On Worst-Case Allocations in the Presence of Indivisible Goods

Evangelos Markakis[*] Christos-Alexandros Psomas[*]

Abstract

We study a fair division problem, where a set of indivisible goods is to be allocated to a set of n agents. Each agent may have different preferences, represented by a valuation function that is a probability distribution on the set of goods. In the continuous case, where goods are infinitely divisible, it is well known that proportional allocations always exist, i.e., allocations where every agent receives a bundle of goods worth to him at least $1/n$. In the presence of indivisible goods however, this is not the case and one would like to find worst case guarantees on the value that every agent can have. We focus on algorithmic and mechanism design aspects of this problem.

In the work of Hill [7], an explicit lower bound was identified, as a function of the number of agents and the maximum value of any agent for a single good, such that for any instance, there exists an allocation that provides at least this guarantee to every agent. The proof however did not imply an efficient algorithm for finding such allocations. Following upon the work of [7], we first provide a slight strengthening of the guarantee we can make for every agent, as well as a polynomial time algorithm for computing such allocations. We then move to the design of truthful mechanisms. For deterministic mechanisms, we obtain a negative result showing that a truthful $2/3$-approximation of these guarantees is impossible. We complement this by exhibiting a simple truthful algorithm that can achieve a constant approximation when the number of goods is bounded. Regarding randomized mechanisms, we also provide a negative result, showing that we cannot have truthful in expectation mechanisms under the restrictions that they are Pareto-efficient and satisfy certain symmetry requirements.

[*]Athens University of Economics and Business, Dept. of Informatics. Emails: {markakis, alexpsomi}@gmail.com. Research supported by the Basic Research Funding Program (BRFP) of the Athens University of Economics and Business.

1 Introduction

Fair division problems have attracted the attention of various scientific disciplines, including among others, mathematics, economics, and political science. Ever since the first attempt for a formal treatment by Steinhaus, Banach, and Knaster [13], many interesting and challenging questions have emerged. Over the last decades, a vast literature has developed, see e.g., [2, 12], and several notions of fairness have been suggested. In the recent years, this area has also gained popularity in computer science, see among others, [6, 3], as most of the questions that have been posed are algorithmic in nature.

The objective in fair division problems is to allocate a set of goods (resources) to a set of n agents in a way that leaves every agent satisfied. In the continuous case, the available resources are typically represented by the interval [0, 1], whereas in the discrete case, we have a set of distinct, indivisible goods. Combinations of divisible and indivisible goods may also be allowed. Each agent has a valuation function, which is usually normalized to be a probability distribution on the set of goods, i.e., it is an additive function and the value of an agent for the whole set is 1. Given such a setup, many solution concepts have been proposed as to what constitutes a fair solution. Some of the standard concepts that have been studied include *proportionality, envy-freeness, equitability* and many variants of them. The most related concept to our work is proportionality, meaning that every agent receives a bundle of the goods that is worth at least $1/n$, according to his valuation function.

In the continuous case, it has long been shown that proportional allocations always exist [5]. In the presence of indivisible goods however, this is not the case. Proportional allocations may still exist in cases where there is not much conflict between the preferences of the agents. But if, for example, we have two agents who have a very high value for one of the available goods, then any allocation will leave one of the two people unhappy. In this work we are interested in finding allocations with worst case guarantees in instances with indivisible goods. In particular, we focus on algorithmic and mechanism design aspects of this problem.

1.1 Related Work and Contribution

Consider a set of m indivisible items, and a set of n agents with additive valuations for the goods, as is usually the case in the fair division literature. The problem can then be described by a matrix V, where v_{ij} denotes the value of agent i for good j. Given that proportional allocations do not always exist for indivisible goods, a natural question is whether we can provide any lower bound as to the value that we can ensure to every agent. This question was studied by Hill in [7], where an explicit such guarantee was given. In particular, a certain function was identified, denoted by $V_n(\cdot)$ (defined in Section 2), such that, when the maximum value in the matrix V is at most α, then there always exists an allocation where every agent can receive a bundle that is worth at least $V_n(\alpha)$, according to his valuation. Clearly when α is large, $V_n(\alpha)$ may be 0, e.g., in the case where two agents value only one out of the m goods, hence $\alpha = 1$. For smaller values of α however this is a positive result, showing that we can ensure a relatively fair solution.

