

Nobody Left Behind: Fair Allocation of Indivisible Goods

Ivona Bezáková and Varsha Dani

Department of Computer Science, University of Chicago, Chicago, IL 60637, USA
{ivona,varsha}@cs.uchicago.edu

Abstract. The Max-Min Fairness problem is as follows: Given m indivisible goods and k players, each with a specified valuation function on the subsets of the goods, how should the goods be split between the players so as to maximize the minimum valuation. Viewing the problem from a game theoretic perspective, we show that for two players and additive valuations the expected minimum of the (randomized) cut-and-choose mechanism is a $1/2$ -approximation of the optimum. To complement this result we show that no truthful mechanism can compute the exact optimum. We also consider the algorithmic perspective when the (true) additive valuation functions are part of the input. We present a simple $1/(m - k + 1)$ approximation algorithm which allocates to every player at least $1/k$ fraction of the value of all but the $k - 1$ heaviest items. We also give an algorithm with additive error against the fractional optimum bounded by the value of the largest item. The two approximation algorithms are incomparable in the sense that there exist instances when one outperforms the other. We conclude with a $1/2 + \varepsilon$ factor NP-hardness of approximation result.

1 Introduction

The need for fair division of a set of objects among several parties emerges naturally in many real-life scenarios, ranging from inheritance and divorce settlements to border disputes. Naturally, the political science and economic literature addresses the issues of fair division under various assumptions and objectives (see, e.g., Brams and Taylor [2]). But few of the methods view the problem from a computational perspective and most assume the existence of several divisible objects or the possibility of a monetary compensation in order to achieve a balanced division. In certain situations such as some inheritance disputes the items are often indivisible, of different value to each of the parties, and the parties might not be able to compensate the other players financially.

General Max-Min Fairness Problem. There are k players and a set A of m indivisible objects. Player i has a non-negative valuation function $v_i : 2^A \rightarrow \mathbf{R}_0^+$ which is normalized to $v_i(A) = 1$. The goal is to find a partition A_1, \dots, A_k of the goods which maximizes $\min_i v_i(A_i)$.

The inspiration for the Max-Min Fairness problem comes from its divisible counterpart: the cake-cutting problem. Initiated by the Polish school of mathematicians in the 1950's, in the cake-cutting problem k parties attempt to divide a cake between them in a fair manner. Generally, one of two objectives comes under consideration: envy-freeness or fair-fractions. A division is *envy-free* if every player thinks she got at least as much as any other player. A division is divided into *fair fractions*, if each person gets at least $1/k$ of the whole cake (from her perspective). The cake-cutting represents a problem from the class of divisible fair allocation problems and several solutions for both objective functions are well-known (see, e.g., Robertson and Webb [9]). Perhaps the most interesting aspect of the cake-cutting solutions is their truthfulness; in other words, players are given guarantees if they do not lie about their valuation functions.

Lipton, Markakis, Mossel and Saberi [6] studied the indivisible variant of the envy-free cake-cutting problem. Under the assumption that players reveal their true valuation functions, Lipton *et al.* presented a simple polynomial-time algorithm producing an allocation with an absolute bound on the envy. Their envy is guaranteed to be at most the maximal marginal utility defined as the largest value by which a value of a set increases after enlarging it by exactly one element. Their algorithm works for (essentially) unrestricted classes of valuation functions, with the caveat that the true valuation functions are known to the algorithm. It is not known whether there is a reasonable mechanism when the players can lie about their valuations in order to obtain a larger portion.

The Max-Min Fairness problem is an indivisible analogue of the fair-fractions cake-cutting problem, and it has been proposed by Lipton *et al.* as an open direction. We note that this problem and the envy-minimization problem are of different nature as there exist instances when the minimum value obtained by the envy-minimizing splits is arbitrarily worse than the minimum in the max-min allocations. Similarly, the envy of the max-min allocations might be much bigger than the envy in the envy-minimizing splits.

We consider two different input settings. In the “*known valuations*” setting the algorithm is given an oracle access to the v_i ’s. In the opposite “*unknown valuations*” setting, the algorithm asks the players directly for the v_i values. This setting is conceptually more difficult, since the players might opt to lie in the hope of achieving a better split for themselves. We assume that the players do not know anything about the other players’ valuations.

A general valuation function assigns a real number to every subset of A , and one may need to specify 2^m numbers to describe it. Therefore we will work with two restricted but natural classes of valuations. We call a valuation *additive* if the value of a set equals the sum of the values of the individual items. A valuation is *maximal* if the value of a set of items equals the value of the most expensive item in the set. This situation corresponds to a set of items of similar functionality such as finding a full-time job. Notice that the above valuations are induced by m real numbers, the values of single items from A .

In the case of known additive valuations, the Max-Min Fairness problem is a special case of a scheduling problem where the completion time of any single machine is maximized. Woeginger [11] and Epstein and Sgall [3] presented polynomial-time approximation schemes for uniformly related machines. These results immediately translate to instances of the Max-Min Fairness problem where all the players have equal valuations. Woeginger [12] gave a pseudo-polynomial algorithm for constant number of machines (players).

