Algorithmic Solutions
for Maximizing Shareable Costs

Rong Zou
University of Twente
Enschede, The Netherlands
r.zou@utwente.nl

Boyue Lin
Northwestern Polytechnical University
Xi’an, Shaanxi, China
linboyuexigongda@163.com

Marc Uetz
University of Twente
Enschede, The Netherlands
m.uetz@utwente.nl

Matthias Walter
University of Twente
Enschede, The Netherlands
m.walter@utwente.nl

March 2, 2023

Abstract

This paper addresses the optimization problem to maximize the total costs that can be shared among a group of agents, while maintaining stability in the sense of the core constraints of a cooperative transferable utility game, or TU game. This means that all subsets of agents have an outside option at a certain cost, and stability requires that the cost shares are defined so that none of the outside options is preferable. When maximizing total shareable costs, the cost shares must satisfy all constraints that define the core of a TU game, except for being budget balanced. The paper gives a fairly complete picture of the computational complexity of this optimization problem, in relation to classical computational problems on the core. We also show that, for games with an empty core, the problem is equivalent to computing minimal core relaxations for several relaxations that have been proposed earlier. As an example for a class of cost sharing games with non-empty core, we address minimum cost spanning tree games. While it is known that cost shares in the core can be found efficiently, we show that the computation of maximal cost shares is NP-hard for minimum cost spanning tree games. We also derive a 2-approximation algorithm. Our work opens several directions for future work.

Keywords Cost Sharing, Core, Minimum Spanning Tree Game, Approximation

1 Introduction

The fundamental algorithmic question that is addressed in this paper is: Can we maximize the total costs that can be shared among a set of agents, while maintaining coalitional stability? Here, coalitional stability refers to the core constraints of an underlying cooperative transferable utility game, or TU game: Any proper subset of the set of all agents has an outside option at a certain cost, and coalitional stability of cost shares means that all subsets of agents are willing to accept the cost shares, because their outside option is less attractive, meaning that it is at least as costly as the sum of their cost shares.

This question is arguably a fundamental question for the design of cost sharing mechanisms, and as it turns out, the algorithmic question to maximize the total shareable costs has not been addressed systematically.
Several closely related results exist and will be discussed later, but these are scattered and partially ignorant of each other, and moreover they mostly do not address the underlying algorithmic question.

That said, the main contributions of this paper are as follows. We introduce a basic polyhedral object that we refer to as the “almost core” of a cooperative game. It is obtained from the core by relaxing the requirement that the cost shares must be budget balanced. By definition, this polyhedron is non-empty. The algorithmic problem that we address is to maximize cost shares which lie in the almost core, which is a linear optimization problem over that polyhedron. For the case that the underlying core of the cooperative game is empty, we show that the computational problem to maximize shareable costs is equivalent to finding a minimal non-empty core relaxation for several of the core relaxations that have been proposed earlier in the literature. While this is maybe not surprising in hindsight, it does not seem to appear anywhere in the literature. The paper further establishes complexity theoretic results that relate computational problems for the almost core with corresponding problems for the classical core. While it turns out that general linear optimization over almost core and core share the same algorithmic complexity, it turns out that there are classes of games where core elements can be efficiently computed, while the computation of maximal shareable costs is NP-hard. That NP-hardness result is obtained for a well studied class of games with non-empty core, namely minimum cost spanning tree games. This class of games is interesting also because the resulting cost function is subadditive but generally not submodular. And while submodularity yields polynomial-time algorithms, our hardness result shows that subadditivity does not suffice. For minimum cost spanning tree games, we further show how to obtain a 2-approximation algorithm for maximizing shareable costs.

The structure of this paper is as follows. The basic notions and definitions are given in Section 2. As previous papers have mostly focused on core relaxations for unbalanced games (that is, with an empty core), we briefly review these in Section 3 and discuss how they relate to the problem to compute maximal “almost core” cost shares. A novel aspect of our approach is to also address games that have a non-empty core. Section 4 therefore relates linear optimization over the “almost core” to the core, and we derive some algorithmic consequences. Section 5 then addresses the problem to compute maximal cost shares for minimum cost spanning tree (MST) games, showing NP-hardness, as well as giving a 2-approximation algorithm. We conclude with some open problems in Section 6.

## 2 Core and Almost Core for TU Games

A cooperative game with transferable utility (henceforth TU game) is described by a pair \((N,c)\) where \(N = \{1,\ldots,n\}\) denotes the set of agents, and \(c : 2^N \rightarrow \mathbb{R}_{\geq 0}\) is the characteristic function that assigns to every coalition \(S\) a value \(c(S)\) representing the cost of an “outside option”, which is the minimum total cost that the agents in \(S\) can achieve if they cooperate amongst themselves. Denote by \(G^N\) the set of all TU games over \(N\), and with a slight overload of notation write \(n = |N|\) for the number of agents. An allocation for \((N,c)\) is a vector \(x \in \mathbb{R}^n\) with \(x_i\) being the cost share allocated to agent \(i \in N\). For convenience, we write \(x(S) = \sum_{i \in S} x_i\). An allocation \(x\) is said to be budget balanced if \(x(N) = c(N)\). That means that the total cost of the so-called grand coalition \(N\) is being distributed over the individual agents. It is called stable if it satisfies coalitional stability, i.e., \(x(S) \leq c(S)\) for all \(S \not\subseteq N\). The core \(\text{core}[20]\) of game \((N,c)\), arguably one of the most important concepts in cooperative game theory, consists of all budget balanced allocations satisfying coalitional stability. The core of a TU game is given by

\[
C_{(N,c)} := \{ x \in \mathbb{R}^n : x(S) \leq c(S) \ \forall S \not\subseteq N, \ x(N) = c(N) \}.
\]

The core of a TU game is non-empty iff the game is balanced \([7,38]\). In fact, being balanced is just a dual characterization of the non-emptiness of the polyhedron \(C_{(N,c)}\).