Regarding the complexity of finding such allocations, the result of [7] does not yield an efficient algorithm. The proof is based on certain combinatorial arguments, at the heart of which, lies an NP-hard problem (see the discussion in Section 3). Motivated by this fact, we start in Section 3 with studying the question of efficiently producing allocations that respect the bound of $V_n(\alpha)$. Our main result of Section 3 is that (i) we can have a slight strengthening of the guarantee of [7] so that every agent can have a bundle worth at least $V_n(\alpha_i)$, where α_i is the maximum value in the valuation of agent i (hence an agent is not penalized if someone else has a much higher maximum value than him) (ii) we provide a simple polynomial time algorithm for computing such an allocation. The algorithm is derived after first delving into understanding better the behavior of the function $V_n(\cdot)$, as explained in Section 3.

In Section 4, we move to mechanism design aspects. As with the majority of the cake-cutting literature, we do not allow any side-payments here, hence the problem falls within the realm of mechanism design without money, e.g., see [11][chapter 10]. We first show that no deterministic truthful mechanism can guarantee an allocation that is worth at least $2/3 \cdot V_n(\alpha_i)$ for every agent i, even for the case of two agents. The proof of this statement is achieved in two steps: first we argue about *permutation-respecting* mechanisms, i.e., mechanisms that when faced with a permutation of a given input, return a permutation of the initial output. We later use this to argue about general mechanisms. This approach may be of independent interest for proving other negative results in this context. We then complement this negative result by a simple algorithm showing that for two agents and a small number of goods, we can have a truthful, constant approximation to $V_n(\alpha_i)$ (becoming almost tight with the lower bound in the case of 4 goods). It still remains as an open question, whether we can achieve a constant approximation as the number of goods becomes larger, even for two agents. in Subsection 4.2, we turn to randomized algorithms. The picture is far less clear there and in general, there have been very few attempts for randomized algorithms in cake-cutting, such as [6, 3, 10]. We focus on *truthful in expectation* mechanisms and present an impossibility result for Pareto-efficient mechanisms under certain symmetry requirements.

Finally, in Section 5, we study a slightly different question. Since proportional allocations do not always exist, can we at least decide when this is the case? And can we compute them when it is so? In [4] it has already been proved that deciding if for a given instance there exists a proportional allocation, is NP-hard. We strengthen this negative result by providing a different reduction showing that it is NP-hard to decide even if there exists an allocation where every person receives a bundle of value at least $1/cn$ for any constant $c \geq 1$.

We should also note here that a related problem is the problem of computing max-min fair allocations, where the objective is to maximize the value of the least happy person. This is an NP-hard problem and several approximation algorithms have been proposed [1], mainly based on linear programming. In our case, our algorithmic problem is less demanding, since we do not want to compute an optimal (or approximately optimal) max-min fair allocation but simply an allocation that reaches the threshold of $V_n(\alpha_i)$ for every agent i. An approximate solution to max-min fairness problem does not necessarily achieve this.

2 Definitions and Preliminaries

Let $N = \{1, ..., n\}$ be a set of n agents and $M = \{1, ..., m\}$ be a set of m indivisible goods. The input to our problem is a valuation matrix V so that v_{ij} is the utility derived by agent i for obtaining good j. We assume the usual normalization in the fair division context that $\sum_j v_{ij} = 1$. Let $S_{n,m}$ be the set of $n \times m$ matrices satisfying the above requirement, i.e., the set of row stochastic matrices.

An allocation of the goods is denoted by a tuple $(S_1, ..., S_n)$, where S_i is the set allocated to player i, such that $\bigcup_i S_i = M$ and $S_i \cap S_j = \emptyset$, (implying that the algorithm has to allocate the whole set M). The total value of player i for an allocation $(S_1, ..., S_n)$, is $v_i(S_i) = \sum_{j \in S_i} v_{ij}$. In [7], Hill defined the following function.