1.1 Our Results

Mechanism design. In the case of unknown valuations, the mechanism needs to elicit relevant information from the players who might try to play deceitfully. Our main result is a randomized allocation procedure for two players, which achieves at least half the optimum for both players in expectation. This result might not sound surprising. A trivial algorithm gives all the items to a single player, determining the recipient by an unbiased coin flip. Every player expects $1/2$ value of the goods, so the minimum of the expected values is $1/2$. However, the expected minimum is 0, since the other player does not get anything. We want to avoid this situation.

We say that a randomized algorithm for the Max-Min Fairness problem c -approximates the optimum, if the algorithm guarantees that the expected minimum (as opposed to the minimum of the expected values) is within a c factor of the deterministic optimum. Our algorithm, a randomized cut-and-choose mechanism, is an expected $1/2$ -approximation, provided that the players play rationally. However, full rationality requires the cutter to solve an NP -complete problem: finding the optimal cut. Fortunately this is not necessary. We show that it suffices to be rational in a limited sense: the cutter needs to find a locally optimal cut for which we provide a polynomial-time algorithm. This is the first truthful mechanism with provable guarantees for a problem of fair allocation of indivisible goods.

Omniscient algorithms. We present a simple polynomial matching-based algorithm for known maximal valuations. Building on this algorithm we obtain a $1/(m - k + 1)$ approximation of the optimal allocation for known additive utilities. Iteration of a variant of this algorithm guarantees a $1/k$ fraction of all but the $k - 1$ heaviest items to every player. More precisely, every player gets a bundle that is at least as good as getting every k -th item, where the items are sorted decreasingly by their individual values.

We also present an algorithm based on an integer rounding of the corresponding linear programming relaxation, which allocates to every player the fractional optimum value minus the heaviest item. The rounding technique is inspired by the work of Lenstra, Shmoys and Tardos [5]. They considered the makespan-minimization problem when scheduling a set of jobs on unrelated machines. Their rounding method is based on the insight that the graph underlying the fractional solution is a bipartite pseudoforest, which may then be rounded to a matching. We use the same framework for the first part of our rounding scheme. However, since our problem is of an opposite nature (a maximization rather than a minimization), we need to round the pseudoforest edges to a structure which has a matching-like behavior for one partition of the underlying bipartite graph while in the other partition it behaves like an “anti-matching”. Our contribution consists of finding such a structure, described in Lemma 4.

We note that the two algorithms we exhibit for the additive valuations are complementary, in the sense that each outperforms the other on some inputs.

Our algorithms are not only polynomial time but are efficiently implementable.

Finally, using an idea similar to that of Lenstra *et al.* [5], we prove that no ρ approximation algorithm exists for $\rho > 1/2$ unless $P = NP$. A linear factor approximation algorithm and a $1/2 + \varepsilon$ hardness result were obtained independently by Markakis and Saberi [7].

1.2 Related Economic Work

In the economic and political sciences literature (see, e.g., Brams and Taylor [2]) the goal differs somewhat from ours: the quality of an allocation is measured in terms of equitability (did the players get the same utility?), envy-freeness, and efficiency (no other allocation gives a larger utility to all players simultaneously). Probably the most well-known method is the so-called adjusted winner procedure. It uses a monetary rebalancing in order to satisfy the equitability, envy-freeness and efficiency conditions simultaneously.

Moulin [8] considered a related problem where m *identical* indivisible objects need to be allocated to k parties, each of which specifies a request for a certain number of objects. He gave a proportional allocation method and proved that it is the only fair method (in the sense of “Equal Treatment of Equals”, i.e., two agents with identical claims should receive the same random allocation) satisfying other desirable structural invariance properties.

2 Notation

The valuation functions can be expressed conveniently as a complete weighted bipartite graph which we refer to as the *valuations graph*. One partition represents the set of players and the other to the set of goods where the weight of the edge (i, j) is $v_i(j)$.

We denote the optimum value by OPT . We say that a randomized algorithm is an *expected c -approximation* of the optimum if $E(\min_i v_i(A_i)) \geq cOPT$.

3 Known maximal valuations

In this section we present a polynomial-time algorithm to find a fairness maximizing allocation for known valuations of maximal type. This problem is equivalent to the following matching problem applied to the valuations graph of the Max-Min Fairness instance where A is the set of goods and B is the set of players.

Max-Min Matching Problem.

Input: A bipartite graph $G = (A \cup B, E)$ and edge-weights $w : E \rightarrow \mathbf{R}_{\geq 0}$. Goal: A matching $M \subseteq E$ such that

1. M covers B , i.e. for every $b \in B$ there exists a s.t. $(a, b) \in M$,
2. minimum weight edge in M is as large as possible, i.e., for every M' covering B , $\min_{e \in M} w(e) \geq \min_{e \in M'} w(e)$.