When we drop the equality constraint that a core allocation is budget balanced, so do not require that \(x(N) = c(N)\), it allows to vary the total cost that is distributed over the set of agents, resulting in a problem that always has a feasible solution. This captures the idea that, depending on the underlying game, one may have to, or want to, distribute either less or more than \(c(N)\).
For convenience, we refer to the set of all such allocations as the *almost core*, $AC$. Formally, given a TU game $(N,c) \in \mathcal{G}^N$, define the almost core for $(N,c)$ by

$$AC_{(N,c)} := \{ x \in \mathbb{R}^n : x(S) \leq c(S) \ \forall S \subseteq N \}.$$ 

Obviously, $C_{(N,c)} \subseteq AC_{(N,c)}$. The major motivation for this definition is to systematically study the algorithmic complexity of cooperative games without having to obey to budget balance, so optimization over the polyhedron $AC_{(N,c)}$. Let us motivate the relevance of this problem.

On the one hand, if the total costs $c(N)$ of the grand coalition cannot be distributed over the set of agents while maintaining coalitional stability, i.e., the game is unbalanced, it is a natural question to ask what fraction of the total cost $c(N)$ can be maximally distributed while maintaining coalitional stability. This problem has been addressed under different names, among them the price of stability of a cooperative game [3]. It has received quite some attention in the literature, e.g. [1, 2, 3, 4, 8, 21, 24, 30, 31, 32]. Indeed, for games with empty core, maximizing $x(N)$ over the almost core is equivalent to computing the price of stability, and also to computing some other minimal core relaxations proposed earlier in the literature; see Section 3 for details.

On the other hand, also if the core is non-empty one may be interested in maximizing the total cost that can be distributed over the set of agents. It reveals the maximal value for $c(N)$ that would still yield a non-empty core. One motivation for this maximization problem is to determine the maximal tax rate that could be levied on a given $c(N)$, without any subset of agents $S \subseteq N$ wanting to deviate.

That said, the object of interest of this paper is the following linear program.

$$\max \{ x(N) : x \in AC_{(N,c)} \}. \tag{1}$$

The objective value of this linear program indicates the largest total cost that can be shared among the agents while retaining stability in the sense that no subset of agents $S \subseteq N$ would prefer to deviate to the outside option. We call an optimal solution value for this linear program the *almost core optimum*, and any maximizer is called an *optimal almost core allocation*. Sometimes we also consider the restricted problem where we also require that $x \geq 0$, which means that agents must not receive subsidies.

Clearly, the core of a game is non-empty if and only if the almost core optimum is larger than or equal to $c(N)$. We study problem (1) mainly for games with non-empty core, while for games with empty core we next give a fairly complete overview of its relation to earlier proposed core relaxations.

### 3 Equivalent and Related Relaxations of the Core

In this section we review several well-known and related concepts that were introduced in order to deal with games having an empty core and discuss their relationship to the almost core (optimum).

The first relaxation of the core, introduced by Shapley and Shubik [37], is the *strong ε-core*, defined as

$$C^*_\varepsilon(N,c) := \{ x \in \mathbb{R}^n : x(S) \leq c(S) + \varepsilon \ \forall S \subseteq N, \ x(N) = c(N) \}.$$ 

We denote the smallest $\varepsilon \geq 0$ for which this set is non-empty by $\varepsilon^*_\varepsilon$. The corresponding set $C^*_\varepsilon(N,c)$ is called the *least core* [29].

Shapley and Shubik [37] also introduced the *weak ε-core* as

$$C^w_\varepsilon(N,c) := \{ x \in \mathbb{R}^n : x(S) \leq c(S) + \varepsilon \cdot |S| \ \forall S \subseteq N, \ x(N) = c(N) \}.$$ 

We denote the smallest $\varepsilon \geq 0$ for which this set is non-empty by $\varepsilon^*_w$. Note that by definition, for any $\varepsilon \geq 0$, $C^*_\varepsilon(N,c) \subseteq C^w_\varepsilon(N,c)$, and hence $\varepsilon^*_w \leq \varepsilon^*_\varepsilon$.

Instead of using an additive relaxation of the constraints, Faigle and Kern [17] defined the multiplicative ε-core as

$$C^*_m(N,c) := \{ x \in \mathbb{R}^n : x(S) \leq (1 + \varepsilon) \cdot c(S) \ \forall S \subseteq N, \ x(N) = c(N) \}.$$ 

Denote the smallest $\varepsilon \geq 0$ for which this set is non-empty by $\varepsilon^*_m$. 


A different viewpoint is called approximate core or $\gamma$-core \cite{24} for some $\gamma \in [0,1]$, it is defined as
\[ C^\gamma(N,c) := \{ x \in \mathbb{R}^n : x(S) \leq c(S) \forall S \subseteq N, \gamma \cdot c(N) \leq x(N) \}. \]

Denote the largest $\gamma \leq 1$ for which this set is non-empty by $\gamma^*_\ast$.

The gap between the almost core optimum and the total cost of the grand coalition $c(N)$ was called the cost of stability for an unbalanced cooperative game by Bachrach et al. \cite{3}. For (unbalanced) cost sharing games it is defined by Meir et al. \cite{30} as
\[ \delta_{\text{CoS}} := c(N) - \max \{ x(N) : x(S) \leq c(S) \forall S \subseteq N \}. \]

An alternative viewpoint was independently introduced in a paper by Bejan and Gómez \cite{4} who considered, for profit sharing games, the so-called extended core. In order to define it for cost sharing games, let
\[ \delta_{\text{ec}} := \min \{ t(N) : \exists (x,t) \in \mathbb{R}^n \times \mathbb{R}_{\geq 0}^n, x(N) = c(N), (x-t)(S) \leq c(S) \forall S \not\subseteq N \}. \]

The extended core is now the set of all budget balanced $x \in \mathbb{R}^n$, so all $x$ with $x(N) = c(N)$ for which the minimum above is attained (for suitable $t \in \mathbb{R}_{\geq 0}^n$).