Definition 1 *Given any integer $n \geq 2$, let $V_n : [0,1] \to \left[0, n^{-1}\right]$ be the unique nonincreasing function satisfying $V_n(\alpha) = 1/n$ for $\alpha = 0$, whereas for $\alpha > 0$:*

$$V_n(\alpha) = \begin{cases} 1 - k(n-1)\alpha & \text{if } \alpha \in I(n,k) \\ 1 - \frac{(k+1)(n-1)}{(k+1)n-1} & \text{if } \alpha \in NI(n,k) \end{cases}$$

where for any integer $k \geq 1$,

$$I(n,k) = \left[\frac{k+1}{k((k+1)n-1)}, \frac{1}{kn-1}\right]$$

and

$$NI(n,k) = \left(\frac{1}{(k+1)n-1}, \frac{k+1}{k((k+1)n-1)}\right)$$

Definition 2 *For integers $n, k \geq 1$, let $r(n,k) = \frac{1}{kn-1}$, i.e., the right endpoint of the interval $I(n,k)$. Similarly let $l(n,k) = \frac{k+1}{k((k+1)n-1)}$, the left endpoint of $I(n,k)$.*

Example 1 *In Figure 1, one can see the function $V_n(\cdot)$ for $n = 2$ and $n = 3$. For larger n, it has a similar form. The function alternates between segments where it is strictly decreasing (and the decrease is linear in α), and segments where it is constant and equal to the value at the left endpoint of the previous decreasing segment. The intervals $I(n,k)$ in Definition 1 correspond to the decreasing segments, whereas the intervals $NI(n,k)$ correspond to the constant segments. For example, for $n = 2$, looking at the function from right to left, we can see that the rightmost decreasing segment is $I(2,1) = [2/3, 1]$, which is followed by $NI(2,1) = (1/3, 2/3)$, followed by $I(2,2) = [0.3, 1/3]$, then followed by $NI(2,2) = (1/5, 0.3)$, and so on.*

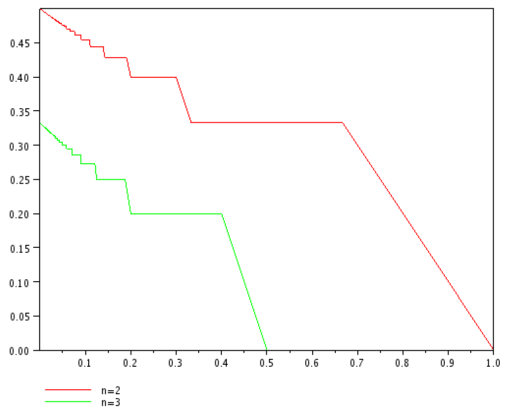

Figure 1: The function $V_n(\cdot)$ for $n = 2$ and $n = 3$.

The function $V_n(\cdot)$ has a number of useful properties that will be needed in the following sections:

Fact 1

1. *$V_n(\frac{1}{n-1}) = V_n(r(n, 1)) = 0$, and for any $\alpha \geq \frac{1}{n-1}$, $V_n(\alpha) = 0$.*
2. *The equation $\alpha = V_n(\alpha)$ has a unique solution at $\alpha = \frac{1}{2n-1} = r(n, 2)$. For $\alpha < r(n, 2)$, $V_n(\alpha) > \alpha$ and for $\alpha > r(n, 2)$, $V_n(\alpha) < \alpha$.*

In [7] it was proved that:

Theorem 2 *[7] For any instance, with $\max_{i,j} v_{ij} \leq \alpha$, there is an allocation where the total value of every player is at least $V_n(\alpha)$.*