Lemma 1. *There is a polynomial-time algorithm for Max-Min Matching.*

Proof. The algorithm works as follows: Choose a threshold t and create an unweighted graph G_t by omitting edges in G of weight $< t$. Check whether G_t contains a perfect matching. Then G has a max-min matching $\geq t$ if and only if G_t contains a perfect matching. We can now find the optimal t by binary search. \square

4 Known additive valuations

In the case of known additive valuations, finding a fairness maximizing allocation is *NP*-complete since the SUBSET SUM problem (see, e.g., Garey and Johnson [4]) is a special case for 2 players and identical valuations, i.e., $v_1 \equiv v_2$. We present an approximation guarantee of $1/(m - k + 1)$ where m is the number of items and k the number of players. Moreover every player gets at least the value of every k -th item if items are sorted decreasingly by this player’s valuations. Inspired by the work of Lenstra *et al.* [5] we analyze an LP based algorithm with an additive guarantee against the *fractional* optimum. Our algorithm gives a utility of at least the fractional optimum minus the player’s largest item to every player. These two algorithms are incomparable, in the sense that neither performs consistently better than the other (as discussed in Appendix 7.2). We note that it is NP-hard to approximate the optimal solution to the Max-Min Fairness problem to within factor ρ for any $\rho > \frac{1}{2}$. This can be seen via a reduction from 3-MATCHING; the details are in Appendix 7.3.

4.1 An approach via Matching

Lemma 2. *Max-Min Matching gives a $1/(m - k + 1)$ approximation to the Max-Min Fairness problem with additive valuations.*

Proof. Let OPT be the value of the optimal solution for a Max-Min Fairness instance with additive valuations and let A_i be the set of items allocated to player i in the optimal solution. Without loss of generality, we can assume that every person is allocated at least one object. Otherwise, the optimum is 0 which happens if and only if the graph obtained from the valuations graph by deleting edges of 0 weight does not have a matching of size k . This can be checked in polynomial time.

Let $x_i \in A_i$ be the largest object in the i -th player's valuation. By the pigeon-hole principle $v_i(x_i) \geq v_i(A_i)/|A_i|$ and $|A_i| \leq (m - k + 1)$. We create a matching M' by matching up the i -th player with x_i . Suppose M is an optimum matching in the Max-Min Matching problem. Then $\min_{(i,j) \in M} v_i(j) \geq \min_{(i,j) \in M'} v_i(j) \geq OPT/(m - k + 1)$. \square

The instance of two players with valuations $v_1 = (\frac{m-1-\epsilon}{m-\epsilon}, \frac{1}{m-\epsilon}, 0, 0, \dots, 0)$ and $v_2 = (\frac{1}{m-\epsilon}, \frac{1}{m}, \frac{1}{m}, \dots, \frac{1}{m}, \frac{m-2\epsilon}{m(m-\epsilon)})$ for m items shows that the approximation ratio $1/(m - k + 1)$ is tight for this algorithm, and cannot be improved even by assigning the remaining items. However we do get an absolute guarantee when the input does not contain a large item.

Theorem 1. *Let $x_{i,1}, \dots, x_{i,m}$ be a decreasingly sorted sequence of goods according to the i -th player's valuation, i. e., $v_i(x_{i,j}) \geq v_i(x_{i,j+1})$ for every j . There exists a polynomial-time algorithm which produces a partition of the goods such that each player i is guaranteed a value of at least $\sum_{\ell} v_i(x_{i,k\ell}) \geq (1 - \sum_{j=1}^{k-1} v_i(x_{i,j}))/k$. Moreover, the algorithm has a $1/(m - k + 1)$ approximation guarantee, where m is the number of goods and k is the number of players.*

Before we prove the theorem, it will be advantageous to state a generalization of the Max-Min Matching problem.

Generalized Max-Min Matching Problem. Input: A bipartite graph $G = (A \cup B, E)$, edge-weights $w : E \rightarrow \mathbf{R}_{\geq 0}$, and vertex values $v(b) \in \mathbf{R}^+$ for every $b \in B$. Goal: A matching $M \subseteq E$ such that

1. M covers B , i.e. for every $b \in B$ there exists a s.t. $(a, b) \in M$,
2. minimum weight edge in M , together with the value of the adjacent vertex, is as large as possible, i. e., for every M' covering A , $\min_{(a,b) \in M} w(a, b) + v(b) \geq \min_{(a,b) \in M'} w(a, b) + v(b)$.

Lemma 3. *There exists a polynomial-time algorithm for the Generalized Max-Min Matching problem.*

Proof. We will reduce the Generalized Max-Min Matching to the Max-Min Matching problem. Given an input instance to the Generalized Max-Min Matching problem G , w and v , we create an instance G' , w' of the Max-Min Matching by letting $w'(a, b) = w(a, b) + v(b)$. Then the optimum solutions for both instances, respectively, are identical. \square

Proof of Theorem 1. We run the Generalized Max-Min Matching algorithm iteratively. We start with the valuations graph of the Max-Min Fairness instance with player vertices valued initially at 0. After finding the max-min matching M , we assign the goods in M to the corresponding people. We update the vertex values to $v_{new}(i) = v_{old}(i) + w(i, j)$ where $(i, j) \in M$ and we erase the used items: $G_{new} := G_{old} \setminus Goods(M)$ where $Goods(M)$ is the set of goods covered by M . We find the Generalized Max-Min of the new instance and continue until the number of goods is less than people.