Yet another concept to stabilize an unbalanced game was considered by Zick, Polukarov, and Jennings \cite{39}. Interpreting $t_i$ in the definition of the extended core of Bejan and Gómez \cite{4} as a discount offered to agent $i$, in \cite{39} a coalitional discount $t_S$ is offered to each agent set $S$. This is an exponential blowup of the solution space, which however gives more flexibility.

For unbalanced games, computing the almost core optimum is clearly equivalent to computing the cost of stability $\delta^*_{\text{CoS}}$. The following theorem further shows how the different core relaxations are related with respect to optimization. We do believe some of these equivalences were known, yet we could not find them anywhere in the literature. Hence we summarize them here, and also give the short proof.

**Theorem 1.** For any TU game $(N,c)$ with empty core, the optimization problems for the weak $\varepsilon$-core, the multiplicative $\varepsilon$-core, the cost of stability and the extended core are equivalent. In particular, the values satisfy
\[ \delta_{\text{ec}} = (1 - \gamma^*_\ast) \cdot c(N) = \frac{\varepsilon^*_\ast}{1 + \varepsilon^*_m} \cdot c(N) = \delta_{\text{CoS}} = \varepsilon^*_w \cdot n. \]

**Proof.** First, we establish $\delta^*_{\text{CoS}} = \delta^*_{\text{ec}}$. We substitute $x - t$ by $x'$ in \cite{2} and obtain
\[ \delta^*_{\text{ec}} = \min \{ t(N) : \exists (x',t) \in \mathbb{R}^n \times \mathbb{R}_{\geq 0}^n, x'(N) + t(N) = c(N), x'(S) \leq c(S) \forall S \not\subseteq N \}. \]

Now it is easy to see that the actual entries of $t$ do not matter (except for nonnegativity), but only the value $t(N)$ is important. This yields $\delta^*_{\text{CoS}} = \delta^*_{\text{ec}}$.

Second, we show $\delta^*_{\text{CoS}} = (1 - \gamma^*_\ast) \cdot c(N)$. To this end, observe
\[ \gamma^*_\ast = \max \{ \gamma \in \mathbb{R} : \exists x \in \mathbb{R}^n, x(S) \leq c(S) \forall S \subseteq N, x(N) = \gamma c(N) \}. \]

Clearly, the maximum is attained by $x^* \in \mathbb{R}^n$ with $x^*(N)$ maximum. Moreover, the value of $\gamma^*_\ast$ is then equal to $x^*(N)/c(N)$. This shows $\delta^*_{\text{CoS}}/c(N) = 1 - \gamma^*_\ast$.

Third, we show $1 - \gamma^*_\ast = \varepsilon^*_m/(1 + \varepsilon^*_m)$. Observe that the map $\pi : \mathbb{R}^n \to \mathbb{R}^n$ defined by $\pi(x) = (1 + \varepsilon) x$ induces a bijection between allocations $x \in \mathbb{R}^n$ with $x(S) \leq c(S)$ for all $S \subseteq N$ and allocations $\pi(x)$ with $x(S) \leq (1 + \varepsilon)c(S)$ for all $S \subseteq N$. Moreover, $x(\pi(N)) = (1 + \varepsilon) x(N)$. Hence, $C^\gamma_m(N,c)$ is (non-)empty if and only if $C^\gamma_m(N,c)$ is (non-)empty, where $\gamma = 1/(1 + \varepsilon)$ holds. This implies $\gamma^*_\ast = 1/(1 + \varepsilon^*_m)$.

We finally show $\delta^*_{\text{CoS}} = \varepsilon^*_w \cdot n$. To this end, in
\[ \varepsilon^*_w = \min \{ \varepsilon \geq 0 : \exists x, x(S) \leq c(S) + \varepsilon \cdot |S| \forall S \not\subseteq N, x(N) = c(N) \} \]
we substitute $x$ by $x' + (\varepsilon, \varepsilon, \ldots, \varepsilon)$ which yields
\[ \varepsilon^*_w = \min \{ \varepsilon \geq 0 : \exists x', x'(S) \leq c(S) \forall S \not\subseteq N, x'(N) + \varepsilon \cdot n = c(N) \}. \]

Clearly, the minimum $\varepsilon^*_w$ is attained if and only if $\varepsilon \cdot n = \delta^*_{\text{CoS}}$ holds. \hfill $\square$
Moreover, it was shown in [31] Section 4 that \( \varepsilon_N^* \geq \frac{1}{\sqrt{n}} \varepsilon_\star \). Further relations between the cost of stability \( \delta_{\text{Cost}}^* \) and other core relaxations for specific classes of games appear in [31, 2], e.g., it is true that for superraditive (profit sharing) games, \( \delta_{\text{Cost}}^* \leq \sqrt{n} \varepsilon_\star \) and \( \sqrt{n} \varepsilon_\star w \leq \varepsilon_\star \).

To the best of our knowledge, most of the previous work in this direction was about determining bounds on the cost of stability (see [3, 30, 31, 32, 8]) or other structural insights [4]. Algorithmic considerations were made for specific (unbalanced) games such as weighted voting games [3] and threshold network flow games [35]. Aziz, Brandt and Harrenstein [1] settle the computational complexity of computing the cost of stability (and other measures) for many other specific games, and Chalkiadakis, Greco and Markakis extend this work under additional assumptions on the so-called interaction graph [10]. Approximations of \( \varepsilon_w^* \) for the multiplicative \((1+\varepsilon)\)-core and corresponding allocations have also been obtained for the symmetric traveling salesman game by Faigle et al. [16], and for the asymmetric case also by Bläser et al. [6].