Example 2 *Suppose $n = 2$. For the family of instances, where the maximum value for a single good is at most $1/3$, then $V_2(1/3) = 1/3$, implying that no matter what the rest of the input is, and independent of the number of goods, both agents can have a bundle worth at least $1/3$ to them. The worst possible case is achieved when there are 3 goods equally valued by both players. When we look at the family of instances with maximum value at most $1/2$, then this contains the previous family with $\alpha \leq 1/3$, hence the worst case should be at least $1/3$. By Figure 1, we see that $V_2(1/2)$ is again $1/3$, and the worst case is the same as before. More generally, for $\alpha \in NI(n, k)$ one of the the worst cases for the family of instances where $\max_{i,j} v_{ij} \leq \alpha$ is achieved at the point to the left of $NI(n, k)$, which is the right endpoint of $I(n, k + 1)$.*

Remark 1 *The result of [7] holds in a more general model, where the valuations of the agents are probability distributions on $[0, 1]$ which are allowed to have atoms of size at most α. This includes our setting.*

3 The algorithm

The proof of Theorem 2 does not imply an efficient algorithm for producing the desired allocation. Part of the proof relies on the solution of an NP-hard problem (in particular, finding max-min fair allocations, see Proposition 3.3 of [7]). Furthermore, in the tight example provided in [7], all agents have the same maximum valuation, α. It could still be feasible to provide better guarantees to agents whose maximum value is lower than the maximum value over all agents, i.e., the fact that some agents have very high values should penalize only themselves.

The main result of this Section is the following theorem, which provides (i) a slight strengthening of [7] in terms of the guarantee that each agent can have, and (ii) a simple, efficient algorithm for finding such allocations.

Theorem 3 *There exists a polynomial time algorithm that for any instance, it produces an allocation such that, each player i receives a bundle with total value at least $V_n(\alpha_i)$, where $\alpha_i = \max_j v_{ij}$.*

The algorithm is achieved by obtaining a better understanding of the behavior of $V_n(\cdot)$. Although the algorithm itself turns out to be quite simple, the analysis is more involved and is based on a series of Lemmas. We start our analysis with proving some useful properties of the function $V_n(\cdot)$. The next simple fact, which will be used repeatedly when arguing about $V_n(\alpha)$ for $\alpha \in [0,1] \setminus \bigcup_{k=1}^{\infty} I(n,k)$, i.e., in the constant segments, simply says that for $\alpha \in NI(n,k)$, $V_n(\alpha)$ is the same as the value at the right endpoint of the decreasing segment to the left of α, or as the value at the left endpoint of the decreasing segment to the right of α.

Fact 4 *If $\alpha \in NI(n,k)$, for some $k \geq 1$, then $V_n(\alpha) = V_n(r(n,k+1)) = V_n(l(n,k))$.*

The main ingredients that make our algorithm work are the properties stated in Lemma 5 and Lemma 6. Lemma 5 below provides a relation between $V_n(\cdot)$ and $V_{n-1}(\cdot)$, which is needed for the inductive analysis of the algorithm later on.

Lemma 5 *For fixed $n \geq 3$ and $k \geq 1$, and for $\alpha \in I(n,k) \cup NI(n,k)$,*

$$(1-k\alpha)V_{n-1}(\frac{\alpha}{1-k\alpha}) \geq V_n(\alpha) \tag{1}$$

Proof : First we argue that the denominator, $1-k\alpha$, in the statement of the Lemma is strictly positive. If not, then $\alpha \geq 1/k$. But $\alpha \in I(n,k) \cup NI(n,k)$, which implies that $\alpha \leq r(n,k)$. Hence

$$\frac{1}{k} \leq \frac{1}{kn-1} \Leftrightarrow k \leq \frac{1}{n-1}$$

Since k is an integer, this is possible only when $n = 2$, but we have assumed that $n \geq 3$. We now consider two cases for the value of α.

Case 1: $\alpha \in I(n,k)$

We will first show that when $\alpha \in I(n,k)$ then $\frac{\alpha}{1-k\alpha} \in I(n-1,k)$.