Since we use the original Max-Min Matching algorithm in the first iteration, the $1/(m - k + 1)$ approximation factor follows from Lemma 2. Notice that the first Generalized Max-Min Matching always matches i with an item $x_{i,j}$ where $j \leq k$. In general, the ℓ -th Generalized Max-Min Matching matches i with $x_{i,j}$ where $j \leq k\ell$. The worst such allocation for i would be getting $x_{i,k\ell}$ for $\ell = 1, \dots, \lfloor m/k \rfloor$. But $x_{i,k\ell}$ is the largest item among $x_{i,j}$ for $j = k\ell, \dots, k(\ell + 1) - 1$. Therefore i gets a value of at least $(1 - \sum_{j=1}^{k-1} v_i(x_{i,j}))/k$. \square

The guarantee to player i is tight for the algorithm, as can be seen from the following example of k players having identical valuations, with $k - 1$ items of value $1/k$ each and $nk + 1$ items of value $1/(nk^2 + k)$ each, so that their total value is $1/k$. Then the above algorithm (assuming every iteration matches maximal matching), gives value $1/k + n/(nk^2 + k)$ to $k - 1$ players, while the remaining player gets $(n + 1)/(nk^2 + k)$. This converges (for large n) to $1/k^2$.

4.2 Linear Programming Relaxation

Another approach to solving the Max-Min Fairness problem is to formulate it as an integer program. Let $x_{i,j}$ denote the indicator variable for “player i gets item j ”. Then $x_{i,j} \in \{0, 1\}$. The constraints on the program are that each item must be allocated exactly once and that each player’s utility must be at least as much as the objective function. Thus the optimal allocation for the Max-Min Fairness problem instance is the solution to the integer program:

$$\text{Maximize } \omega \text{ subject to: } x_{i,j} \in \{0, 1\}, \forall j : \sum_i x_{i,j} = 1 \text{ and } \forall i : \omega \leq \sum_j v_i(j) x_{i,j}$$

We consider the linear programming relaxation that allows fractional allocations, or in other words, arbitrarily divisible goods.

$$\text{Maximize } \omega \text{ subject to: } 0 \leq x_{i,j} \leq 1, \forall j : \sum_i x_{i,j} = 1 \text{ and } \forall i : \omega \leq \sum_j v_i(j) x_{i,j}$$

Note that the optimal value of ω for the linear program, henceforth denoted $FOPT$, is at least $1/k$ where k is the number of players. Moreover the optimal (fractional) allocation satisfies the property that all players in a connected component of the valuations graph receive the same total value.

Given a feasible solution $\mathbf{x} = (x_{i,j})_{1 \leq i \leq k, 1 \leq j \leq m}$ to the above linear program, let $\mathbf{x}_i = (x_{i,j})_{1 \leq j \leq m}$ denote the items allocated to player i . By abuse of notation, we’ll use $v_i(\mathbf{x}_i)$ to denote $\sum_j v_i(j) x_{i,j}$, the utility to player i from this allocation.

Theorem 2. *Let $\hat{\mathbf{x}}$ be an optimal (fractional) solution of the above program, with $FOPT_i := v_i(\hat{\mathbf{x}}_i)$. Then there exists an integer solution \mathbf{y} such that $v_i(\mathbf{y}_i) \geq \max(0, FOPT_i - \max_j v_i(j))$. It follows that $\min_i v_i(\mathbf{y}_i) \geq \max(0, FOPT - \max_{i,j} v_i(j))$. Moreover, \mathbf{y} can be found in polynomial time.*

We credit part of the proof of the above theorem to Lenstra, Shmoys and Tardos [5] who used a similar theorem in a different scheduling context. The Max-Min Fairness is a special case of a scheduling problem on unrelated machines where the minimum machine completion time is maximized. The opposite problem of minimizing the maximum machine completion time (also called makespan) has been extensively studied. Lenstra *et al.* [5] provided a 2-approximation algorithm for unrelated machines, as well as a $4/3 - \varepsilon$ factor non-approximability result if $P \neq NP$. Their algorithm is based on an integer rounding of the corresponding fractional solution. Our contribution is summarized in the following lemma.

Lemma 4. *Let G be a bipartite pseudotree, i. e., a connected graph with at most as many edges as vertices. Let A and B be the vertex sets of G and let E be the edge set. Then there is a set $S \subset E$ such that for all $j \in A$, S covers j by exactly one edge, and for every $i \in B$, S covers i by at least $\deg(i) - 1$ edges. This set can be found in polynomial time.*

Proof. First of all, note that a graph G is a pseudotree if and only if it has the property that it is either a tree or a tree plus one edge. If G' is the graph obtained by deleting from G a vertex of degree 1 together with the (unique) edge incident with it, then G' is also a pseudotree. The following procedure constructs the required set $S \subset E$.

- Case 1: G contains a degree 1 vertex $v \in A$. In this case let e be the edge incident with v . Let G' be the graph obtained by removing v and e from G . Let $S' \subset E$ be the set of edges obtained by recursively applying this procedure to G' . (Note that if G' is a two vertex graph with a single edge, then we can set S' to contain the single edge.) Set $S = S' \cup \{e\}$.
- Case 2: G contains no vertices of degree 1 in A , but does contain a degree 1 vertex $v \in B$. Note that such a vertex v can afford to have no edge incident with it in S . As before, let G' be the graph obtained by removing v and its incident edge from G , and S' be the recursive solution for G' . Then set $S = S'$.
- Case 3: G contains no vertices of degree 1. In this case G could not have been a tree, so it must be a tree plus one edge. Moreover, it must in fact be a cycle. (A tree always contains at least two vertices of degree 1; the only way for a tree plus one edge to have no degree 1 vertices is if the tree had exactly two degree 1 vertices and the extra edge connected these two. In this case it is a cycle). Finally, since G is bipartite, it must be an even cycle. Thus we can set S to be either set of alternating edges in G .