There are also papers that attack the problem from a computational point of view, however without having immediate implications for the results in this paper. We mention them briefly. Under the name “optimal cost salesman game by Faigle et al. [16], and for the asymmetric case also by Bläser et al. [6].

In this section we investigate the computational complexity of optimization problems related to the (non-negative functions \( P_{x} \) and \( AC_{(N,c)} \) as well as optimization over \( P_{(N,c)} \cap \mathbb{R}^n_\geq 0 \) and \( AC_{(N,c)} \cap \mathbb{R}^n_\geq 0 \) for families of games \((N,c)\). Note that if the core is non-empty then it is the set of optimal solutions when maximizing \( 1 \cdot x \) over \( P_{(N,c)} \). Also note that whenever the core of a game \((N,c)\) is empty, this means that the constraint \( x(N) \leq c(N) \) is implied by the set of constraints \( x(S) \leq c(S) \), \( S \not\subseteq N \), which in turn implies \( P_{(N,c)} = AC_{(N,c)} \). For games with non-empty core, we get the following correspondence between the optimization problems for the two polyhedra.

**Theorem 2.** For a family of games \((N,c)\), linear optimization problems over \( AC_{(N,c)} \) can be solved in polynomial time if and only if linear optimization problems over \( P_{(N,c)} \) can be solved in polynomial time.

**Proof.** In order to prove the result we make use of the equivalence of optimization and separation [28, 25, 33]. This means, we only need to show that we can solve the separation problem for \( P_{(N,c)} \) if and only if we can solve the separation problem for \( AC_{(N,c)} \). Since \( P_{(N,c)} = \{ x \in AC_{(N,c)} : x(N) \leq c(N) \} \) holds, separation over \( P_{(N,c)} \) reduces to separation over \( AC_{(N,c)} \) plus an explicit check of a single inequality.

It remains to show how to solve the separation problem for \( AC_{(N,c)} \). For given \( \hat{x} \in \mathbb{R}^n \), we construct \( n \) points \( \hat{x}^k \in \mathbb{R}^n \) \( (k = 1, 2, \ldots, n) \) which are copies of \( \hat{x} \) except for \( \hat{x}^k := \min(\hat{x}_k, c(N) - \sum_{i \in N \setminus \{k\}} \hat{x}_i) \). Note that by construction \( \hat{x}^k \leq \hat{x} \) and \( \hat{x}^k(N) \leq c(N) \) hold. We then query a separation oracle of \( P_{(N,c)} \) with each \( \hat{x}^k \).

Suppose such a query yields \( \hat{x}^k(S) > c(S) \) for some \( S \subseteq N \). Due to \( \hat{x}^k(N) \leq c(N) \) we have \( S \neq N \). Moreover, \( \hat{x} \leq \hat{x}^k \) implies \( \hat{x}(S) > c(S) \), and we can return the same violated inequality.
Otherwise, we have \( \hat{x}^k \in P_{(N,c)} \) for all \( k \in N \) and claim \( \hat{x} \in AC_{(N,c)} \). To prove this claim we assume that, for the sake of contradiction, \( \hat{x}(S) > c(S) \) holds for some \( S \subseteq N \). Let \( k \in N \setminus S \). Since \( \hat{x}_k = \hat{x}_1 \) holds for all \( i \in S \), we have \( \hat{x}^k(S) = \hat{x}(S) > c(S) \). This contradicts the fact that \( \hat{x}^k \in P_{(N,c)} \) holds.

It turns out that almost the same result is true when we also require that there are no subsidies, that is \( x \geq 0 \). For linking the non-negative core to the non-negative almost core, it requires an assumption on the characteristic function.

\[
c(N \setminus \{k\}) \leq c(N) \quad \forall k \in N.
\]

This condition holds, for instance, for monotone functions \( c \), and implies that the core is contained in \( \mathbb{R}^n_{\geq 0} \) (see Lemma 2 and Theorem 1 in [14]).

**Theorem 3.** For a family of games \((N,c)\) satisfying (3), linear optimization problems over \( AC_{(N,c)} \cap \mathbb{R}^n_{\geq 0} \) can be solved in polynomial time if and only if linear optimization problems over \( P_{(N,c)} \cap \mathbb{R}^n_{\geq 0} \) can be solved in polynomial time.

The proof is a rather straightforward extension of that of Theorem 2, additionally making use of condition (3) to guarantee nonnegativity. We obtain an immediate consequence from these two theorems.

**Corollary 1.** For a family of games \((N,c)\) for which \( c(\cdot) \) is submodular (and (3) holds) one can find a (non-negative) optimal almost core allocation in polynomial time.

**Proof.** For submodular \( c(\cdot) \) one can optimize any linear objective function over \( P_{(N,c)} \) using the Greedy algorithm [13]. The result follows from Theorems 2 and 3.

These results only make statements about optimizing arbitrary objective vectors over these polyhedra. In particular we cannot draw conclusions about hardness of the computation of an almost core allocation. However, it is easy to see that this problem cannot be easier than deciding non-emptiness of the core.

**Theorem 4.** Consider a family of games \((N,c)\) for which deciding non-emptiness of the core is (co)NP-hard. Then finding an optimal almost core allocation is also (co)NP-hard.

**Proof.** By the premise of the theorem there exists a Karp reduction from some NP-hard problem \( \mathcal{P} \) to the non-emptiness decision problem for our family of games. The reduction turns (in polynomial time) an instance \( I \) of \( \mathcal{P} \) into a game \((N,c)\) such that \( I \) is a YES-instance (resp. NO-instance) if and only if \((N,c)\) has a non-empty core. The same reduction then works for the almost core since \( I \) is a YES-instance (resp. NO-instance) if and only if the almost core optimum is at least \( c(N) \).

It is well known that there exist games for which it is NP-hard to decide non-emptiness of the core, e.g., the weighted graph game [13]. Hence, we cannot hope for a polynomial-time algorithm that computes an optimal almost core allocation for arbitrary games.

