Approximate Solutions To Max-Min Fair and Proportionally Fair Allocations of Indivisible Goods

Nhan-Tam Nguyen
Heinrich-Heine-Universität
Düsseldorf, Germany
nguyen@cs.uni-duesseldorf.de

Trung Thanh Nguyen
New York University
Abu Dhabi, UAE
ttn1@nyu.edu

Jörg Rothe
Heinrich-Heine-Universität
Düsseldorf, Germany
rothe@cs.uni-duesseldorf.de

ABSTRACT

Max-min fair allocations and proportionally fair allocations are desirable outcomes in a fair division of indivisible goods. Unfortunately, such allocations do not always exist, not even in very simple settings with few agents. A natural question is to ask about the largest value c for which there is an allocation such that every agent has utility of at least c times her fair share. Our goal is to approximate this value c. For additive utilities, we show that when the number of agents is fixed, one can approximate c by a polynomial-time approximation scheme. We show that the case when utility functions are defined based on scoring vectors (binary, Borda, and lexicographic vectors) is tractable. For 2-additive functions, we show that a bounded constant for max-min fair allocations does not exist, not even when there are only two agents. We explore a class of symmetric submodular functions for which a tight $\frac{1}{2}$-max-min fair allocation exists and show how it can be approximated within a factor of $\frac{1}{4}$.

Keywords

Fair allocation; max-min share; proportional share; approximation scheme

1. INTRODUCTION

Fair division of indivisible goods is an important field (see, e.g., the book chapters by Bouveret et al. [12] and Lang and Rothe [31]) that deals with the allocation of indivisible goods (or items or objects) to some agents. The goal is to find an assignment of items amongst agents that meets a certain notion of fairness. Among others, the notion of max-min (fair) share proposed by Budish [16] and the notion of proportional share originally introduced by Steinhaus [36] have gained increasing interest in the AI and theoretical computer science literature [15, 35, 1, 32, 19, 26, 30, 18, 4]. The max-min share of an agent is defined as the bundle that the agent can guarantee for herself when partitioning the items into bundles but choosing last. In a proportionally fair allocation between n agents, each agent receives a bundle of value at least $\frac{1}{n}$ of the whole. These two fairness criteria have attractive theoretical and practical properties (see the references cited above). Unfortunately, as argued in the literature (see, for example, [32, 35, 30]), an allocation satisfying any of the these fairness notions is not guaranteed to exist in general. Hence, in this paper we aim at answering the following question: *Given a problem instance I, what is the largest possible number c for which there is a c-max-min (c-proportionally) fair allocation, which means that every agent receives a subset of items worth at least c times her max-min (proportional) share?* Our question above is similar but slightly different from the one addressed by Procaccia and Wang [35] who ask whether c is lower-bounded by some constant for every problem instance. It is obvious that, given a problem instance and a number k, checking if $c \geq k$ is at least as hard as the problem of checking the existence of a max-min (proportionally) fair allocation (for $k = 1$). In fact, the former problem was known to be NP-hard for proportionally fair allocations [32, 15]. This means that computing c exactly is as hard as solving an NP-hard problem. In this paper, we are therefore interested in approximating c via polynomial-time algorithms. We focus on both additive and nonadditive utility functions, and explore utility functions with additional structure for which c can be solved exactly or approximately within a small constant.

Contribution.

Our first contribution is a general technique that gives a polynomial-time approximation scheme (PTAS) for approximating the value c under the assumption that we have a fixed number of agents with additive utilities. This technique works as long as there is an exact polynomial-time algorithm or a PTAS for the fair share of an individual agent. Since the max-min share and the proportional share criterion satisfy this condition, this result subsumes a result by Aziz et al. [6, 5]. Their method essentially relies on a reduction to the machine scheduling problem which is already known to have a PTAS. In contrast, our method solves the problem in a direct way, by formulating it as a binary multi-criteria optimization problem and extending an idea used by Erlebach et al. [22]. The key ingredient is a novel technical lemma, which for every $\varepsilon > 0$ guarantees the existence of a polynomial-size set that contains for every allocation another approximate allocation such that every agent loses at most an ε-fraction of utility. In fact, our technique can potentially be applied to other variants of fair allocation problems such as the *min-max fair allocation* problem [15] (using a PTAS by Hochbaum and Shmoys [29]), or more general, the *rank-weighted utilitarianism* problem studied by Heinen et al. [26].

We investigate special cases of additive utilities functions where the value of c can be computed exactly or lower-bounded by some constant. When the utilities of items are restricted to the domain $\{0, 1\}$, we prove that, given a number k, one can check if $c \geq k$ in polynomial time. When the number of agents is fixed, we show that the value of c can be computed in polynomial time for both *Borda* and *lexicographic scoring vectors*.

Finally, we consider the problem of computing max-min fair allocations when utility functions are represented in 2-additive form.

Appears in: *Proc. of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017), S. Das, E. Durfee, K. Larson, M. Winikoff (eds.), May 8–12, 2017, São Paulo, Brazil.*
Copyright © 2017, International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

We first show that in general c cannot be lower-bounded by any constant for every problem instance (even with two agents only). Furthermore, we prove that $c \geq 1/2$ for every problem instance with two agents having symmetric submodular functions, and this bound is tight in the sense that there exists an instance for which there is no $(1/2+\varepsilon)$-max-min fair allocation, for any $\varepsilon > 0$. Also, we provide an algorithm that outputs $1/4$-max-min fair allocations in polynomial time.

Related Work.

Most work so far has been devoted to finding the lower bounds of c for additive utilities. For max-min fair allocations, Procaccia and Wang [35] show that $c \geq 2/3$ is guaranteed for every problem instance. By redesigning parts of the algorithm of Procaccia and Wang [35], Amanatidis et al. [1] showed that a $2/3$-max-min fair allocation can be found in polynomial time. For a small number of agents, the problem is understood better. For instance, a *cut-and-choose* protocol will result in a max-min fair allocation for two agents and gives a lower bound of 1 for c in this case. For three agents, it was shown by Amanatidis et al. [1] that $c \geq 7/8$. It still remains open whether the lower bound of $7/8$ can be further improved for a higher constant number of agents. Very recently, Aziz et al. [6] have given a polynomial-time approximation scheme for computing an optimal c-max-min fair allocation, assuming that the number of agents is fixed. This result is closely related to some of our results, as we consider this problem with respect to *proportional* fairness. For the special case when the utility of every single item is restricted to the domain $\{0, 1, 2\}$, a max-min fair allocation (if it exists) is computationally easy to find [1]. This extends an earlier result of Bouveret and Lemaître [15] for binary utility functions. When the utility functions are defined by a scoring vector, Bouveret and Lemaître [15] showed that there always exists a max-min fair allocation.

For proportionally fair allocations, the problem of checking if $c \geq 1$ is NP-complete, even with two identical agents (Bouveret and Lemaître [15]). A first lower bound for c, which is essentially a function depending both on the number of agents and on the maximum value of any agent for a single good, is provided by Hill [28]. Markakis and Psomas [32] presented a polynomial-time algorithm for constructing an allocation with respect to this lower bound. An improvement of this bound was then proposed by Gourvès et al. [24, 25]. Darmann and Klamler [19] considered Borda-scoring-based utility functions and studied the special case when the number of items is a multiple of the number of agents. They characterized under which conditions proportional fair allocations exist and can be found easily.

2. MODEL

Let $\mathcal{A} = \{1, \ldots, n\}$ be a set of agents, $\mathcal{O} = \{o_1, \ldots, o_m\}$ a set of indivisible and nonshareable goods (or items or objects), and let $u_i : 2^{\mathcal{O}} \to \mathbb{Q}^+$ be the nonnegative utility function of agent $i \in \mathcal{A}$. We assume that every utility function u_i is represented in the k-additive form for some constant $k \geq 1$ ([17]). In a formal way, such a function u_i is associated with a weight mapping δ_i from $2^{\mathcal{O}}$ to $\mathbb{Q}^+$, which assigns zero weight to every subset S of cardinality $> k$ (i.e., $\delta_i(S) = 0$). Then, for each subset $S \subseteq \mathcal{O}$, $u_i(S)$ is defined as $\sum_{T \subseteq S} \delta_i(T)$. We focus on the cases when $k = 1$ (note that 1-additive functions are the additive functions, i.e., $u_i(S) = \sum_{o \in S} u_i(o)$ for any $S \subseteq \mathcal{O}$) or $k = 2$. Let $\mathcal{U} = \{u_1, \ldots, u_n\}$. We call $(\mathcal{A}, \mathcal{O}, \mathcal{U})$ an *allocation setting*. An *allocation* is a partition $\pi = (\pi_1, \ldots, \pi_n)$ of $\mathcal{O}$, where π_i is agent i's share. We denote by Π the set of all allocations.