This completes the proof. □

Sketch of the proof of Theorem 2 (complete proof can be found in the Appendix 7.1). We define a bipartite graph G with partitions I , the set of users, and J , the set of goods. An edge i, j is included if $\hat{x}_{i,j} > 0$. The technique of Lenstra *et al.* implies that G is a pseudoforest (each component is either a tree or a tree with an additional edge).

Using the Lemma 4, we can round \hat{x} into (integer-coordinate) y as follows: For every component C of G let S_C be the set of edges defined by applying Lemma 4 to C . Let $y_{i,j} = 1$ if and only if it corresponds to an edge from an S_C . Clearly, rounding $y_{i,j}$ to 1 does not decrease the value for player i and since we round at most one edge incident to every player down to 0, the player loses at most $\max_j v_i(j)$. \square

5 Unknown Additive Valuations

We now consider the case where the algorithm does not know the players' true valuations as part of its input but must instead elicit information about them from the players. Thus the task of finding an optimal allocation becomes a mechanism design problem. In this setting, one cannot assume that the players will not try to cheat to their own advantage. Ideally, one would like a mechanism which forces all the players to be truthful (by arranging the payoffs so that revealing their true valuation is the players' best strategy). Unfortunately, as the next example shows, no mechanism (deterministic or randomized) for the Max-Min fair allocation problem can be truthful.

Example 1. Suppose \mathcal{M} is a mechanism to compute a Max-Min fair allocation. Consider two players with the following valuations for three items.

| | g_1 | g_2 | g_3 |
|-------|-------|-------|-------|
| Alice | 2/3 | 1/3 | 0 |
| Bob | 0 | 1/2 | 1/2 |

\mathcal{M} will allocate g_1 to Alice and the rest to Bob, giving Alice a utility of $\frac{2}{3}$. However by declaring her valuation as $(\frac{1}{3}, \frac{2}{3}, 0)$ Alice can trick the mechanism into giving Bob only g_3 , and increasing her utility to 1.

In light of this, we would like to examine what can be accomplished by mechanisms which elicit only partial information about the valuations from the players. In the spirit of the classical cake cutting problem of Banach, Knaster, and Steinhaus [10] we would like to have a mechanism that gives each player a guarantee based on the information received from the player, such that deceitful play always results in a worse guarantee to the player, even if not a worse payoff in every instance. Also, we would like deceitful play by one player to affect the guarantees of other players as little as possible. We would like the guarantee provided by the mechanism to be as nice as possible.

As a first attempt, we consider mechanisms that get no information from the players. This does not give the players any opportunity to cheat. Consider the mechanism that simply assigns all the goods to a single player chosen uniformly at random. Then each player's expected utility is $1/k$ and hence the minimum expected utility is $1/k$. However the mechanism always gives nothing to all but one player. Thus the actual minimum utility (and hence also the expected minimum utility) is always 0, which is rather unsatisfactory.

We can try to fix this problem by assigning the goods at random, but having a separate lottery for each item. Thus the mechanism throws m unbiased k -sided dice and thereby comes up with an allocation. Again, by linearity of expectation, each player's expected utility and the minimum expected utility are $1/k$. As the following example shows however, this mechanism also does not have a very good guarantee on the expected minimum utility.

Example 2. Consider k players and k items. Suppose for each i , $1 \leq i \leq k$, player i 's valuation is given by $v_i(i) = 1$, $v_i(j) = 0$ for all $j \neq i$. The optimal allocation is to assign each item to the unique player desiring it, so that everyone has utility 1. The randomized mechanism described above achieves this allocation $1/k^k$ of the time. The rest of the time it gives at least one item to a "wrong" player, so that the minimum utility is 0. Thus, the expected minimum utility is $1/k^k$.

Thus, unless the players provide some information about their valuations, no reasonable guarantee can be made for the the expected minimum utility.

In Section 5.1 we present a randomized mechanism for two players which with some assumptions on the players' strategies, gives an expected $\frac{1}{2}$ -approximation to the optimal allocation, while simultaneously guaranteeing each player at least $1/2$ of what they would have got in the optimal allocation. How to extend this to more than two players is as yet unclear.

5.1 Two Players: Cut-and-Choose

In this section we investigate the cut-and-choose mechanism for two players. In the basic cut-and-choose mechanism player 1 divides the goods into two parts, and player 2 chooses one of the parts. Player 1 then receives the other part. Even in the case of divisible goods, player 2 may have an advantage in this game. We'll consider the more symmetric version in which the first player ("cutter") is chosen by a fair coin flip. (We do not assume that the players have the same valuations.)

We take the standpoint that each player wishes to optimize her worst case outcome against adversarial play by the opponent. Thus the cutter's goal is to divide the goods as equally as possible. However, the cutter's optimization problem is NP-hard by reduction from SUBSET SUM. With this in mind we introduce the following notion of local optimality.

Definition 1. We will call a partition (S, T) of the goods *locally optimal* for valuation v if moving a single item from either side to the other does not decrease the disparity of the partition, $|v(S) - v(T)|$. Moreover we require that whenever the disparity is positive, all items of zero value are on the smaller side.

We now observe that a locally optimal partition can be computed in nearly-linear time.