When $\alpha \leq r(n,k)$ then

$$\frac{\alpha}{1-k\alpha} \leq \frac{\frac{1}{kn-1}}{1-\frac{k}{kn-1}} = \frac{\frac{1}{kn-1}}{\frac{kn-1-k}{kn-1}} = \frac{1}{k(n-1)-1} = r(n-1,k)$$

When $\alpha \geq l(n,k)$ then

$$\frac{\alpha}{1-k\alpha} \geq \frac{\frac{k+1}{k((k+1)n-1)}}{1-\frac{k(k+1)}{k((k+1)n-1)}} = \frac{\frac{k+1}{k((k+1)n-1)}}{\frac{k^2n+kn-k-(k^2+k)}{k((k+1)n-1)}} = \frac{k+1}{k(kn+n-k-2)} =$$

$$= \frac{k+1}{k(k(n-1)+n-1-1)} = \frac{k+1}{k((k+1)(n-1)-1)} = l(n-1,k)$$

Hence, $\alpha/(1-k\alpha) \in [l(n-1,k), r(n-1,k)] = I(n-1,k)$. Thus, by Definition 1 we have:

$$(1-k\alpha)V_{n-1}(\frac{\alpha}{1-k\alpha}) = (1-k\alpha)(1-k(n-2)\frac{\alpha}{1-k\alpha}) = 1-k(n-1)\alpha = V_n(\alpha)$$

Note that we get exact equality in this case.

Case 2: $\alpha \in NI(n,k)$

By Fact 4, if $\alpha \in NI(n,k)$ then $V_n(\alpha) = V_n(l(n,k)) = V_n(r(n,k+1))$. Let $\tilde{a} = l(n,k)$. From Case 1 it holds that $(1-k\tilde{a})V_{n-1}(\frac{\tilde{a}}{1-k\tilde{a}}) = V_n(\tilde{a})$. But, since $\alpha < \tilde{a}$ and $V_n(\cdot)$ is nonincreasing:

$$(1-k\alpha)V_{n-1}(\frac{\alpha}{1-k\alpha}) > (1-k\tilde{a})V_{n-1}(\frac{\tilde{a}}{1-k\tilde{a}}) = V_n(\tilde{a}) = V_n(\alpha)$$

□

The next Lemma is crucial as it provides an upper bound on the left-over value of the remaining items, according to the preferences of a given agent i, once we satisfy the agent. Clearly the upper bound holds for any other agent that cannot be satisfied by the bundle allocated to agent i.

Lemma 6 *For $n \geq 2$, let i be an agent with $\alpha_i < V_n(\alpha_i)$. Suppose we start allocating items to player i, starting from his most desirable item and proceeding in decreasing order, as induced by his valuation, until the total value for i, say s, equals or exceeds $V_n(\alpha_i)$. Then,*

$$s \leq \begin{cases} k\alpha_i \; \left(= \frac{1-V_n(\alpha_i)}{n-1}\right), & \textit{if } \alpha_i \in I(n,k), \textit{ for some } k \geq 1 \\ k \cdot l(n,k), & \textit{if } \alpha_i \in NI(n,k), \textit{ for some } k \geq 1 \end{cases}$$

Proof :

Case 1: $\alpha_i \in I(n,k)$ for some $k \geq 1$.

Suppose the statement of the lemma is not true, i.e., once we surpass $V_n(\alpha_i)$ for agent i, we have $s > k\alpha_i$. Then, before the final item is given to player i, the t first items had total value c, where

$$V_n(\alpha_i) > c > (k-1)\alpha_i \tag{2}$$

Otherwise it would not be possible to exceed $k\alpha_i$ with the last item. Note that we can assume that $t \geq 1$, since $\alpha_i < V_n(\alpha_i)$. Let S be the set of these t items, and let w be their average value, $w = \sum_{j \in S} v_{ij}/t$. We can write their total value as $c = tw$, and we can easily see that $\alpha_i \geq w \geq k\alpha_i - V_n(\alpha_i) = kn\alpha_i - 1$. The leftmost inequality is trivial. As for the rightmost inequality, since we examine the goods in the decreasing order of agent i's valuation, the next good allocated to i after the first t goods would have value at most w. But then if $w < k\alpha_i - V_n(\alpha_i)$, it would be impossible to exceed $k\alpha_i$.