We consider two fairness criteria, max-min share and proportional share, which are defined formally below. Proportional share is defined only for additive utility functions, though.

DEFINITION 1. *Given an allocation setting $(\mathcal{A}, \mathcal{O}, \mathcal{U})$, define agent i's* max-min share *as*

$$\mathrm{MMS}_i = \max_{\pi \in \Pi} \min_{j \in \mathcal{A}} u_i(\pi_j).$$

An allocation π is a max-min fair allocation *if $u_i(\pi_i) \geq \mathrm{MMS}_i$ for all $i \in \mathcal{A}$.*

DEFINITION 2. *Given an allocation setting $(\mathcal{A}, \mathcal{O}, \mathcal{U})$ where all the utility function are additive, define agent i's* proportional share *as*

$$\mathrm{PS}_i = \frac{1}{n} u_i(\mathcal{O}).$$

An allocation π is proportionally fair *if $u_i(\pi_i) \geq \mathrm{PS}_i$ for all $i \in \mathcal{A}$.*

Let I be an instance with additive utility functions. We assume, w.l.o.g., that the max-min share (proportional share) of every agent is positive and that there is at least one allocation in which the utility of every agent is positive. This ensures that the value of c is nonzero and bounded. Concretely, denote by $c_M(I)$ ($c_P(I)$) the largest value such that there exists an allocation in which the utility of every agent is at least $c_M(I)$ ($c_P(I)$) of her max-min share (proportional share). For an allocation $\pi = (\pi_1, \ldots, \pi_n)$, we define

$$\alpha(\pi) = \min_{i \in \mathcal{A}} \frac{u_i(\pi_i)}{\mathrm{MMS}_i} \quad \text{and} \quad \beta(\pi) = \min_{i \in \mathcal{A}} \frac{u_i(\pi_i)}{\mathrm{PS}_i}$$

Now we can define $c_M(I)$ and $c_P(I)$ as follows:

$$c_M(I) = \max_{\pi \in \Pi} \alpha(\pi) \quad \text{and} \quad c_P(I) = \max_{\pi \in \Pi} \beta(\pi).$$

We will simply write c_M and c_P when the instance I is clear from the context. If $c_M \geq 1$ ($c_P \geq 1$) then a max-min fair allocation (a proportionally fair allocation) for I exists.

For nonadditive utility functions, the assumption above is strong and, in fact, hard to verify in general. Hence, in this paper (see Section 5) we are only concerned with the problem of checking whether there is a c-max-min fair allocation for every instance with nonadditive utility functions, for some small constant c.

We will also consider the *egalitarian welfare* of an allocation π, which is the utility of the worst-off agents in that allocation, or more formally, is defined as $\min_{i \in \mathcal{A}} u_i(\pi_i)$.

3. ADDITIVE PREFERENCES

In this section we exemplify our technique for the max-min share and for the proportional share. Assume that every agent has an additive utility function over the bundles of items with $u_i(o) \geq 1$ for every $o \in \mathcal{O}$ (by appropriate scaling). Given an instance $I = (\mathcal{A}, \mathcal{O}, \mathcal{U})$, where $\|\mathcal{A}\|$ is fixed, we show that both c_M and c_P can be approximated in polynomial time to within a factor of $1 - \varepsilon$, for any fixed $\varepsilon > 0$. That is, we present a PTAS for this problem.

In order to find a proportionally (max-min) fair allocation of a given problem instance I, one idea is to try to allocate items to agents in such a way that all agents' utilities are maximized simultaneously. Mathematically, this can be modeled by a multi-criteria binary linear program (MBLP) as follows:

$$\begin{aligned} \text{maximize} \quad & \left\{ \sum_{j=1}^{m} u_{1j} x_{1j}, \ldots, \sum_{j=1}^{m} u_{nj} x_{nj} \right\} \\ \text{subject to} \quad & \sum_{i=1}^{n} x_{ij} \leq 1, \quad j \in [m], \\ & x_{ij} \in \{0, 1\}, \quad i \in [n], j \in [m], \end{aligned}$$

where $[m]$ is a shorthand for $\{1, \ldots, m\}$, u_{ij} denotes the value of item o_j to agent i, and the variable $x_{ij} = 1$ if and only if item o_j is assigned to agent i, for all i, j.

Due to possible conflicts between agents, there does not exist an allocation that simultaneously optimizes each agent's utility. One instead constructs in time polynomial in the size of the instance a *small* subset of allocations, denoted by $\mathcal{P}_\varepsilon$, which has the following property: For any allocation $\pi \in \Pi$, there exists an allocation $\pi' \in \mathcal{P}_\varepsilon$ such that $u_i(\pi'_i) \geq (1-\varepsilon)u_i(\pi_i)$ for all $i \in \mathcal{A}$, for any fixed $\varepsilon > 0$. Lemma 1 below will show how to obtain such a set $\mathcal{P}_\varepsilon$.

LEMMA 1. *For any fixed $\varepsilon > 0$, $\mathcal{P}_\varepsilon$ can be found in time polynomial in the size of the input.*

PROOF. The construction of $\mathcal{P}_\varepsilon$ is done by Algorithm 1 below. For a high-level description, note that the implementation is divided into two phases. In the first phase, the algorithm guesses the lower bounds on the values of the bundles assigned to each of the first $n-1$ agents. This should be done carefully so as to guarantee that the numbers of lower bounds is bounded by the size of the instance. Having these lower bounds, in the second phase the algorithm tries to allocate items to agents in such a way that the utility of the last agent is maximized. This can be formulated as a binary linear program (BLP), which is then solved approximately. All the (approximate) solutions will be collected in a set $\mathcal{P}_\varepsilon$.

In the guessing phase, we consider all possibilities of choosing $n-1$ lower bounds, one for each of the first $n-1$ agents. For each $i \in \mathcal{A}$, we assume, w.l.o.g., that $u_i(\mathcal{O}) > 1$, and we let $B_i = \left\lceil \log_{1+\varepsilon/2} u_i(\mathcal{O}) \right\rceil$ and define the lower bounds with respect to i as:

$$LB_i(z) = \left(1 + \frac{\varepsilon}{2}\right)^{z-1}, \text{ for } z \in \mathbb{Z} \cap [1, B_i + 1]. \quad (1)$$

A lower bound of zero is also possible for an agent. Since we can ignore such an agent, we are not going to consider this case from now on.

For each agent i, we consider in total $B_i + 1$ lower bounds, and thus the number of possibilities of choosing $n-1$ lower bounds, one for each of the first $n-1$ agents, is bounded by $O(B_1 \cdots B_{n-1})$.

In the second phase, assume that the lower bounds of the first $n-1$ agents are described by $LB_1(z_1), \ldots, LB_{n-1}(z_{n-1})$, for some $z_1, \ldots, z_{n-1}$. We consider the following binary linear program:

$$\begin{aligned}
\text{maximize} \quad & \sum_{j=1}^{m} u_{nj} x_{nj} \\
\text{subject to} \quad & \sum_{j=1}^{m} u_{ij} x_{ij} \geq LB_i(z_i), && i \in [n-1], \\
& \sum_{i=1}^{n} x_{ij} \leq 1, && j \in [m], \\
& x_{ij} \in \{0,1\} && i \in [n], j \in [m].
\end{aligned}$$

In order to find approximate solutions to this BLP, we rely on the polynomial solvability of the convex relaxation of a BLP, called LP, which is obtained by replacing the constraints $x_{ij} \in \{0,1\}$ by the weaker constraints $x_{ij} \in [0,1]$:

$$\begin{aligned}
\text{maximize} \quad & \sum_{j=1}^{m} u_{nj} x_{nj} \\
\text{subject to} \quad & \sum_{j=1}^{m} u_{ij} x_{ij} \geq LB_i(z_i), && i \in [n-1], \quad (2) \\
& \sum_{i=1}^{n} x_{ij} \leq 1, && j \in [m], \quad (3) \\
& x_{ij} \geq 0 && i \in [n], j \in [m]. \quad (4)
\end{aligned}$$

Let $p = \lceil 4(n-1)/\varepsilon \rceil$. The algorithm guesses, for each of the agents, the p highest-utility items assigned to her. We denote by G_i the set of the p highest-utility items received by agent i. If $G_1, \ldots, G_n$ are mutually disjoint sets, we define the values of x_{ij} accordingly:

$$x_{ij} = \begin{cases} 1 & \text{if} \quad j \in G_i, \\ 0 & \text{if} \quad j \notin G_i \quad \text{and } u_{ij} > \min\{u_{ik} \mid k \in G_i\}. \end{cases} \quad (5)$$

Let $G = \{G_1, \ldots, G_n\}$ and denote by LP(G) the resulting linear program according to setting (5).