Proposition 1. *Given any valuation v , a locally optimal partition can be computed in time $O(m \log m)$.*

Proof. Sort the goods in decreasing order of value. Iteratively assign the most valuable remaining good to the less valuable of S and T , breaking ties arbitrarily. The sorting requires $O(m \log m)$ time, and the assignments take only $O(m)$ time. All that remains is to check that the resulting partition is locally optimal.

First note that if moving the smallest good on the bigger side does not shrink the gap, then neither will moving any other good. This smallest good, j , was the last good to be added to the bigger side. Without loss of generality, assume $v(S) > v(T)$. Let $S_0 = S \setminus \{j\}$, and let $T_0 \subseteq T$ be those goods in T which were allocated before j . Then $v(S_0) \leq v(T_0)$, or the algorithm would have assigned j to T . Thus

$$v(S) - v(T) = v(S_0) + v(j) - v(T) \leq v(S_0) + v(j) - v(T_0) \leq v(j),$$

which implies moving j from S to T would not decrease the gap. □

Our main result in this section is that, assuming both players produce locally optimal partitions when cutting and select the larger piece when choosing the randomized cut-and-choose mechanism produces an allocation whose expected minimum utility is at least half the maximum minimum utility. Moreover, it guarantees each player at least half what they would have received in an optimal solution. Note also that these guarantees are preserved if the players are truly rational, i. e., unhindered by computational constraints, since the true optimal partition *does* satisfy the local optimality conditions.

Theorem 3. *Let players P_1 and P_2 have additive valuation functions v_1 and v_2 . Let A_1, A_2 be the optimal allocation for these valuations, i. e., the allocation maximizing $\min_i v_i(A_i)$ and let $OPT = \min_i v_i(A_i)$. Then, assuming both players produce locally optimal partitions, the randomized cut and choose mechanism produces an allocation S_1, S_2 such that*

$$\mathbf{E} \left(\min_i v_i(S_i) \right) \geq \frac{1}{2} OPT \quad \text{and for each } i \quad \mathbf{E} (v_i(S_i)) \geq \frac{1}{2} v_i(A_i)$$

Proof. Suppose player P_1 cuts. Let (S, T) be the cut made by P_1 and let $v_1(S) = \frac{1}{2} - \delta$ and $v_1(T) = \frac{1}{2} + \delta$. Player P_2 may now choose either of the sets, and P_1 gets the other. Note that in this protocol P_2 is guaranteed a utility of at least $\frac{1}{2}$, which is at least $\frac{1}{2} v_2(A_2)$.

- Case 1: P_2 chooses S . This means that $v_2(S) \geq \frac{1}{2}$. Thus the allocation $S_1 = T, S_2 = S$ satisfies $\min_i v_i(S_i) \geq \frac{1}{2}$. Since $OPT \leq 1$ we have $\min_i v_i(S_i) \geq \frac{1}{2} OPT$ in this case.
- Case 2: P_2 chooses T and $|T| = 1$. This means that the single item j in T is valued by both players at at least $\frac{1}{2}$. In this case, the optimal allocation is to give j to the player valuing it more and the rest of the items to the other player. The allocation $S_1 = S, S_2 = T$ achieved by the mechanism is the same as this optimal allocation when $v_2(j) \geq v_1(j)$, and otherwise it is flipped, in which case $\min_i v_i(S_i) \geq 0$.

- Case 3: P_2 chooses T and $|T| = m' \geq 2$. This means that $v_2(T) \geq \frac{1}{2}$. Let $j \in T$ be the item that P_1 values least, so that $v_1(j) \leq (\frac{1}{2} + \delta)/m'$. Since P_1 cannot improve the split by moving j from T to S , it follows that $v_1(j) - \delta \geq \delta$ or in other words, $v_1(j) \geq 2\delta$. These two inequalities, together with the fact that $m' \geq 2$ imply that $\delta \leq \frac{1}{6}$.
We have $S_1 = S$, $S_2 = T$ and $\min_i v_i(S_i) = \frac{1}{2} - \delta$. In order to show this is at least $\frac{1}{2}OPT$ it suffices to show that $OPT \leq 1 - 2\delta$. We will show this by contradiction. Suppose B_1, B_2 is an allocation in which $\min_i v_i(B_i) > 1 - 2\delta$. Then $v_1(B_1) > 1 - 2\delta$. Since P_1 values every item in T at least 2δ , it follows that $T \subset B_1$. But this means that $B_2 \subset S$, so that $v_2(B_2) \leq v_2(S) = 1 - v_2(T) \leq \frac{1}{2} < 1 - 2\delta$ since $\delta \leq \frac{1}{6}$. This gives the desired contradiction. In fact, the above argument shows that P_1 cannot get more than $1 - 2\delta$ in the optimal allocation, so that we also have $v_i(S_i) \geq \frac{1}{2}v_i(A_i)$

By symmetry, all of the above holds if we switch players P_1 and P_2 . In cases 1 and 3 we achieve $\frac{1}{2}OPT$ and each player gets at least half what they would have in the optimal allocation, *regardless of who cuts*. In case 2 we achieve the optimal allocation with probability $\frac{1}{2}$ and give each player something non-negative otherwise. It follows that using an unbiased coin flip to determine who cuts gives an allocation with the desired property. \square

Note that the above analysis is tight: suppose there are four items that the players value at $(\frac{1}{2}, \frac{1}{2}, 0, 0)$ and $(0, 0, \frac{1}{2}, \frac{1}{2})$ respectively. Assuming that the players do not know each other's valuations and that the cutter makes a locally optimal cut, no matter who cuts, in the resulting allocation both players get utility $\frac{1}{2}$. On the other hand, the optimal allocation gives both players utility 1.

