \{1, \ldots, n+1\}, t \in \{1, \ldots, k-3\}$ such that $D_s' \not\subseteq P'(d_s^t)$. By Claim 7(1), d_s^t prefers D_s to $P'(d_i^j)$. Thus D_s blocks P', a contradiction. By identical reasoning on Claim 7(2), we have that $P'(t_1) = P'(t_2) = P'(t_3)$. For every $s \in \{1, \ldots, n+1\}, t \in \{1, \ldots, k-3\}$, let $P'(D_s) := P'(d_s^t)$. Because all the elements in D_s are in the same coalition in P', this is well-defined. Similarly let $P'(T) := P'(t_1)$.

Next we show that for every $s, s' \in \{1, \ldots, n+1\}, s \neq s'$, we have that $P'(D_s) \neq P'(D_{s'})$. Assume, towards a contradiction, that $P'(D_s) = P'(D_{s'})$ for some $s, s' \in \{1, \ldots, n+1\}, s \neq s'$. Then there are at most n coalitions that contain $D_{s''}$ for some $s'' \in \{1, \ldots, n+1\}$. Since $|U \cup T| = 3n + 3$, there must be at least three vertices in $U \cup T$ that do not have any vertices from D in their coalition. Since $P'(t_1) = P'(t_2) = P'(t_3)$, if one of the vertices in T does not have vertices from D in their coalition, none of them does. In this case $D_s \cup T$ blocks P' by Claim 7(2) and (7). In the case that none of these vertices is in T, let us call an arbitrary size-3 subset of them U^3. The coalition $U^3 \cup D_s$ blocks P' by Claim 7(4) and (7).

Next we show that if $k \geq 4$, then every coalition containing some triple of vertices in U must also contain a clique of vertices from D. Assume, towards a contradiction, that there is some $i \in A$ such that $D_s \not\subseteq P'(u_i)$ for any $s \in \{1, \ldots, n+1\}$ and $|P'(u_i) \cap U| \geq 3$. Let U^3 be an arbitrary size 3 subset of $P'(u_i) \cap U$. Since there are $3n$ agents in U and $n+1$ cliques in D, there must be a clique $D_s, s \in \{1, \ldots, n+1\}$ such that $|P'(D_s) \cap U| \leq 3$. By Claim 7(4), every agent $u_j \in U^3$ prefers $U^3 \cup D_s$ to $P'(u_j)$. By Claim 7(3) and (7), every agent in D_s prefers $U^3 \cup D_s$ to $P'(D_s)$. Thus P' is not stable.

We proceed to show that for every $u_i, i \in A$, $|P'(u_i) \cap U| = 3$. Assume, towards a contradiction, that for some $u_i, i \in A$, $|P'(u_i) \cap U| > 3$. If $k = 3$, this trivially leads to a contradiction. Thus assume $k \geq 4$. Then $|P'(u_i) \cap D| < k-3$. Since every vertex in D must have its whole clique in the coalition, $P'(u_i) \cap D = \emptyset$. Since there are $\{1, \ldots, n+1\}$ cliques in D and $3n$ agents in U, there must be some $D_{s'}, s' \in \{1, \ldots, n+1\}$ such that $|P'(D_{s'}) \cap U| < 3$. Let U^3 be an arbitrary subset of $P'(u_i)$ such that $|U^3| = 3$. By Claim 7(5), every vertex in U^3 prefers $U^3 \cup D_{s'}$ to its current coalition. By Claim 7(3) and (5), the vertices in $D_{s'}$ also prefer $U^3 \cup D_{s'}$ to $P'(D_{s'})$, meaning that $U^3 \cup D_{s'}$ blocks P', a contradiction.

If there is some $u_i, i \in A$, such that $|P'(u_i) \cap U| < 3$, then by previous paragraph there must be at least two other agents in $j, \ell \in A$ such that $|P'(u_j) \cap U| < 3$ and $|P'(u_\ell) \cap U| < 3$. If $k \geq 4$, since there are $3n$ agents in U and $n+1$ cliques in D, there must be some clique $D_s, s \in \{1, \ldots, n+1\}$ such that $|P'(D_s) \cap U| \leq 3$. By Claim 7(1) and (6), every agent $x \in \{u_i, u_j, u_\ell\}$ prefers $\{u_i, u_j, u_\ell\} \cup D_s$ to $P'(x)$. By Claim 7(3) and (7), every agent in D_s prefers $\{u_i, u_j, u_\ell\} \cup D_s$ to $P'(D_s)$. Thus P' is not stable. If $k = 3$, then by Claim 7(6), every agent $x \in \{u_i, u_j, u_\ell\}$ prefers $\{u_i, u_j, u_\ell\}$ to $P'(x)$, a contradiction.