Algorithm 1

```
p ← ⌈4(n−1)/ε⌉
for each (LB_1(z_1), ..., LB_{n−1}(z_{n−1})) do
    for each G = (G_1, ..., G_n), ||G_i|| ≤ p do
        if G_i ∩ G_k = ∅ for all i ≠ k then
            Compute an optimal basic solution x^G of LP(G)
            x̄^G_ij ← ⌊x^G_ij⌋ for all i ∈ A, j ∈ [m]
            for i ∈ A do
                π_i ← {j | x̄^G_ij = 1, for j ∈ [m]}
        P_ε ← P_ε ∪ {(π_1, ..., π_n)}
return Pε
```

Now let $\mathcal{P}_\varepsilon$ be the output of Algorithm 1. Let π be an arbitrary allocation to the instance I. We will prove that there always exists a solution $\pi' \in \mathcal{P}_\varepsilon$ such that

$$u_i(\pi'_i) \geq (1-\varepsilon)u_i(\pi_i), \quad (6)$$

for every $i \in [n]$. Let us define

$$\hat{z}_i = \max\{z_i \in \mathbb{Z} \cap [1, B_i + 1] \mid LB_i(z_i) \leq u_i(\pi_i)\}, \quad (7)$$

for all $i = 1, \ldots, n-1$. By the definition of $\hat{z}_i$ and of $LB_i(z_i)$, we must have that

$$LB_i(\hat{z}_i) \leq u_i(\pi_i) < LB_i(\hat{z}_i + 1) = (1 + \varepsilon/2)LB_i(\hat{z}_i), \quad (8)$$

for all $i = 1, \ldots, n-1$.

For each $i \in \mathcal{A}$, let G_i by the set that contains $\min\{p, \|\pi_i\|\}$ highest-utility items in π_i. Let $G = (G_1, \ldots, G_n)$. Consider the linear program LP(G) with the lower bounds $LB_1(\hat{z}_1), \ldots, LB_{n-1}(\hat{z}_{n-1})$ defined as above. Note that LP(G) is feasible, since π constitutes a feasible solution by the choice of $LB_i(\hat{z}_i)$, $1 \leq i \leq n-1$. Let x^G be a basic optimal solution of LP(G) and let $\bar{x}^G$ be the integral solution obtained by rounding down x^G. Let $\pi' = (\pi'_1, \ldots, \pi'_n)$ be an allocation, where $\pi'_i = \{o_j \mid \bar{x}^G_{ij} = 1, \text{ for } j \in [m]\}$. We now prove that the inequalities (6) hold. We need the following facts in our analysis (see the appendix for the detailed proofs).

FACT 1. *x^G has at most $2(n-1)$ fractional components.*

FACT 2. *For every agent $i \in \mathcal{A}$, it holds that*

$$\sum_{j \in [m]} u_{ij} x^G_{ij} \geq \frac{1}{1 + \varepsilon/2} u_i(\pi_i).$$

If $\|\pi_i\| \leq p$ for some $i \in [n]$ (i.e., π_i has at most p items), the rounded solution π'_i will contain π_i. Let T be a subset of agents such that each of them is assigned more than p items in π. It suffices to prove that (6) holds for these agents. For $i \in T$, let $u_{i\ell} = u_i(o_\ell)$ be the smallest utility among the p items of highest utility in π_i, for some o_ℓ. Then $\sum_{j \in [m]} u_{ij} x^G_{ij} \geq p u_{i\ell}$. On the other hand, since $\bar{x}^G$ is obtained by rounding down x^G, and from Fact 1, which says that x^G has at most $2(n-1)$ fractional components, it follows that

$$u_i(\pi'_i) \geq \sum_{j \in [m]} u_{ij} x^G_{ij} - \frac{2(n-1)}{p} p u_{i\ell} \geq \left(1 - \frac{\varepsilon}{2}\right) \sum_{j \in [m]} u_{ij} x^G_{ij}.$$

By Fact 2, we have

$$u_i(\pi'_i) \geq \frac{1-\varepsilon/2}{1+\varepsilon/2} u_i(\pi_i) \geq (1-\varepsilon) u_i(\pi_i),$$

for all $i \in \mathcal{A}$.

To complete the proof of Lemma 1, we show that the running time of Algorithm 1 is polynomial in the size of I. It is not hard to see that the running time of Algorithm 1 is dominated by the following steps:

- guessing the lower bounds $LB_i(z_i), 1 \leq i \leq n-1$;
- guessing the sets $G_i, 1 \leq i \leq n$, each has at most p items; and
- solving the linear program in line 5 of Algorithm 1.

As argued earlier, the number of possible lower bounds is bounded by $O(B_1 \cdots B_{n-1})$ and thus is polynomial in the size of the input since n is constant and $B_i \in O(|I|)$ for all $i \in [n-1]$. The number of possible sets G_i for n agents is $O(m^{np})$ and is polynomial, as p and n are constants. Finally, solving a linear program is well known to be feasible in polynomial time ([33]). The proof of the lemma is complete. ❑

THEOREM 1. *If the number of agents is fixed, there is a polynomial-time algorithm that produces an allocation such that each agent i is assigned a bundle of value at least $(1-\varepsilon)c_M \cdot \mathrm{MMS}_i$, for any constant $\varepsilon > 0$.*

PROOF. Consider Algorithm 2 below:

Algorithm 2

1: Compute $\mathcal{P}_\varepsilon$ using Algorithm 1
2: Compute $k_i \leftarrow \lceil (1-\varepsilon)\mathrm{MMS}_i \rceil$ for all i
3: $\pi^* \leftarrow \mathrm{argmax}_{\pi \in \mathcal{P}_\varepsilon} \min_{i \in \mathcal{A}} \dfrac{u_i(\pi_i)}{k_i}$
4: **return** π^*

Note that for any agent i, MMS_i can be approximated within a factor of $1-\varepsilon$, for any $\varepsilon > 0$ (see the work of Woeginger [37]). Now assume that π^* is output by Algorithm 2, and let π be an allocation such that $u_i(\pi_i) \geq c_M \cdot \mathrm{MMS}_i$, for all $i \in \mathcal{A}$. There must be an allocation $\pi' = (\pi'_1, \ldots, \pi'_n) \in \mathcal{P}_\varepsilon$ such that

$$u_i(\pi'_i) \geq (1-\varepsilon) \cdot u_i(\pi_i) \geq c_M(1-\varepsilon) \cdot \mathrm{MMS}_i$$

for all $i \in \mathcal{A}$. On the other hand, we have

$$\min_{i \in \mathcal{A}} \left\{ \frac{u_i(\pi^*_i)}{k_i} \right\} \geq \min_{i \in \mathcal{A}} \left\{ \frac{u_i(\pi'_i)}{k_i} \right\} \geq c_M.$$

It follows that $u_i(\pi^*_i) \geq k_i \cdot c_M \geq (1-\varepsilon)c_M \cdot \mathrm{MMS}_i$, for all $i \in \mathcal{A}$. This completes the proof of Theorem 1. ❑

THEOREM 2. *If the number of agents is fixed, there is a polynomial-time algorithm that produces an allocation such that each agent i is assigned a bundle of value at least $(1-\varepsilon)c_P \cdot \mathrm{PS}_i$, for any constant $\varepsilon > 0$.*

PROOF. The proof is similar to the one of Theorem 1, except that k_i is set to PS_i for all $i \in \mathcal{A}$ in Algorithm 2. ❑

4. RESTRICTIONS ON ADDITIVITY

We study restrictions to binary and to scoring-based utilities.