6 Conclusions and Open Problems

We presented a linear factor approximation algorithm for the Max-Min Fairness problem with additive valuations, and a corresponding hardness result of factor $1/2$. We also analyzed a (deterministic) rounding technique which obtains an integer solution with an additive guarantee against the fractional optimum. Can randomized rounding be used to give a better-than-linear approximation guarantee? Can we get a better non-approximability factor using Probabilistically Checkable Proofs [1]?

We showed that the randomized cut-and-choose mechanism gives an expected $1/2$ -approximation guarantee against the deterministic optimum for two players. Can we obtain an improved approximation factor for two players? How can one generalize the two player mechanism for three or more players?

References

1. S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, 1998.
2. S. J. Brams and A. D. Taylor. *Fair Division : From Cake-Cutting to Dispute Resolution*. Cambridge University Press, 1996.
3. L. Epstein and J. Sgall. Approximation schemes for scheduling on uniformly related and identical parallel machines. *Algorithmica*, 39(1):43–57, 2004.
4. M. Garey and D. Johnson. *Computers and Intractability*. Freeman and Co., New York, 1979.
5. J. K. Lenstra, D. B. Shmoys, and É. Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Math. Program.*, 46:259–271, 1990.
6. R. J. Lipton, E. Markakis, E. Mossel, and A. Saberi. On approximately fair allocations of indivisible goods. In *ACM Conference on Electronic Commerce*, pages 125–131, 2004.
7. E. Markakis and A. Saberi, 2004. Personal communication.
8. H. Moulin. The proportional random allocation of indivisible units. *Social Choice and Welfare*, 19(2):381–413, 2002.
9. J. Robertson and W. Webb. *Cake Cutting Algorithms: Be Fair If You Can*. A. K. Peters, Ltd., 1998.
10. H. Steinhaus. The problem of fair division. *Econometrica*, 16(1):101–104, January 1948.
11. G. J. Woeginger. A polynomial time approximation scheme for maximizing the minimum machine completion time. *Operations Research Letters*, 20:149–154, 1997.
12. G. J. Woeginger. When does a dynamic programming formulation guarantee the existence of a fully polynomial time approximation scheme (FPTAS)? *INFORMS Journal on Computing*, 12:57–75, 2000.

7 Appendix

7.1 Complete proof of Theorem 2

Proof of Theorem 2. Since the inequalities $x_{i,j} \leq 1$ are redundant, we have $km + k + m$ inequalities defining the polytope of feasible solutions of the linear program. There are km variables $x_{i,j}$ and one ω . Suppose $\hat{\mathbf{x}}$ is an optimal solution located in the “corner” of the polytope. Therefore at least $km + 1$ of the inequalities defining the polytope are satisfied as equalities, implying that at most $k + m - 1$ of the variables are non-zero.

We define a bipartite graph G with partitions I , the set of users, and J , the set of goods. An edge i, j is included if $\hat{x}_{i,j} > 0$. We will show that G is a pseudoforest (each component is either a tree or a tree with an additional edge).

Let C be a connected component of G , let I_C, J_C be the corresponding partitions. Let $\hat{\mathbf{x}}^{(C)}$ be the part of $\hat{\mathbf{x}}$ restricted to C . Let us fix $\hat{\omega}$ and $\hat{\mathbf{x}}$ except for the variables from C and let L_C be the linear program restricted to variables defined by C . We will prove that $\hat{\mathbf{x}}^{(C)}$ is an extreme point of L_C . Suppose not, then there exist feasible points $\mathbf{y}_1, \mathbf{y}_2$ in L_C such that $(\mathbf{y}_1 + \mathbf{y}_2)/2 = \hat{\mathbf{x}}^{(C)}$. Let \mathbf{z}_i be a vector identical to $\hat{\mathbf{x}}$ but with $\hat{\mathbf{x}}^{(C)}$ replaced with \mathbf{y}_i . Then \mathbf{z}_i is a feasible point of the original linear program and $\hat{\mathbf{x}} = (\mathbf{z}_1 + \mathbf{z}_2)/2$, a contradiction with its extremity.

Since $\hat{\mathbf{x}}^{(C)}$ lies in the “corner” of L_C , by the above argument at most $|I_C| + |J_C|$ of the corresponding variables are non-zero, i.e. there are at most $|I_C| + |J_C|$ edges in C . Therefore C is either a tree or a tree with one additional edge.

Using this structure and Lemma 4, we can round $\hat{\mathbf{x}}$ into (integer-coordinate) y as follows: For every component C of G let S_C be the set of edges defined by applying Lemma 4 to C . Let $y_{i,j} = 1$ if and only if it corresponds to an edge from an S_C . Clearly, rounding $y_{i,j}$ to 1 does not decrease the value for player i and since we round at most one edge incident to every player down to 0, the player loses at most $\max_j v_i(j)$. \square

7.2 Why Two Algorithms?

In Section 4 we presented two approaches for approximating the optimal solution to the Max-Min Fairness problem. The guarantees of these two algorithms are of different types. Whereas the first approach, based on matching, gives a multiplicative guarantee, the linear programming approach gives an additive guarantee against the fractional optimum, which is at least as good as (and often significantly better than) the integer optimum. These algorithms are not comparable, in the sense that neither performs consistently better than the other in all situations as the following examples demonstrate.