In contrast, the maximization of \( x(N) \) becomes trivial for games \((N,c)\) with superadditive characteristic function \( c(\cdot) \), as the set of constraints \( x(\{i\}) \leq c(\{i\}) \), \( i = 1, \ldots, n \), already imply all other constraints \( x(S) \leq c(S) \), \( S \subseteq N \), and one can simply define \( x_i := c(\{i\}) \). In particular, \( x(N) \leq c(N) \) is implied and \( P_{(N,c)} = AC_{(N,c)} \). Generalizing, the same is true for classes of games where a polynomial number of constraints can be shown to be sufficient to define the complete core. As an example we mention matching games in undirected graphs [20], where the core is completely defined by the polynomially many core constraints induced by all edges of the underlying graph, as these can be shown to imply all other core constraints.

**Proposition 1.** Whenever \( P_{(N,c)} \) is described by a polynomial number of inequalities, finding an optimal (almost) core allocation can be done in polynomial time by linear programming.

Note that Proposition 1 includes supermodular cost functions. It is therefore interesting to note that for supermodular cost games, it is NP-hard to approximate the least core value \( \varepsilon^*_n \) better than a factor \( 17/16 \) [39].

It also turns out that condition (3) implies that the value of an almost core allocation cannot exceed that of a core allocation by much.
Proposition 2. Let \((N, c)\) be a game that satisfies (3). Then every \(x \in AC_{(N, c)}\) satisfies
\[
x(N) \leq (1 + \frac{1}{n - 1})c(N)
\].

Proof. Let \(x \in AC_{(N, c)}\). We obtain
\[
(n - 1) \cdot x(N) = \sum_{k \in N} x(N \setminus \{k\})
\leq \sum_{k \in N} c(N \setminus \{k\}) \leq \sum_{k \in N} c(N) = n \cdot c(N),
\]
where the first inequality follows from feasibility of \(x\) and the second follows from (3).

Condition (3) implies non-negativity for all core allocations and all optimal almost core allocations. However, this does not mean that a non-negativity requirement implies that the almost core optimum is close to \(c(N)\). In the next section, we will see that this gap can be arbitrarily large (see Proposition 3).

5 Minimum Spanning Tree Games and Approximation

In this section we address a well known special class of games known as minimum cost spanning tree (MST) games \([11, 5, 22]\), where the cost of the outside option for a set of agents is determined by the cost of a minimum cost spanning tree in the subgraph induced by these agents. MST games are known to have a non-empty core. Moreover, it is known that finding some element in the core is computationally easy and can be done by computing a minimum spanning tree \([22]\). The optimization problem that we address asks for the maximal amount that can be charged to the agents while no proper subset of agents would prefer the outside option.

From the more theoretical perspective, one can easily see that for MST games the cost function \(c(\cdot)\) is subadditive, yet it is not submodular in general; see \([27]\) for a characterization when it is. Recalling that the computation of maximum shareable costs can be done in polynomial time when \(c(\cdot)\) is submodular, it is a natural question to ask if this still holds for subadditive cost functions. In that respect, note that the weighted graph games as studied in \([13]\) have polynomial time algorithms to decide non-emptiness of the core whenever \(c(\cdot)\) is subadditive, hence Theorem 4 applied to weighted graphs games does not give an answer to this question. We will see that for MST games, maximizing shareable costs cannot be done efficiently unless P=NP.

In fact, one could define an even more general class of problems in the spirit of cooperative games with restricted coalition formation, by defining a (downward-closed) set system that describes all those subsets of agents that are able to cooperate and hence have access to an outside option, while all other subsets do not have that option. The almost core as studied in this paper arises as the special case where the set system is the \((n - 1)\)-uniform matroid.

5.1 Preliminaries

Let us first define the problem and recall what is known. We are given an edge-weighted, undirected graph \(G = (N \cup \{0\}, E)\) with non-negative edge weights \(w : E \rightarrow \mathbb{R}_{\geq 0}\), where node 0 is a special node referred to as “supplier” node. Without loss of generality we may assume that the graph is complete by adding dummy edges with large enough cost. The agents of the game are the vertices \(N\) of the graph, and the characteristic function of the game is given by costs of minimum spanning trees. That is, the cost of any subset of vertices \(S \subseteq N\) is defined as the cost of a minimum spanning tree on the subgraph induced by vertex set \(S \cup \{0\}\). So if we let \(T(S)\) be the set of spanning trees for the subgraph induced by vertex set \(S \cup \{0\}\), then the characteristic function is defined as:
\[
c(S) := \min_{T \in T(S)} \left\{ \sum_{e \in T} w(e) \right\}.
\]
Following [22], the associated monotonized minimum cost spanning tree game \((N, \bar{c})\) is obtained by defining the characteristic function using the monotonized cost function \(\bar{c}(S) := \min_{R \supseteq S} c(R)\). Indeed, note that \(\bar{c}(S) \leq c(R)\) for \(S \subseteq R\), and for the associated cores of these two games, we have that \(C_{(N, \bar{c})} \subseteq C_{(N, c)}\). Moreover, it is well known that the core of both games is non-empty, and a core allocation \(x \in C_{(N, c)}\) is obtained in polynomial time by just one minimum cost spanning tree computation: if \(T\) is some MST, let \(e_v \in T\) be the edge incident with \(v\) on the unique path from \(v\) to the supplier node 0 in \(T\), then letting
\[
x_v := w(e_v),
\]
one gets an element \(x\) in the core of the monotonized minimum cost spanning tree game \((N, \bar{c})\) [22], and hence also a core element for the game \((N, c)\). One convenient way of thinking about this core allocation is a run of Prim’s algorithm to compute minimum cost spanning trees [34]: starting to build the tree with vertex 0, whenever a new vertex \(v\) is added to the spanning tree constructed so far, \(v\) gets charged the cost of the edge \(e_v\) that connects \(v\).