4.1 Binary Utilities

We consider the case when utilities are either 0 or 1 for every item. Bouveret and Lemaître [15] showed that a max-min fair allocation can be found via a simple picking-sequence protocol. However, this is not the case with proportionally fair allocations.[1]

THEOREM 3. *Given a number k, checking if $c_P \geq k$ can be done in polynomial time if the value of each item belongs to the domain $\{0,1\}$.*

PROOF. We follow a linear-programming-based approach. Let I be a problem instance with n agents and m items, each having a utility of 0 or 1 for every agent. For each agent i, let s_i be the number of items that have utility 1 for agent i. Now, checking $c_P \geq k$ is equivalent to checking whether integer solutions of value at least $k \cdot s_n/n$ exist to the following binary linear program:

$$\begin{array}{lll} \text{maximize} & \sum_{j=1}^{m} a_{nj} x_{nj} & \\ \text{subject to} & \sum_{j=1}^{m} a_{ij} x_{ij} \geq k \cdot s_i/n, & i \in [n-1], \\ & \sum_{i=1}^{n} x_{ij} \leq 1, & j \in [m], \\ & x_{ij} \in \{0,1\}, & \text{for } i \in [n], j \in [m], \end{array}$$

where variable $x_{ij} = 1$ if and only if item o_j is assigned to agent i, and $a_{ij} \in \{0,1\}$ is the value of item o_j to agent i. The first $n-1$ constraints ensure that every agent $i \in [n]$ gets a bundle of value at least her proportional share, while the next m constraints ensure that each item is assigned to at most one agent.

One can check that the constraint coefficients form a totally unimodular matrix, for which every square submatrix has determinant $0, 1$, or -1 (see, e.g., [27]). The relaxation of the binary linear program above, which is obtained by replacing the constraint that variables can take values 0 and 1 by the weaker constraint that they belong to the domain $[0,1]$, is convex and thus can be solved in polynomial time (see, e.g., [33]). On the other hand, since the constraint matrix is totally unimodular, all the fractional solutions to the relaxation problem are integers. ❑

Consequently, by considering $k = 1$ in the binary linear programming above, one can compute a proportionally fair allocation (if there exists one) in polynomial time. However, it remains open how the exact value of c_P can be determined in this case.

4.2 Scoring-Based Utilities

We consider a setting in which utility functions are defined via (strict) rankings over items along with a (nonnegative) decreasing vector $s = (s_1, \ldots, s_m) \in \mathbb{Q}^m_+$. By decreasing vector we mean that all the entries of s are in decreasing order from left to right, i.e., $s_1 > \cdots > s_m$, where m is the number of items. Given a ranking $\succ$ over items, a utility function associated with $\succ$ will assign a utility of s_j to an item ranked at position j, for all $j = 1, \ldots, m$. In this section we restrict our attention to Borda and lexicographic scoring vectors. A scoring vector s is *lexicographic* if $s_j > \sum_{k>j} s_k$ for all $j = 1, \ldots, m-1$, and is *Borda* if $s_j = m - j + 1$ for all $j = 1, \ldots, m$. An example of a lexicographic scoring vector is $s^* = (2^{m-1}, \ldots, 2, 1)$. We call utility functions induced by Borda (lexicographic) scoring vectors Borda (lexicographic) utility functions.

[1] By using a picking sequence, we can only guarantee that each agent i will receive a bundle whose value is at least $\lfloor s_i/n \rfloor$, where s_i is the number of items that have utility 1 for agent i. Interestingly, for the proportional share we need a guarantee of $\lceil s_i/n \rceil$. So the picking protocol is too weak by only a very small margin.

Fair allocation of indivisible goods with Borda and lexicographic utility functions has been considered by several authors (see, e.g., [14, 9, 15, 20, 19, 3, 23]). Bouveret and Lemaître [15] argued that there always exists a max-min fair allocation for any problem instance when the utility functions are induced by a scoring vector. However, this is not the case with the proportional share. In particular, we will show that if all agents' utility functions are identical and based on the lexicographic vector s^*, then there is no proportionally fair allocation. We start with the following easy-to-get observation.

OBSERVATION 1. *If all the utility functions are induced by a decreasing scoring vector $s \in \mathbb{Q}_+^m$, all agents have the same proportional share. Therefore, there is a proportionally fair allocation iff the value of a maximum egalitarian welfare allocation is at least the proportional share.*

It is NP-hard to determine whether a given allocation has maximum egalitarian welfare, even with utility functions based on the Borda (or lexicographic) scoring vector [9]. Several approximation algorithms have been developed for general additive utility functions (see, e.g., [7, 2], the survey [34], and the references therein).

PROPOSITION 1. *There is no proportionally fair allocation if all agents have the same utility functions induced by the lexicographic scoring vector s^*.*

PROOF. The proportional share of every agent is computed as $\mathrm{PS} = 1/n \sum_{i=0}^{m-1} 2^i = (2^m - 1)/n$. By the observation above, it suffices to prove that the worst-off agents in an optimal egalitarian welfare allocation have utility less than PS. Indeed, it is not hard to see that such an agent will receive the last $m - (n-1)$ items in her ranking (each of $n-1$ remaining agents gets only one item from her most-preferred $n-1$ items). Hence, her utility is equal to $\sum_{i=0}^{m-n} 2^i = 2^{m-n+1} - 1 = 2^m/2^{n-1} - 1$, which is less than $2^m/n - 1/n$, for all $m, n \geq 2$. ❑

In the following we characterize the existence of proportionally fair allocations for the case when $n = 2$. The proof is not hard and we omit it here due to the lack of space.

PROPOSITION 2. *Assume that all agents' utility functions are induced by the lexicographic scoring vector s^*. If $n = 2$, then there is a proportionally fair allocation if and only if the agents rank different items at the top.*

For more than two agents, it seems that achieving a succinct characterization on the existence of proportionally fair allocations seems to be unreasonable. A natural question is whether there are efficient algorithms for examining the existence of proportionally fair allocations for both Borda and lexicographic utility functions. For a fixed number of agents, we will answer this question positively by showing that there is a polynomial-time algorithm that can return, for very problem instance, the exact value of c_P. The starting point is Corollary 1 below, which is a consequence of Observation 1:

COROLLARY 1. *If we can compute an allocation of optimal egalitarian welfare in polynomial time, then one can determine exactly the value of c_P in polynomial time.*

For Borda utility functions, the problem of maximizing the egalitarian welfare can be done in polynomial time by using dynamic programming, if the number of agents is bounded, (see [10, 8]). Thus, following from Corollary 1, we have:

PROPOSITION 3. *For Borda utility functions, one can compute c_P in polynomial time when the number of agents is fixed.*

For lexicographic utility functions (based on the vector s^*), Baumeister et al. [10] gave very basic ideas for computing an allocation of optimal egalitarian welfare, for two agents. Their argument can be generalized to achieve a similar result for any lexicographic function and for any constant number of agents. In the following we will give the proof in detail.

LEMMA 2. *For lexicographic utility functions, the problem of maximizing egalitarian welfare with two agents is solvable in linear time in the number of items.*

PROOF. In the following, we denote by o_j^i the jth preferred object over $\mathcal{O}$ of agent i, for every $i \in \{1, 2\}$ and $j \in \{1, \ldots, m\}$. The ranking of the agents can then be described as follows:

$$\text{preference of agent } 1 : \succ_1 \; : \; o_1^1 \; o_2^1 \; o_3^1 \; \ldots \; o_m^1,$$
$$\text{preference of agent } 2 : \succ_2 \; : \; o_1^2 \; o_2^2 \; o_3^2 \; \ldots \; o_m^2.$$

The utilities of items are defined based on a decreasing lexicographic vector $(s_1, s_2, \ldots, s_m)$: $u_1(o_j^1) = u_2(o_j^2) = s_j$ for all $j = 1, \ldots, m$. We write $o_j^1 = o_k^2$ when o_j^1 and o_k^2 are identical, and $o_j^1 \neq o_k^2$ otherwise. Our algorithm involves a sequence of rules when the agents' preferences are examined from the most preferred to the least preferred items, and it relies on the fact that every agent i will be happier with an item of rank j than with a bundle of all objects ranked below position j, i.e., $u_i(o_j^i) > \sum_{k>j} u_i(o_k^i)$, due to the lexicographic scoring vector. Our algorithm starts with the first position $j = 1$, and applies the following rules:

(A) If the agents rank different items at the current position j and none of the objects is assigned yet, both agents get the item from the current position. Proceed with the next position $j = j + 1$.