Therefore, if $k \geq 4$, then every coalition in P' containing vertices in U must be of the form $U^3 \cup D_s$, where $s \in \{1, \ldots, n+1\}$ and $U^3 \subseteq U, |U^3| = 3$. If $k = 3$, every coalition $S \in P'$ must satisfy $|S| = 3$. We construct a partition P of A as follows: For every $i \in A$, we set $P(i) = \{j \mid u_j \in U \cap P'(u_i)\}$. By previous reasoning, this must partition A into triples.

Assume, towards a contradiction, that P is not stable. Then there must be some triple $S = \{i, j, \ell\}$ such that S blocks P. Let $S' := \{u_i, u_j, u_\ell\}$. If $k \geq 4$, we observe that since there are $3n$ agents in U and $n+1$ cliques in B, there must be some clique $D_s, s \in \{1, \ldots, n+1\}$ such that $P(D_s) \cap U = \emptyset$. By Claim 7(7), every agent in D_s prefers $S' \cup D_s$ to $P'(D_s)$. By Claim 7(5), every agent $u_x \in S'$ strictly prefers $S' \cup D_s$ to $P'(u_x)$. Thus P' is not stable, a contradiction. If $k = 3$, by Claim 7(5), every agent $u_x \in S'$ strictly prefers S' to $P'(u_x)$. Thus P' is not stable, a contradiction. □

5.2 Strict Core (SC)

We first show that for every size limit, k, there is at least one graph where there is no k-bounded partition in the strict core. Indeed, given a size limit k, we build the graph $G(V, E)$, which is a clique of size $k + 1$. For every partition P of V, let S be a coalition in P such that $|S| < k$. Now, any set of agents of size k that also contains some $v \in S$ is a weakly blocking k-bounded coalition for P. Furthermore, even verifying the existence of the strict core is a hard problem.

Definition 5.3 (*SC* existence problem). *Given a coalition size limit k and a graph G, decide whether a k-bounded partition in the strict core exists.*

For the hardness proof, we define for each $k \in \mathbb{N}$ the $Cliques_k$ problem, which is as follows.

Definition 5.4 ($Cliques_k$). *Given an undirected and unweighted graph $G(V, E)$, decide whether V can be partitioned into disjoint cliques, such that each clique is composed of exactly k vertices.*

Clearly, $Cliques_2$ can be decided in polynomial time by computing a maximum matching of the graph G, M, and testing whether $|M| = \frac{|V|}{2}$. However, $Cliques_k$ becomes hard when $k \geq 3$.

Lemma 8. *$Cliques_k$ is NP-Complete for every $k \geq 3$.*

Proof. Clearly, $Cliques_k$ is NP for every k. We use induction to show that any $Cliques_k$ is NP-Hard for every $k \geq 3$. $Cliques_3$ is known as the 'partition into triangles' problem, which was shown to be NP-Complete [18]. Given that $Cliques_k$ is NP-Hard we show that $Cliques_{k+1}$ is also NP-Hard. Given an instance of the $Cliques_k$ on a graph $G(V, E)$, we construct the following instance. We build a graph $G'(V', E')$, in which we add a set of nodes $\hat{V} = \hat{v}_1, ..., \hat{v}_{\frac{|V|}{k}}$, i.e., $V' = V \cup \hat{V}$. If $e \in E$ then $e \in E'$, and for every $v \in V, \hat{v} \in \hat{V}$ we add $(v, \hat{v})$ to E'. Clearly, V can be partitioned into disjoint cliques with exactly k vertices if and only if V' can be partitioned into disjoint cliques with exactly $k + 1$ vertices. □