Example 3. Consider four items and two players with the following valuations:

| | g_1 | g_2 | g_3 | g_4 |
|-------|-------|-------|-------|-------|
| Alice | 2/3 | 1/3 | 0 | 0 |
| Bob | 1/3 | 1/4 | 1/4 | 1/6 |

The matching algorithm allocates g_2 and g_4 to Alice and g_1 and g_3 to Bob, resulting in a minimum value of $\frac{1}{3}$. The corresponding linear program actually has an integer optimal solution in which Alice gets g_1 and Bob gets the rest and they both get a value of $\frac{2}{3}$.

Example 4. Consider five items and three players with the following valuations:

| | g_1 | g_2 | g_3 | g_4 | g_5 |
|-------|----------|--------------|-------|--------------|----------|
| Alice | δ | $1 - \delta$ | 0 | 0 | 0 |
| Bob | 0 | 1/3 | 1/3 | 1/3 | 0 |
| Carol | 0 | 0 | 0 | $1 - \delta$ | δ |

where $\delta < \frac{1}{3}$. For this example, the matching algorithm finds the optimal solution, which is to allocate g_1 and g_2 to Alice, g_3 to Bob, and g_4 and g_5 to Carol, with a minimum value of $\frac{1}{3}$. The linear program produces the fractional allocation depicted in figure 1 the fractional edges depend on the exact value of δ . Our rounding procedure then assigns g_3 to Bob, g_1 to Alice, g_5 to Carol, g_2 to Bob(!) and g_4 to Carol, resulting in a minimum value of δ . Since δ is arbitrary, this allocation could be very poor.

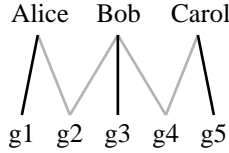


Figure 1: The LP solution. Grey edges denote fractional allocations.

7.3 Non-approximability

In this section we analyze a $1/2 + \varepsilon$ factor hardness result under the assumption $P \neq NP$. Our argument uses a refinement of a reduction due to Lenstra *et al.* [5].

Proposition 2. *There is no polynomial-time ρ -approximation algorithm for Max-Min Fairness for $\rho > 1/2$ unless $P = NP$.*

Proof. We present a reduction from 3-matchings. In the 3-matching problem, the input consists of disjoint sets X, Y, Z , each of size n , and a set $E \subseteq X \times Y \times Z$. The goal is to decide whether there exists $M \subseteq E$ of size n such that $\cup_{(x,y,z) \in M} \{x, y, z\} = X \cup Y \cup Z$. This problem is known to be NP-complete.

For a 3-matching instance we create the Max-Min Fairness instance as follows. Every player corresponds to $e \in E$, thus $k = |E|$. The items are $X \cup Y$ and a set of “dummy” items. For every $z \in Z$ there will be $m_z := |\{(x, y, z) \in E \mid x \in X, y \in Y\}| - 1$ copies of dummies corresponding to c , denoted $d_{z,j}$. (Without loss of generality, we can assume that $m_z > 0$. Otherwise either there is no $e \in E$ incident to z , which is a clear NO for the 3-matching, or there is a single $e \in E$ incident to z and it must be included in M .) Therefore $m = 2n + (|E| - n)$. Player $e = (x, y, z)$ has the following valuations: $v_e(x) = v_e(y) = 1$, $v_e(d_{z,j}) = 2k/m_z$ for every $j \leq m_z$, and $v_e(x') = v_e(y') = v_e(d_{z',j}) = 0$ for every $x' \neq x, y' \neq y, z' \neq z$ and $j \leq m_{z'}$.

Let OPT denote the optimum of the Max-Min Fairness instance. We claim that there exists a 3-matching if and only if $OPT \geq 2$. To see this, suppose there is a 3-matching M . Then we can assign items x, y to every $(x, y, z) \in M$. For $(x, y, z) \in E \setminus M$ we assign one of the dummies, $d_{z,j}$. Since all of the dummies are valued at ≥ 2 by the corresponding players, every player is given a value ≥ 2 .

On the other hand, suppose $OPT \geq 2$. There are at least n people without a dummy in the optimum solution. Every (x, y, z) without a dummy must get both x and y . This is possible if and only if the set of no-dummy people is a 3-matching of the original instance.

Notice that if $OPT < 2$, it is indeed ≤ 1 . Therefore having a ρ -approximation for the Max-Min Fairness for $\rho > 1/2$ would imply a polynomial-time solution for the 3-matching problem. \square

8 Acknowledgments

We are indebted to Lance Fortnow for guiding and encouraging us throughout this work. Many thanks go to Tom Hayes and Daniel Štefankovič for helpful discussions. We are also grateful to Bruno Codenotti, Pedro Felzenszwalb, Prahladh Harsha, Adam Kalai, John Langford, Elchanan Mossel, Martin Pál, Amin Saberi and Gerhard Woeginger for useful comments and pointers to the literature.