(B) If both objects are already assigned, proceed with the next position $j = j + 1$.

Let j be the first position where neither rule A nor rule B can be applied. This means that at this position, either (I) two agents rank the same object that has not been assigned yet; or (II) only one of the two items ranked at this position has been already allocated in previous steps. We denote by (π_1, π_2) the incomplete allocation obtained by applying the rules A and B, and let S be the set of remaining items. Note that $u_1(\pi_1) = u_2(\pi_2)$. One can prove that (π_1, π_2) is a partial allocation of any optimal (complete) allocation.

LEMMA 3. *If (π_1^*, π_2^*) is an optimal (complete) allocation, then it must hold that $\pi_1 \subseteq \pi_1^*$ and $\pi_2 \subseteq \pi_2^*$.*

LEMMA 4. *If (π_1, π_2) is an optimal allocation of items that belong to the first k objects in the ranking of agent 1 or agent 2 and $\|\pi_1\| = \|\pi_2\|$, then (π_1, π_2) must be a partial allocation of an optimal (complete) allocation.*

The proof of Lemma 3 can be found in the appendix; the proof of Lemma 4 is quite similar and thus omitted due to lack of space.

Lemma 3 indicates that it is sufficient to find an optimal assignment $\pi' = (\pi_1', \pi_2')$ of items in S to agents, and then an optimal allocation (π_1^*, π_2^*) will be computed by setting $\pi_1^* = \pi_1 \cup \pi_1'$ and $\pi_2^* = \pi_2 \cup \pi_2'$. Now, imagine that we are at position j where there are two possible scenarios: (I) and (II) as noted above.

Case (I): In this case, both agents rank the same object o that has not been assigned yet on the current position. It is not hard to see that the optimal allocation π' is either $(\{o\}, S \setminus \{o\})$ or $(S \setminus \{o\}, \{o\})$. If they have different egalitarian welfare, pick the one of largest welfare.

Object	Description
j	First position where rules cannot be applied
S	Set of remaining items
o_k^1	Most preferred item in S by agent 1
$\hat{S}$	Set of available items between j and k for agent 2
Type 1	Agent 2 gets one item from $\hat{S} \setminus \{o_k^2\}$, agent 1 gets the remaining items
Type 2	Agent 2 does not get any item from $\hat{S} \setminus \{o_k^2\}$
$o_{k'}^1$	Least preferred item in $\hat{S} \setminus \{o_k^2\}$ by agent 1
S_1	Set of available items between k and k' for agent 1
S_2	Set of available items without S_1 between k and k' for agent 2

Table 1: Notation in Lemma 2

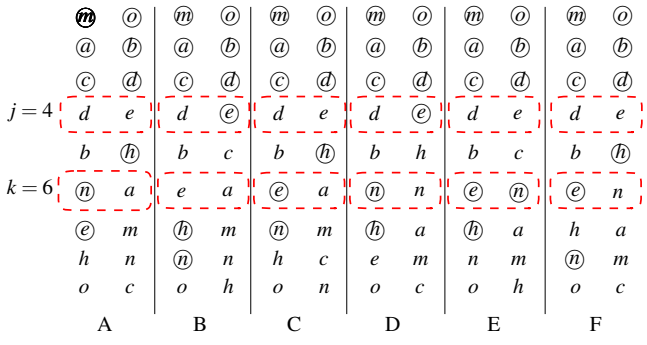

Figure 1: The objects are ordered from the top (most preferred) to the bottom (least preferred) according to their rank. The objects that are circled show an optimal allocation.

Case (II): Only one among two objects at this position has already been assigned in the previous steps, and one can assume, w.l.o.g., that it is o_j^1. Let o_k^1 be the most preferred item of agent 1 among the ones in S. Recall that S is the set of available items at this position j, including o_j^2. Also, we denote by $\hat{S} \subseteq S$ the set containing all available objects ranked by agent 2 from position j to position k. Making the decision on how to proceed at this position j will depend on the availability of object o_k^2. This task turns out to be more complicated than that of Case (I). In fact, it should be further divided into subcases such that an optimal allocation can be found easily in each case.

Case (II-1): $o_k^2 \notin S$. If $o_k^1 \neq o_j^2$, the egalitarian welfare of π' is at least s_k. This can only be guaranteed by giving o_k^1 to 1, and some item o in $\hat{S}$ to 2. Note that 2 cannot receive more than one item from $\hat{S}$, as one can obtain a better allocation by reallocating one of these items to 1. Hence, o must be the only item 2 can get in π'. Finally, by checking totally $\|\hat{S}\|$ possible allocations from which we can find an allocation of maximum egalitarian welfare (see Figure 1-A).

Consider the case when $o_k^1 = o_j^2 = o$. If o_j^2 is the only available item in $\hat{S}$, then anyone who is not assigned o will get the remaining items of total value less than s_k. Therefore, π' is one of the following allocations: $(\{o\}, \{S \setminus \{o\}\})$ or $(\{S \setminus \{o\}\}, \{o\})$, whichever has better egalitarian welfare (see Figure 1-B). If there are more than two available items in $\hat{S}$, then π' can be found as in the previous case with $o_k^1 \neq o_j^2$ (see Figure 1-C).

Case (II-2): $o_k^2 \in S$. If $o_k^1 = o_k^2 = o$, an optimal allocation will assign one item from $\hat{S} \setminus \{o\}$ to 2 and the remaining items to 1. The way to find it is exactly the same as the one in case $o_k^2 \notin S$ and $o_k^1 \neq o_j^2$ (see Figure 1-D). Now, for the remainder of this proof, consider the other case: $o_k^1 \neq o_k^2$.

If $o_k^1 = o_j^2$ and there is no available object ranked (by agent 2) between positions k and j, then it is easily seen that $(\pi_1 \cup \{o_k^1\}, \pi_2 \cup \{o_k^2\})$ will be an optimal allocation of items that belong to the first k objects in the ranking of agent 1 or agent 2. By Lemma 4, to find the optimal allocation (π_1^*, π_2^*), we just allocate o_k^1 and o_k^2 to 1 and 2, respectively, move to the next position $j = k+1$, set $S \leftarrow S \setminus \{o_k^1, o_k^2\}$, and repeat all the steps as we have done from the beginning, starting with applying the rules A and B (see Figure 1-E). If there is at least one available object o_t^2 for some $t, j < t < k$, the optimal allocation π^* is that 2 gets π_2 and o_t^2, where $t \in (j,k)$ such that o_t^2 is an available item of lowest rank according to agent 1's ranking, and agent 1 gets π_1 and the rest (see Figure 1-F).

If $o_k^1 \neq o_j^2$, then three objects o_k^1, o_j^2, o_k^2 are pairwise different. The optimal allocation π' could be one of the following types: (type 1) agent 2 gets one item from $\hat{S} \setminus \{o_k^2\}$ and 1 gets the remaining ones; (type 2) agent 2 does not get any item from $\hat{S} \setminus \{o_k^2\}$. To find π', we first compute the best allocations of type 1 and of type 2, and then pick the one of maximum welfare. As argued above, the best allocation of type 1 is $\mu = (S \setminus \{o\}, \{o\})$, where o is the object of lowest rank (according to $\succ_1$) among the ones in $\hat{S} \setminus \{o_k^2\}$. In addition, the egalitarian welfare of this allocation is $u_1(S \setminus \{o\})$ (because o is at a better position than k for agent 2 and agent 1 gets items that are at position k or worse). We now determine the best allocation of type 2. Let k' be the position such that $o = o_{k'}^1$. Let S_1 be the set of all available objects ranked by agent 1 from position k to position k', and $S_2 \subseteq S \setminus S_1$ be the set of the remaining available objects ranked by agent 2 from position k to position k'. Note that $o_{k'}^1$ has lowest rank among the ones in S_1. We consider two cases.

If $o_k^2 \in S_1$: It follows $o_k^2 \succ_1 o$. The best allocation of type 2 will assign o_k^1 and o_k^2 to 1 and 2, respectively. Hence, the best possible bundle agent 1 can get is $S \setminus \{o_k^2\}$, which has value less than the value of $S \setminus \{o\}$, since $o_k^2 \succ_1 o$. Therefore, the optimal allocation π' in this case is of type 1, i.e., $\pi' = \mu$ (see Figure 2-A).

If $o_k^2 \notin S_1$: We consider the following three subcases.