In summary, computing some core allocation can be done efficiently, while linear optimization over the core of MST games is co-NP hard (under Turing reductions) [18]. We are interested in the same questions but for the case that the budget balance constraint is absent. So we seek solutions to the almost core maximization problem
\[
\max x(N) \text{ s.t. } x \in AC_{(N, c)}, \tag{4}
\]
when \(c(\cdot)\) is the characteristic function defined by costs of minimum spanning trees. The interpretation of the lacking constraint \(x(N) = c(N)\) is that the grand coalition cannot establish the solution with cost \(c(N)\) on its own, say by legal restrictions.

5.2 Computational Complexity

As a first result, and not surprising, linear optimization over the almost core is hard for MST games.

**Corollary 2.** For minimum cost spanning tree games \((N, c)\), a polynomial-time algorithm for linear optimization over \(AC_{(N, c)}\) would yield \(P=NP\).

**Proof.** The result follows from Theorem 2 and the fact that the membership problem for the core of \((N, c)\) is a coNP-hard problem for MST games [18].

What is more interesting is that optimizing \(1\cdot x\) over the almost core remains hard for MST games.

**Theorem 5.** Computing an optimal almost core allocation in \((N, c)\) for minimum cost spanning tree games is NP-hard.

**Proof.** Let \(\varepsilon^*\) be the largest \(\varepsilon\) for which the linear inequality system
\[
x(S) \leq (1-\varepsilon)c(S) \ \forall S \subseteq N, \quad x(N) = c(N) \tag{5}
\]
has a solution. In [19] it is shown that finding a feasible solution \(x\) for (5) with respect to \(\varepsilon^*\) is NP-hard. Note that in the reduction leading to this hardness result, \(c(N) > 0\). Then, given an optimum almost core allocation \(x^{AC}\), \(x^{AC}(N) \geq c(N) > 0\), and we can obtain \(\varepsilon^* := 1 - c(N)/x^{AC}(N)\). It is now easy to see that the vector \(x' := (1 - \varepsilon^*)x^{AC}\) is a feasible solution for (5). To see that the so-defined \(\varepsilon^*\) is indeed maximal, observe that scaling any feasible vector in (5) by \(1/(1-\varepsilon^*)\) yields an almost core allocation. Hence, computation of an almost core optimum for MST games yields a solution for an NP-hard problem.

Next, we note that in general, the almost core optimum may be arbitrarily larger than \(c(N)\) for MST games. This is remarkable in view of Proposition 2 which shows that under condition (5), any core allocation yields a good approximation for an optimal almost core allocation, as they differ by a factor at most \(n/(n-1)\). A fortiori, the same holds for the monotonized MST games \((N, \bar{c})\). For general MST games \((N, c)\), and without condition (5), this gap can be large.

**Proposition 3.** The almost core optimum can be arbitrarily larger than \(c(N)\), even for minimum cost spanning tree games and when we require that \(x \geq 0\).
Proof. Consider the instance depicted in Figure 1(a) for some value \( k > 0 \). Then \( c(N) = 0 \), while \( x = (0, 0, k) \) is an optimal non-negative almost core allocation with value \( k \).

In the following we consider problem (4) but with the added constraint that \( x \geq 0 \).

\[
\max x(N) \text{ s.t. } x \in AC_{(N,c)}, \text{ and } x \geq 0.
\]  \( (6) \)

The presence of the constraint \( x \geq 0 \) means that agents must not be subsidized. As can be seen from the example in Figure 1(b) such subsidies may indeed be necessary in the optimal solution to (4). First, we show that even this restricted problem remains NP-hard for MST games.

**Theorem 6.** Computing an optimal non-negative almost core allocation in (6) for minimum cost spanning tree games is NP-hard.

**Proof.** The claim follows by showing that problem (4) can be reduced in polynomial time to problem (6). The reduction works as follows. Given an instance of (4) with costs \( c \), we define a new cost function \( c'(e) := c(e) + M, e \in E \), for some large enough constant \( M \). Now consider an optimal solution \( x' \) to problem (6) for cost function \( c' \), and define \( x := x' - (M, \ldots, M) \). Now we have \( x(S) = x'(S) - |S| \cdot M \leq c'(S) - |S| \cdot M = c(S) \) for all \( S \subseteq N \), so \( x \) is feasible for problem (4). We claim that \( x \) is also optimal for problem (4). This is true since for any solution \( \tilde{x} \) that is optimal for (4), we have \( y' := \tilde{x} + (M, \ldots, M) \geq 0 \) for large enough \( M \), and \( y'(S) = \tilde{x}(S) + |S| \cdot M \leq c(S) + |S| \cdot M = c'(S) \), so \( y' \) is feasible for (6) with cost function \( c' \). Hence, \( \tilde{x}(N) > x(N) \) yields the contradiction \( y'(N) > x'(N) \). Finally, observe in problem (4) we maximize \( x(N) \), hence for any optimal solution \( \tilde{x} \) there exists \( M > 0 \) so that \( \tilde{x}_i \geq -M \) for all \( i \in N \), e.g. one can easily see that \( M := \sum_{j \in N} c(\{j\}) \) suffices. This is true because for an optimal solution \( \tilde{x} \), for all \( i \in N \) there exists some \( S \supseteq i \) so that constraint \( x(S) \leq c(S) \) is tight, and \( \tilde{x}_i \leq c(\{i\}) \) for all \( i \in S \).

REMARK. The above reduction of computing arbitrary allocations to computing non-negative allocations generalizes to general cost sharing games \( (N,c) \), by defining \( c'(S) := c(S) + |S| \cdot M \) for all subsets \( S \subseteq N \).