Theorem 9. *The SC existence problem is NP-hard.*

Proof. Given an instance of the $Cliques_k$ on a graph $G(V, E)$, we construct the following instance. We build a graph $G'(V', E')$ such that V' contains all the nodes from V. In addition, for every $v_x \in V$ we add the nodes $\hat{v}_x$ and $v_x^1, \ldots, v_x^{k-1}$ to V'. Now, E' contains all the edges of E, and for every $v_x \in V$ and $1 \leq i \leq k - 1$ we add (v_x, v_x^i), $(v_x^i, \hat{v}_x)$ to E'. Finally, for every $v_x \in V$ and $1 \leq i, j \leq k-1$, $i \neq j$ we add (v_x^i, v_x^j) to E'. We first show that if G cannot be partitioned into disjoint cliques of size k, then the strict core is empty. Indeed, assume that G cannot be partitioned into disjoint cliques of size k, and let P be a k-bounded partition of V'. Then, there is at least one vertex $v_x \in V$ that belongs to a coalition $S \in P$, such that either: (1) $W(v_x, S) < k - 1$, or (2) $v_x^i \in S$ for some i between 1 and $k-1$. In case 1, the coalition $\{v_x, v_x^1, \ldots, v_x^{k-1}\}$ is a weakly blocking k-bounded coalition. In case 2, the coalition $\{\hat{v}_x, v_x^1, \ldots, v_x^{k-1}\}$ is a weakly blocking k-bounded coalition. Therefore, if the strict core is not empty, then G can be partitioned into disjoint cliques of size k.

We now show that if G can be partitioned into disjoint cliques of size k, then the strict core is not empty. Clearly, in this case G' can be partitioned into disjoint cliques of size k, and this partition is in the strict core. Therefore, if the strict core is empty, then G cannot be partitioned into disjoint cliques of size k. □

5.3 Contractual Strict Core (CSC)

We show that the CSC is never empty. Indeed, given any (G, k), the following algorithm finds a k-bounded partition in the CSC:

(1) Start with a k-bounded partition P, where all the agents are singletons.
(2) Iterate over all the coalitions in P until two coalitions, S_1, S_2, are found, such that $|S_1| + |S_2| \leq k$ and $u(P) < u(P^{-S_1 \cup S_2})$.
(3) Update P to be $P^{-S_1 \cup S_2}$, and return to step (2).

The algorithm terminates when step 2 does not find two coalitions that meet the required conditions.

Theorem 10. *There always exists a k-bounded partition in the CSC, and it can be found in polynomial time.*

Proof. At each iteration, the number of the coalitions in P decreases and thus the algorithm must terminate after at most $k - 1$ iterations. Consider the k-bounded partition P when the algorithm terminates. Clearly, there are no two coalitions in P that can benefit from breaking off and joining together. In addition, observe that every coalition $S \in P$ is a connected component. Thus, no coalition $S' \subsetneq S$ can break off without decreasing the utility of at least one agent from $S \setminus S'$. Therefore, P is in the CSC. □

6 CONCLUSIONS AND FUTURE WORK

In this paper, we study ASHGs with a bounded coalition size. We provide MnM, an approximation algorithm for maximizing the utilitarian social welfare and study the computational aspects of the core, the SC, and the CSC. We note that MnM can be improved by running the algorithm and iteratively joining together any two coalitions that improve the social welfare (without violating the size constraint). This improved version is guaranteed to find a partition that is in the CSC while maintaining the approximation ratio for the MaxUtil problem. Unfortunately, this improvement does not result in an improved approximation ratio when $k = 3$, and whether it improves the approximation ratio when $k > 3$ remains an open problem. Generally, providing an inapproximability result or a better approximation algorithm for the MaxUtil problem is an important open problem. Furthermore, the existence of the core in the unweighted setting when $k > 3$ is an essential open problem.

In addition, there are several interesting directions for extending our work. Since the MaxUtil problem is computationally hard, it will be interesting to investigate some variants. For example, the problem of finding a k-bounded partition, such that each agent will be matched with at least one friend in its coalition. Another interesting research direction is to incorporate skills in our model, motivated by coalitional skill games [4]. That is, each agent has a set of skills, and each coalition is required to have at least one agent that acquires each of the skills.

ACKNOWLEDGMENTS

This research has been partly supported by the Ministry of Science, Technology & Space (MOST), Israel.

REFERENCES

[1] Haris Aziz, Felix Brandt, and Hans Georg Seedig. 2013. Computing Desirable Partitions in Additively Separable Hedonic Games. *Artificial Intelligence* 195 (2013), 316–334.

[2] Haris Aziz and Rahul Savani. 2016. Hedonic Games. In *Handbook of Computational Social Choice*, Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D. Procaccia (Eds.). Cambridge University Press, Chapter 15, 356–376.

[3] Yoram Bachrach, Pushmeet Kohli, Vladimir Kolmogorov, and Morteza Zadimoghaddam. 2013. Optimal Coalition Structure Generation in Cooperative Graph Games. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence*. 81–87.