- $u_1(S_1 \setminus \{o\}) > u_2(S_2)$: Because $S_1 \setminus \{o\} \subseteq S \setminus \{o\}$ and $o \succ_2 o'$ for every $o' \in S_2$, the egalitarian welfare of every allocation of type 2 is always less than that of the best allocation of type 1, thus $\pi' = \mu$ (see Figure 2-B).

- $u_1(S_1 \setminus \{o\}) = u_2(S_2)$: It is clear that $u_1(S_1) > u_2(S_2)$. Note that $\hat{S} \setminus o_k^2 \subseteq S_1$ by definition of position k'. Therefore a best allocation of type 2 is $(S_1, S \setminus S_1)$. Because S_1 contains the same items as S up to item o, $u_1(S_1) > u_1(S \setminus \{o\})$. Thus we need only compare $u_2(S \setminus S_1)$ and $u_1(S \setminus \{o\})$ to see if the allocation $(S_1, S \setminus S_1)$ has better welfare than μ. The optimal allocation π' is $(S_1, S \setminus S_1)$ if $u_2(S \setminus S_1) > u_1(S \setminus \{o\})$, and is μ otherwise (see Figure 2-C-D).

- $u_1(S_1 \setminus \{o\}) < u_2(S_2)$: It follows that $u_1(S_1) = u_2(S_2)$ or $u_1(S_1) < u_2(S_2)$. One can see that the optimal allocation π' will be of type 2 in both cases. If $u_1(S_1) = u_2(S_2)$ then $(\pi_1 \cup S_1, \pi_2 \cup S_2)$ is a partial allocation of π^*. Hence, we simply assign S_1 and S_2 to 1 and 2, respectively, and then optimize the allocation of the remaining items in $S \setminus \{S_1 \cup S_2\}$ to agents. To this end, we move to the next position $j = k'+1$, set $S \leftarrow S \setminus \{S_1 \cup S_2\}$, and repeat all the steps as we have done so far from the beginning (see Figure 2-E). In the case when $u_1(S_1) < u_2(S_2)$, π' will be of the form $(S \setminus S', S')$ for some $S' \subseteq S_2$. Hence, we just need to compute a subset S' that maximizes the utility of agent 1 to the bundle $(S \setminus S')$, while keeping $u_2(S')$ not less than $u_1(S_1)$. This can be done as follows. We first rank the items in S_2 in order of decreasing utilities

according to agent 1's ranking. We then try to pass through each item, from the most to the least preferred ones, to agent 2, under the condition that the utility of 2 for the set of remaining items is not less than $u_1(S_1)$. When we have finished with the last item, we will obtain exactly an optimal subset S' which, in fact, includes all items that have been passed (see Figure 2-F).

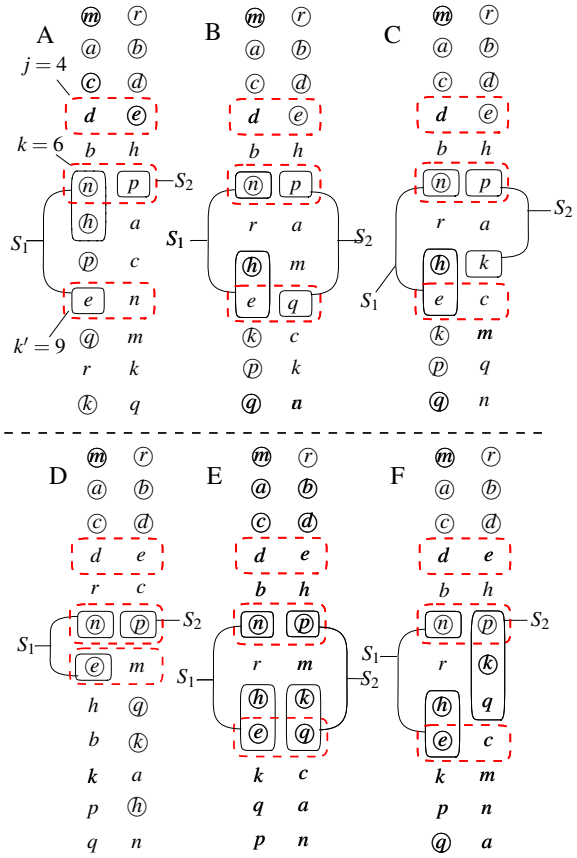

Figure 2: The objects are ordered from the top (most preferred) to the bottom (least preferred) according to their rank. The objects that are circled show an optimal allocation.

Since there are totally m positions of items that need to be considered, the running time of our algorithm will be bounded by $O(m)$, and this completes the proof. ❑

One can extend the result above to the case with any constant number of agents.

THEOREM 4. *For lexicographic utility functions, the problem of maximizing egalitarian welfare with a fixed number of agents is solvable in polynomial time in the number of items.*

Proof Sketch. The statement is proven by induction on the (constant) number n of agents. Indeed, the induction base $n = 2$ has been already shown. Suppose that the statement holds for the case of $n-1$ agents. We need to prove that every problem instance with n agents can be solved in polynomial time as well. Let I be a problem instance with n agents. We develop an algorithm that enumerates in polynomial time all possible optimal allocations.

Algorithm 3.

Step 1. $T \leftarrow \mathcal{A}$; $j \leftarrow 1$; $R_j \leftarrow \mathcal{O}$; $\pi_i \leftarrow \varnothing$ for all i.

Step 2. If $\|R_j\| < n$ then assign items to agents in an arbitrary way, update the allocation π and go to Step 4. Otherwise, compute a set W_j of all possible allocations such that in each such allocation every agent in T gets only one item from R_j, and the utility of the worst-off agent is maximized. Note that $\|W_j\|$ is bounded by $O(m^n)$. Furthermore, there is at most one allocation in W_j, denoted by ω^*, such that every agent has the same utility.

Step 3. For each allocation $\omega \in W_j$, let $T(\omega) \subseteq T$ be the set containing all the worst-off agents (w.r.t. this ω), and $R_j(\omega)$ be the subset of items which is obtained from R_j by deleting all items that have been allocated in ω. If $\omega \neq \omega^*$, then $T \setminus T(\omega) \neq \varnothing$, and thus $\|T(\omega)\| < n$. Hence, one can solve the new instance $(T(\omega), R_j(\omega))$ in polynomial time (by the induction hypothesis). It is enough to consider the worst-off agents because of lexicographic utility. The obtained optimal allocation is then combined with ω to yield a *possible* allocation $A(\omega)$ for I. We set $V_j = \bigcup_{\omega \in W_j \setminus \{\omega^*\}} \{A(\omega)\}$. For allocation ω^*, we first combine the allocation $\omega^* = (\omega_1^*, \ldots, \omega_n^*)$ with $\pi = (\pi_1, \ldots, \pi_n)$, i.e., set $\pi_i \leftarrow \pi_i \cup \omega_i^*$, for all $i \in T$; set $j \leftarrow j+1$; and set R_j to be the subset of items obtained from R_{j-1} by deleting all items those have been allocated in ω^* and go back to Step 2.

Step 4. Return an allocation in $\{\pi\} \cup \bigcup_j V_j$ of maximum egalitarian welfare.

Our algorithm runs in at most $O(m)$ steps. In each step, we need to solve at most $O(m^n)$ sub-instances (each with less than n agents), each requires $O(m^{O(n)})$ time. Overall, the running time of the algorithm is bounded by $O(m^{O(n)})$ and thus is polynomial in m since n is constant. ❑

From Corollary 1 and Theorem 4 we have the following result:

THEOREM 5. *For lexicographic utility function, one can compute c_p, and thus a c_P-proportional fair allocation, in polynomial time, if the number of agents is fixed.*

For the case when the number of agents is part of the input, it was shown by Baumeister et al. [8] that there is an $1/2$-approximation algorithm for the problem of maximizing egalitarian welfare with lexicographic utility functions. This algorithm immediately results in a $1/2$-proportional fair allocation.

5. 2-ADDITIVE PREFERENCES

In this section we consider the max-min fair allocation problem with 2-additive utility functions. We restrict our attention to the case with two agents only. Bouveret and Lemaître [15] made the first steps towards studying this problem. Among other results, they proved that there is an instance for which there is no max-min fair allocation, even with only two agents. Therefore, it is of great interest to know whether there always exists a c-max-min fair allocation for every problem instance, for some constant $c < 1$. We will answer this question negatively.