![Figure 1: MST games showing that the almost core optimum can be arbitrary far away from the core and showing that subsidies can be necessary.](image)

(a) MST game with large relative gap between almost core optimum and \( c(N) \). (b) MST game where subsidies are necessary for the almost core maximum.

### 5.3 2-Approximation Algorithm

We next propose the following polynomial time algorithm to compute an approximately optimal almost core allocation for Problem (6). For notational convenience, let us define for all \( K \subseteq N \)

\[
N_{-K} := N \setminus K,
\]

and write \( N_{-i} \) instead of \( N_{-\{i\}} \).

The backbone of Algorithm 1 is effectively Prim’s algorithm to compute a minimum cost spanning tree \( [34] \). The additional line \( [5] \) yields the core allocation by Granot and Huberman \( [22] \), which we extend by adding lines \( [7] \) and \( [8] \).
Algorithm 1: Approximation algorithm for the almost core maximization problem \([\text{ALG}]\) for MST games

Input: agents \(N\), edge set \(E\) of complete graph on \(N \cup \{0\}\) and edge weights \(w : E \to \mathbb{R}_{\geq 0}\)  
Output: Almost core allocation \(x\).

Let us first collect some basic properties of Algorithm 1. Henceforth, we assume w.l.o.g. that the agents get assigned their cost shares in order \(1, \ldots, n\) (so that \(\ell = n\) in lines 7 and 8). We denote by \(x^{\text{ALG}}\) a solution computed by Algorithm 1.

Lemma 1. We have that \(x^{\text{ALG}}(I_k) = c(I_k)\) for all \(k = 1, \ldots, n-1\), and for all \(S \subseteq \{1, \ldots, n-1\}\) we have \(x^{\text{ALG}}(S) \leq c(S)\).

Proof. The first claim follows directly because Algorithm 1 equals Prim’s algorithm to compute a minimum cost spanning tree on the vertex set \(\{0, \ldots, n-1\}\). Hence by correctness of Prim’s algorithm \([34]\), \(x^{\text{ALG}}(I_k) = c(I_k)\). The second claim follows by \([22]\) Thm. 3, since the cost allocation for agents \(\{1, \ldots, n-1\}\) is the same as in \([22]\). □

Lemma 2. Suppose \(x^{\text{ALG}}(S) > c(S)\) for some set \(S\) with \(n \in S \not\subseteq N\). Then there is a superset \(T \supseteq S\) with \(|T| = n-1\) such that \(x^{\text{ALG}}(T) > c(T)\).

Proof. Recall the agents got assigned their cost shares in order \(1, \ldots, n\). Define \(k := \max\{i \mid i \notin S\}\) to be the largest index of a agent not in \(S\). Let \(i_1, \ldots, i_k\) be the set of agents so that \(N_{-k} = S \cup \{i_1, \ldots, i_k\}\) and w.l.o.g. \(i_1 < \cdots < i_k\). We show that \(x^{\text{ALG}}(S) > c(S)\) implies \(x^{\text{ALG}}(S \cup \{i_1\}) > c(S \cup \{i_1\})\). Then repeating the same argument, we inductively arrive at the conclusion that \(x^{\text{ALG}}(N_{-k}) > c(N_{-k})\). So observe that

\[
x^{\text{ALG}}(S \cup \{i_1\}) = x^{\text{ALG}}(S) + x_{i_1} \geq c(S) + x_{i_1},
\]

and \(c(S)\) is the cost of a minimum cost spanning tree for \(S\), call it MST\((S)\). Moreover, as \(i_1 \neq n\), \(x_{i_1}\) is the cost of the edge, call it \(e\), that the algorithm used to connect agent \(i_1\). We claim that MST\((S) \cup \{e\}\) is a tree spanning vertices \(S \cup \{0, i_1\}\), hence \(c(S) + x_{i_1}\) is the cost of some tree spanning \(S \cup \{0, i_1\}\). Then, as required we get

\[
x^{\text{ALG}}(S \cup \{i_1\}) > c(S) + x_{i_1} \geq c(S \cup \{i_1\}),
\]

because \(c(S \cup \{i_1\})\) is the cost of a minimum cost tree spanning \(S \cup \{0, i_1\}\). If MST\((S) \cup \{e\}\) was not a spanning tree for vertices \(S \cup \{0, i_1\}\), then edge \(e\) would connect \(i_1\) to some vertex outside \(S\), but this contradicts the choice of \(i_1\) as the vertex outside \(S\) with smallest index. □

Lemma 3. We have \(x^{\text{ALG}} \geq 0\).

Proof. Recall that in minimum cost spanning tree games \([11, 22]\), the weight of edges are non-negative. Since Algorithm 1 computes the allocation for agents in line 8 by the edge weight of the first edge on the unique path to 0, there is \(x^{\text{ALG}}_k \geq 0\) for all \(k = 1, 2, \ldots, n-1\). So we only need to argue about \(x^{\text{ALG}}_n\). To that end, note that an equivalent definition of \(x^{\text{ALG}}_n\) in line 8 of the algorithm is

\[
\max \ x_n \text{ s.t. } x_n \leq c(N_{-k}) - x^{\text{ALG}}(N \setminus \{k, n\}) \text{ for all } k = 1, \ldots, n-1.
\]

We claim that \(\bar{x}_n := c(N) - c(N_{-(n-1)}) \geq 0\) is a feasible solution to this maximization problem, hence the actual value of \(x^{\text{ALG}}_n\) after the update in line 8 can only be larger, and therefore in particular it is non-negative. First, note that after, \(\bar{x}_n \geq 0\), as this is the cost of the last edge that Prim’s algorithm uses to
connect the final vertex \( n \) to the minimum cost spanning tree. That \( \hat{x}_n \) is feasible in (7) follows from the fact that \( \hat{x}_n \) is the cost share that is assigned to agent \( n \) in the core allocation of \([22]\). Indeed, letting \( \hat{x} \) be equal to \( x \) except for \( \hat{x}_n = c(N) - c(N_{(n-1)}) \), we have that \( \hat{x} \) is precisely the cost allocation as proposed in \([22]\).