[4] Yoram Bachrach and Jeffrey S Rosenschein. 2008. Coalitional Skill Games. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems*. 1023–1030.

[5] Coralio Ballester. 2004. NP-completeness in Hedonic Games. *Games and Economic Behavior* 49, 1 (2004), 1–30.

[6] Suryapratim Banerjee, Hideo Konishi, and Tayfun Sönmez. 2001. Core in a Simple Coalition Formation Game. *Social Choice and Welfare* 18, 1 (2001), 135–153.

[7] Mehdi Behzad, Gary Chartrand, and John K Cooper JR. 1967. The Colour Numbers of Complete Graphs. *Journal of the London Mathematical Society* 1, 1 (1967), 226–228.

[8] Vittorio Bilò, Angelo Fanelli, Michele Flammini, Gianpiero Monaco, and Luca Moscardelli. 2019. Optimality and Nash Stability in Additively Separable Generalized Group Activity Selection Problems. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. 102–108.

[9] Vittorio Bilò, Gianpiero Monaco, and Luca Moscardelli. 2022. Hedonic Games with Fixed-Size Coalitions. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence*. 9287–9295.

[10] Anna Bogomolnaia and Matthew O Jackson. 2002. The Stability of Hedonic Coalition Structures. *Games and Economic Behavior* 38, 2 (2002), 201–230.

[11] Katarı Cechlárová, Antonio Romero-Medina, et al. 2001. Stability in Coalition Formation Games. *International Journal of Game Theory* 29, 4 (2001), 487–494.

[12] Ágnes Cseh, Tamás Fleiner, and Petra Harján. 2019. Pareto Optimal Coalitions of Fixed Size. *arXiv preprint arXiv:1901.06737* (2019).

[13] Vladimir G Deineko and Gerhard J Woeginger. 2013. Two Hardness Results for Core Stability in Hedonic Coalition Formation Games. *Discrete Applied Mathematics* 161, 13-14 (2013), 1837–1842.

[14] Xiaotie Deng and Christos H Papadimitriou. 1994. On the Complexity of Cooperative Solution Concepts. *Mathematics of operations research* 19, 2 (1994), 257–266.

[15] Jacques H Dreze and Joseph Greenberg. 1980. Hedonic Coalitions: Optimality and Stability. *Econometrica: Journal of the Econometric Society* (1980), 987–1003.

[16] Jack Edmonds. 1965. Paths, Trees, and Flowers. *Canadian Journal of Mathematics* 17 (1965), 449––467.

[17] Michele Flammini, Gianpiero Monaco, Luca Moscardelli, Mordechai Shalom, and Shmuel Zaks. 2021. On the Online Coalition Structure Generation Problem. *Journal of Artificial Intelligence Research* 72 (2021), 1215–1250.

[18] Michael R Garey and David S Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-completeness*. W. H. Freeman.

[19] Laurent Hyafil and Ronald L Rivest. 1973. *Graph Partitioning and Constructing Optimal Decision Trees are Polynomial Complete Problems*. Technical Report Rapport de Recherche no. 33. IRIA – Laboratoire de Recherche en Informatique et Automatique.

[20] Chaya Levinger, Noam Hazon, Sofia Simola, and Azaria Amos. 2023. Coalition Formation with Bounded Coalition Size. *arXiv preprint arXiv:2306.01378* (2023).

[21] Lily Li, Evi Micha, Aleksandar Nikolov, and Nisarg Shah. 2023. Partitioning Friends Fairly. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence*. 5747–5754.

[22] Eric R. Love. 1980. 64.4 Some Logarithm Inequalities. *The Mathematical Gazette* 64, 427 (1980), 55–57.

[23] Martin Olsen. 2009. Nash Stability in Additively Separable Hedonic Games and Community Structures. *Theory of Computing Systems* 45, 4 (2009), 917–925.

[24] Liat Sless, Noam Hazon, Sarit Kraus, and Michael Wooldridge. 2018. Forming k coalitions and facilitating relationships in social networks. *Artificial Intelligence* 259 (2018), 217–245.

[25] Shao-Chin Sung and Dinko Dimitrov. 2010. Computational Complexity in Additive Hedonic Games. *European Journal of Operational Research* 203, 3 (2010), 635–639.

[26] Mason Wright and Yevgeniy Vorobeychik. 2015. Mechanism Design for Team Formation. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*. 1050–1056.