PROPOSITION 4. *For any constant $c > 0$, there is an instance with two agents such that there is no c-max-min fair allocation.*

PROOF. One can assume, without loss of generality, that $c \in (0, 1)$. We consider an instance involving two agents and three goods $\{o_1, o_2, o_3\}$; the agents' 2-additive utility functions are defined as follows:

$$u_1(\pi_1) = ko_1 + 1o_2 + 1o_3 - 1o_1o_2 + (k-2)o_2o_3 - ko_1o_3,$$
$$u_2(\pi_2) = 1o_1 + ko_2 + 1o_3 - 1o_1o_2 - ko_2o_3 + (k-2)o_1o_3,$$

where π_1 and π_2 denote the agents' respective shares, $k > 2$ is a positive number, and o_1, o_2, and o_3 are binary variables that indicate whether o_i is in the bundle.

One can see that the max-min share of agent 1 is equal to k (an allocation giving this share is $\pi = (\{o_1\}, \{o_2, o_3\})$). Similarly, the max-min share of agent 2 is also k. However, in any feasible allocation it holds that there is at least one agent getting a bundle of value exactly 1. This means there is no allocation in which the utility of every agent is better than $1/k$ of her max-min share. By choosing $k = 2/c$, the proof is completed. ❑

A natural direction is to explore special classes of utility functions for which c can be bounded by a constant. We study the case where utility functions are symmetric and submodular, and show that c can be bounded by a constant when there are only two agents. A utility function u is *submodular* if for every subsets $S, T \subseteq \mathcal{O}$, $u(S) + u(T) \geq u(S \cup T) + u(S \cap T)$, and u is *symmetric* if $u(S) = u(\mathcal{O} \setminus S)$ for any $S \subseteq \mathcal{O}$.

PROPOSITION 5. *If $n = 2$ and the utility functions are symmetric and submodular then $c = 1/2$ and this is tight in the sense that a $(1/2 + \varepsilon)$-max-min fair allocation for any $\varepsilon > 0$ is impossible. Furthermore, a $1/4$-max-min fair allocation can be found in polynomial time.*

Due to space constraints, the proof of Proposition 5 is omitted here but will be contained in the full version of the paper.

6. CONCLUSIONS AND FUTURE WORK

We have addressed the question of computing the best value c for which there is an allocation such that all agents have utility of at least c times either their max-min share or their proportional share. For a fixed number of agents having additive utilities, while Aziz et al. [6] provided a PTAS for approximating c for the max-min share, we have proposed an alternative way for doing so. Arguably, the most interesting aspect of this contribution does not lie in the specific result but in our technique, which we feel is, conceptually, more broadly applicable than the technique that Aziz et al. [6] used to prove their PTAS. In fact, our technique can potentially be applied more easily to other variants of fair allocation problems, such as min-max fair allocation and rank-weighted utilitarianism. We defer a detailed discussion about the application range to the full version of this paper.

On the other hand, we have discovered special cases in which c_P (the above-mentioned value c for the proportional share) can be efficiently solved or approximated. For binary utility functions, one can check in polynomial time if c_P is at least as large as a given number k, but we do not know yet if c_P can be computed in the same amount of time. When the utility functions are induced by Borda or lexicographic scoring vectors, we have obtained exact algorithms for determining c_P.

For 2-additive utility functions, the problem of computing c_M (the above-mentioned value c for the max-min share) as well as finding lower bounds for it becomes harder. We have shown that c_M cannot be bounded by a universal constant, not even for two agents. We have explored a class of symmetric submodular functions which allows us to bound c_M by a (tight) constant of $1/2$. Furthermore, we have developed an approximation algorithm which gives to every agent a share of at least $1/4$ of her max-min share.

There are several interesting directions for future work. First, the most important task is to study the approximability of the problem of computing the value of c with an unbounded number of agents. Second, exploring the special cases of 2-additive utility functions in which c can be bounded easily could provide new insights into the problem. Finally, it would be interesting to know whether there exist max-min fair or proportionally fair allocations that are also Pareto-efficient. A similar question has been studied for envy-free and Pareto-efficient allocations [13, 21, 11].

APPENDIX

Proof of Fact 1. We prove that x^G has at most $2(n-1)$ fractional components. Indeed, by adding slack variables, the constraints (2) and (3) in LP(G) can be rewritten as

$$\sum_{j=1}^{m} u_{ij} x_{ij} + r_i = LB_i(\hat{z}_i), \qquad i \in [n-1],$$
$$\sum_{i=1}^{n} x_{ij} + s_i = 1, \qquad j \in [n'].$$

Note that some variables x_{ij} were set to 1 (or 0) according to setting (5). Hence, the number of constraints (3) here is $n' < m$. Denote by k the number of variables x_{ij}. The total number of variables of LP(G) is $k + n + n' - 1$. Since we have $n + n' - 1$ nontrivial constraints, the basic optimal solution x^G of LP(G) has at most $n + n' - 1$ positive components (not including the ones set by (5)). On the other hand, by the inequalities (3), there is at least one positive component x_{ij}^G for every j. We divide n' inequalities (3) into two types, I and II as follows. For each inequality j of Type-I, $x_{ij}^G = 1$ for some i; and for each inequality j of Type-II, there is at least one $x_{ij}^G \in (0, 1)$. Let r be the number of Type-I inequalities, then the number of inequalities of Type-II is $n' - r$. Since the number of non-integral components x_{ij}^G in each Type-II inequality is at least 2, the total number of positive components of x^G is at least $r + 2(n' - r) = 2n' - r$. We have an upper bound of $n' + n - 1$ on the number of positive components of x^G. Therefore, we must have that $2n' - r \leq n' + n - 1$ or $r \geq n' - n + 1$. So the number of non-integral components of x^G is at most $(n' + n - 1) - (n' - n + 1) = 2(n - 1)$. ❑ Fact 1

Proof of Fact 2. For $i \in [n-1]$, we have

$$\sum_{j \in [m]} u_{ij} x_{ij}^G \geq LB_i(\hat{z}_i) \geq \frac{1}{1 + \varepsilon/2} u_i(\pi_i),$$

where the first inequality is due to the feasibility of G to LP(G) and the last inequality follows from (8).

For $i = n$, we have $\sum_{j \in [m]} u_{nj} x_{nj}^G \geq u_n(\pi_n)$, since x^G is an optimal solution of LP(G). ❑ Fact 2

Proof of Lemma 3. We first assume that π_1 and π_2 are nonempty, otherwise, the claim is trivial. For every $k \in [j-1]$, we denote $S_k^1 = \{o_1^1, \ldots, o_k^1\}$ and $S_k^2 = \{o_1^2, \ldots, o_k^2\}$. Intuitively, S_k^1, S_k^2 are the sets of the first k objects in the ranking of agent 1 and agent 2, respectively. We can assume that $u_1(\pi_1^*) \geq u_2(\pi_2^*)$ (the case when $u_1(\pi_1^*) \leq u_2(\pi_2^*)$ can be treated similarly). Because the utility functions are lexicographic, it follows that $u_1(\pi_1^* \cap S_k^1) \geq u_2(\pi_2^* \cap S_k^2) \geq u_1(\pi_1 \cap S_k^1) = u_2(\pi_2 \cap S_k^2)$ for any $k \in [j-1]$.

We now give the proof by contradiction. Suppose, w.l.o.g., that $\pi_1 \backslash \pi_1^* \neq \varnothing$ and let $o_k^1, (1 < k \leq j-1)$, be the highest-ranked item in π_1^* but not in π_1 (such an item o_k^1 exists as $\pi_1^* \not\subseteq \pi_1$). It must hold that $u_2(\pi_2 \cap S_{k-1}^2) = u_1(\pi_1 \cap S_{k-1}^1) = u_1(\pi_1^* \cap S_{k-1}^1) \geq u_2(\pi_2^* \cap S_{k-1}^2)$. If $u_2(\pi_2 \cap S_{k-1}^2) = u_2(\pi_2^* \cap S_{k-1}^2)$, this means π_2 and π_2^* are the same if restricted on S_{k-1}^2. Note that π_1 and π_1^* are also the same if restricted on S_{k-1}^1. Now since $o_k^1 \in \pi_1^*$, this means this item is available at the position k, and thus must be also allocated in π^1 by the rule A. This is a contradiction. Therefore, $u_2(\pi_2 \cap S_{k-1}^2) > u_2(\pi_2^* \cap S_{k-1}^2)$. By the lexicographic property, it follows that $u_2(\pi_2) > u_2(\pi_2^*)$ and this, again, is a contradiction as π^* is the optimum. Hence $\pi_1 \subseteq \pi_1^*$. It is not hard to see that $\pi_2 \subseteq \pi_2^*$. ❑ Lemma 3

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their helpful comments. We also thank Khaled Elbassioni for very helpful discussions. This work was supported in part by DFG grant RO 1202/14-2.