By the fact that this yields a core allocation, we have that \( \hat{x}(S) \leq c(S) \) for all \( S \subseteq N \), so in particular for all \( k = 1, \ldots, n-1 \),

\[
\hat{x}_n + x^{\text{ALG}}(N \setminus \{k, n\}) = \hat{x}(N_{k}) \leq c(N_{k}),
\]

and hence the claim follows.

**Theorem 7.** Algorithm \([1]\) is a 2-approximation for the almost core maximization problem \([6]\) for minimum cost spanning tree games, and this performance bound is tight for Algorithm \([1]\).

**Proof.** Denote by \( x^{\text{ALG}} \) a solution by Algorithm \([1]\). We first argue that Algorithm \([1]\) yields a feasible solution. For \( S \not\ni n \), this follows from Lemma \([1]\). For \( S \ni n \), assume \( x(S) > c(S) \). Then Lemma \([2]\) yields that there exists some \( N_{k} \ni n \) with \( x^{\text{ALG}}(N_{k}) > c(N_{k}) \). However by definition of \( x_{n} \) in line 8 of the algorithm, we have for all \( k = 1, \ldots, n-1 \)

\[
x_{n}^{\text{ALG}} \leq c(N_{k}) - x^{\text{ALG}}(N \setminus \{k, n\}),
\]

which yields a contradiction to \( x^{\text{ALG}}(N_{k}) > c(N_{k}) \).

To show that the performance guarantee is indeed 2, let \( x^{\text{OPT}} \) be some optimal solution to the almost core maximization problem. Let \( k^{*} \in N_{n} \) be the index for which the minimum in line 8 is attained. Observe that \( x_{n}^{\text{ALG}} \) is updated such that \( x^{\text{ALG}}(N_{k^{*}}) = c(N_{k^{*}}) \) holds. Then by non-negativity of \( x^{\text{OPT}} \) and because of Lemma \([3]\)

\[
x_{n}^{\text{OPT}} \leq x^{\text{OPT}}(N_{k^{*}}) \leq c(N_{k^{*}}) = x^{\text{ALG}}(N_{k^{*}}) \leq x^{\text{ALG}}(N).
\]

Moreover, by definition of \( x^{\text{ALG}} \), we have \( x^{\text{ALG}}(N_{n}) = c(N_{n}) \), and by Lemma \([3]\)

\[
x^{\text{OPT}}(N_{n}) \leq c(N_{n}) = x^{\text{ALG}}(N_{n}) \leq x^{\text{ALG}}(N).
\]

Hence we get

\[
x^{\text{OPT}}(N) = x_{n}^{\text{OPT}} + x^{\text{OPT}}(N_{n}) \leq 2x^{\text{ALG}}(N).
\]

To see that the performance bound 2 is tight for Algorithm \([1]\) consider the instance in Figure 2. Here, Algorithm \([1]\) computes the solution \( x^{\text{ALG}} = (1, 0, \varepsilon) \)

![Figure 2: MST game showing that the analysis of Algorithm \([1]\) cannot be improved.](image)

with value \( 1 + \varepsilon \), as the order in which agents get assigned their cost shares is 1, 2, 3, and in line 8 of the algorithm we get \( x_{3}^{\text{ALG}} = c(\{1, 3\}) - x_{1} = (1 + \varepsilon) - 1 = \varepsilon \). An almost core optimum solution would be \( x^{\text{OPT}} = (0, 1, 1) \) with value 2.

As a matter of fact, the requirement that \( x \geq 0 \) is important in Theorem 7. Allowing that agents receive subsidies, so allowing \( x_{i} < 0 \) for some agents \( i \), the above algorithm does not provide any approximation guarantee. To see that, consider the instance given in Figure \([1]\) and observe that Algorithm \([1]\) yields a cost allocation \( x^{\text{ALG}} = (0, 0, 0) \), while \( x = (-k, k, k) \) is a feasible solution for the almost core. Note that the corresponding monotonized MST game is trivial as \( c(S) = 0 \) for all agent sets \( S \), and the almost core optimum is 0 with an optimal allocation \((0, 0, 0)\).
6 Conclusions

In the literature, one also finds minimum cost spanning tree games defined as profit sharing games, where one defines the value of a coalition $S$ by the cost savings that it can realize in comparison to the situation where all agents in $S$ connect directly to the source,

$$v(S) := \sum_{i \in S} c(\{i\}) - c(S).$$

Then the core constraints, for profit shares $x^v \in \mathbb{R}^n$, are $x^v(S) \geq v(S)$. It is not hard to see that all our results also hold for that version of the problem via the simple transformation $x^v_i := c(\{i\}) - x_i$. In particular, note that for value games all feasible solutions $x^v$ are non-negative, as core stability requires that $x^v_i \geq v(\{i\}) \geq 0$.

Our results imply NP-hardness for computation of minimum profit shares that are coalitionally stable, and the corresponding profit version of Algorithm 1 can be shown to yield a 2-approximation, which also can be shown to be tight.

We finally collect some problems which we believe are interesting. First, we would like to gain more insight into the computational complexity for the almost core problem (1), also for other classes of games. Moreover, we could give a 2-approximation for cost MST games under the additional assumption that subsidies are not allowed. It would be interesting to extend this result to the general, unconstrained case. Also giving lower bounds on the approximability does seem plausible, as the “hard cases” for maximizing shareable costs are those where the minimum cost spanning tree is a (Hamiltonian) path. Finally, we believe the generalization to problems with more general forms of restricted cooperation may lead to interesting combinatorial problems, specifically for MST games.

Acknowledgements

Rong Zou and Boyue Lin acknowledge the support of the China Scholarship Council (Grants No. 202006290073, 202106290010).

References