REFERENCES

[1] G. Amanatidis, E. Markakis, A. Nikzad, and A. Saberi. Approximation algorithms for computing maximin share allocations. In *Proceedings of the 42nd International Colloquium on Automata, Languages, and Programming*, pages 39–51. Springer LNCS 9134, July 2015.

[2] A. Asadpour and A. Saberi. An approximation algorithm for max-min fair allocation of indivisible goods. In *Proceedings of the 39th ACM Symposium on Theory of Computing*, pages 114–121. ACM Press, July 2007.

[3] H. Aziz, P. Biró, J. Lang, J. Lesca, and J. Monnot. Optimal reallocation under additive and ordinal preferences. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, pages 402–410, May 2016.

[4] H. Aziz, S. Gaspers, S. Mackenzie, and T. Walsh. Fair assignment of indivisible objects under ordinal preferences. *Artificial Intelligence*, 227:71–92, 2015.

[5] H. Aziz, G. Rauchecker, G. Schryen, and T. Walsh. Approximation algorithms for max-min share allocations of indivisible chores and goods. Technical Report arXiv: 1604.01435 [cs.GT], ACM Computing Research Repository (CoRR), April 2016.

[6] H. Aziz, G. Rauchecker, G. Schryen, and T. Walsh. Algorithms for max-min share fair allocation of indivisible chores. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*. AAAI Press, Feb. 2017. To appear.

[7] N. Bansal and M. Sviridenko. The Santa Claus problem. In *Proceedings of the 38th ACM Symposium on Theory of Computing*, pages 31–40. ACM Press, July 2006.

[8] D. Baumeister, S. Bouveret, J. Lang, N. Nguyen, T. Nguyen, J. Rothe, and A. Saffidine. Positional scoring-based allocation of indivisible goods. *Autonomous Agents and Multi-Agent Systems*. To appear. DOI: 10.1007/s10458-016-9340-x.

[9] D. Baumeister, S. Bouveret, J. Lang, T. Nguyen, N. Nguyen, and J. Rothe. Scoring rules for the allocation of indivisible goods. In *Proceedings of the 21st European Conference on Artificial Intelligence*, pages 75–80. IOS Press, Aug. 2014.

[10] D. Baumeister, S. Bouveret, J. Lang, T. Nguyen, J. Rothe, and A. Saffidine. Positional scoring rules for the allocation of indivisible goods. In *Proceedings of the 11th European Workshop on Multi-Agent Systems*, December 2013.

[11] B. Bliem, R. Bredereck, and R. Niedermeier. Complexity of efficient and envy-free resource allocation: Few agents, resources, or utility levels. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, pages 102–108. AAAI Press/IJCAI, July 2016.

[12] S. Bouveret, Y. Chevaleyre, and N. Maudet. Fair allocation of indivisible goods. In F. Brandt, V. Conitzer, U. Endriss, J. Lang, and A. Procaccia, editors, *Handbook of Computational Social Choice*, chapter 12, pages 284–310. Cambridge University Press, 2016.

[13] S. Bouveret and J. Lang. Efficiency and envy-freeness in fair division of indivisible goods: Logical representation and complexity. *Journal of Artificial Intelligence Research*, 32:525–564, 2008.

[14] S. Bouveret and J. Lang. A general elicitation-free protocol for allocating indivisible goods. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pages 73–78. AAAI Press/IJCAI, July 2011.

[15] S. Bouveret and M. Lemaître. Characterizing conflicts in fair division of indivisible goods using a scale of criteria. *Journal of Autonomous Agents and Multi-Agent Systems*, 30(2):259–290, 2016.

[16] E. Budish. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Journal of Political Economy*, 119(6):1061–1103, 2011.

[17] Y. Chevaleyre, U. Endriss, S. Estivie, and N. Maudet. Multiagent resource allocation in k -additive domains: preference representation and complexity. *Annals Operations Research*, 163(1):49–62, 2008.

[18] R. Cole, V. Gkatzelis, and G. Goel. Mechanism design for fair division: allocating divisible items without payments. In *Proceedings of the 14th ACM Conference on Electronic Commerce*, pages 251–268. IFAAMAS, June 2013.

[19] A. Darmann and C. Klamler. Proportional Borda allocations. *Social Choice and Welfare*, 47(3):543–558, 2016.

[20] A. Darmann and J. Schauer. Maximizing Nash product social welfare in allocating indivisible goods. *European Journal of Operational Research*, 247(2):548–559, 2015.

[21] B. de Keijzer, S. Bouveret, T. Klos, and Y. Zhang. On the complexity of efficiency and envy-freeness in fair division of indivisible goods with additive preferences. In *Proceedings of the 1st International Conference on Algorithmic Decision Theory*, pages 98–110. Springer-Verlag *Lecture Notes in Artificial Intelligence #5783*, October 2009.

[22] T. Erlebach, H. Kellerer, and U. Pferschy. Approximating multiobjective knapsack problems. *Management Science*, 48(12):1603–1612, 2002.

[23] E. Fujita, J. Lesca, A. Sonoda, T. Todo, and M. Yokoo. A complexity approach for core-selecting exchange with multiple indivisible goods under lexicographic preferences. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pages 907–913. AAAI Press, January 2015.

[24] L. Gourvès, J. Monnot, and L. Tlilane. A matroid approach to the worst case allocation of indivisible goods. In *Proceedings of the 23th International Joint Conference on Artificial Intelligence*, pages 136–142, August 2013.

[25] L. Gourvès, J. Monnot, and L. Tlilane. Worst case compromises in matroids with applications to the allocation of indivisible goods. *Theoretical Computer Science*, 589:121–140, 2015.

[26] T. Heinen, N. Nguyen, and J. Rothe. Fairness and rank-weighted utilitarianism in resource allocation. In *Proceedings of the 4th International Conference on Algorithmic Decision Theory*, pages 521–536. Springer-Verlag *Lecture Notes in Artificial Intelligence #9346*, Sept. 2015.

[27] I. Heller and C. Tompkins. An extension of a theorem of Dantzig's. In H. Kuhn and A. Tucker, editors, *Linear Inequalities and Related Systems*, pages 247–254. Princeton University Press, 1956.

[28] T. Hill. Partitioning general probability measures. *The Annals of Probability*, 15:804–813, 1987.

[29] D. Hochbaum and D. Shmoys. Using dual approximation algorithms for scheduling problems: Theoretical and practical results. *Journal of the ACM*, 34(1):144–162, 1987.

[30] D. Kurokawa, A. Procaccia, and J. Wang. When can the maximin share guarantee be guaranteed? In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 523–529. AAAI Press, Feb. 2016.

[31] J. Lang and J. Rothe. Fair division of indivisible goods. In J. Rothe, editor, *Economics and Computation. An Introduction to Algorithmic Game Theory, Computational Social Choice, and Fair Division*, chapter 8, pages 493–550. Springer-Verlag, 2015.

[32] E. Markakis and C.-A. Psomas. On worst-case allocations in the presence of indivisible goods. In *Proceedings of the 7th International Workshop on Internet & Network Economics*, pages 278–289, Dezember 2011.

[33] A. Nemirovski and M. Todd. Interior-point methods for optimization. *Acta Numerica*, 17:191–234, May 2008.

[34] T. Nguyen, M. Roos, and J. Rothe. A survey of approximability and inapproximability results for social welfare optimization in multiagent resource allocation. *Annals of Mathematics and Artificial Intelligence*, 68(1–3):65–90, 2013.

[35] A. Procaccia and J. Wang. Fair enough: Guaranteeing approximate maximin shares. In *Proceedings of the 15th ACM Conference on Economics and Computation*, pages 675–692. ACM Press, 2014.

[36] H. Steinhaus. The problem of fair division. *Econometrica*, 16(1):101–104, 1948.

[37] G. Woeginger. A polynomial-time approximation scheme for maximizing the minimum machine completion time. *Operations Research Letters*, 20(4):149–154, 1997.