Given this range for the value of w, we now consider the possible values that t can take. Even if all t items had value exactly α_i (i.e., $w = \alpha_i$), we would still need more than $k-1$ items to get to c, by (2). On the other hand, if all items had value exactly $kn\alpha_i - 1$, we would need strictly less than $\frac{V_n(\alpha_i)}{kn\alpha_i - 1}$ items, again by (2). Hence,

$$\frac{1 - k(n-1)\alpha_i}{kn\alpha_i - 1} > t > k - 1 \tag{3}$$

Let x be the difference between the upper and the lower bounds of t:

$$x = \frac{1 - k(n-1)\alpha_i}{kn\alpha_i - 1} - k + 1 = \frac{k\alpha_i}{kn\alpha_i - 1} - k$$

We show that $x \leq 1$. If $x > 1$, then we would get that: $\frac{k\alpha_i}{kn\alpha_i - 1} > k + 1 => k\alpha_i > k^2 n\alpha_i + kn\alpha_i - k - 1 => \alpha_i < \frac{k+1}{k((k+1)n-1)} = l(n,k)$, which is impossible, since $\alpha_i \in I(n,k)$.

Since $x \leq 1$, the right hand side of (3) can be at most k. But this means that t, which is an integer, is greater than $k - 1$ and strictly less than k, a contradiction.

Case 2: $\alpha_i \in NI(n,k)$ for some $k \geq 1$.

The idea is to reduce the proof of this case to case 1. For this we will use the properties that when $\alpha_i \in NI(n,k)$, we know that $\alpha_i < l(n,k)$ and also that $V_n(\alpha_i) = V_n(l(n,k)) = V_n(r(n,k+1))$ by Fact 4. Suppose as in Case 1, that once we surpass $V_n(\alpha_i)$ for agent i, we have $s > k \cdot l(n,k)$. Then, before the final item is given to player i, the first t items had total value c, where

$$V_n(\alpha_i) > c > (k-1) \cdot l(n,k) \tag{4}$$

The rightmost inequality holds because $\alpha_i < l(n,k)$, hence otherwise it would not be possible to exceed $k \cdot l(n,k)$ with the last item. Note that we can assume that $t \geq 1$, since $\alpha_i < V_n(\alpha_i)$. Let S be the set of these t items, and let w be their average value, $w = \sum_{j \in S} v_{ij}/t$. We can write their total value as $c = tw$, and we can see that

$$l(n,k) > \alpha_i \geq w \geq k \cdot l(n,k) - V_n(l(n,k)) = kn \cdot l(n,k) - 1$$

The inequalitie to the left of w are trivial. As for the rightmost inequality, since we examine the goods in the decreasing order of agent i's valuation, the next good allocated to i after the first t goods would have value at most w. But then if $w < k \cdot l(n,k) - V_n(\alpha_i) = k \cdot l(n,k) - V_n(l(n,k))$, it would be impossible to exceed $k \cdot l(n,k)$.

Given this range for the value of w, we now consider the possible values that t can take. Even if all t items had value exactly $\alpha_i < l(n,k)$, we would still need more than $k-1$ items to get to c, by (4). On the other hand, if all items had value exactly $kn \cdot l(n,k) - 1$, we would need strictly less than $\frac{V_n(\alpha_i)}{kn \cdot l(n,k)-1}$ items, again by (4). Hence,

$$\frac{V_n(\alpha_i)}{kn \cdot l(n,k) - 1} = \frac{V_n(l(n,k))}{kn \cdot l(n,k) - 1} = \frac{1 - k(n-1) \cdot l(n,k)}{kn \cdot l(n,k) - 1} > t > k - 1 \tag{5}$$

Hence, we have arrived at the same inequalities for t, as in Equation (3) of Case 1, except that instead of α_i, we now have $l(n,k)$. But in case 1, these inequalities led to a contradiction for any $\alpha_i \in I(n,k)$, therefore we are done. □

Algorithm 1 ALLOCATE(V, N, M)

Input: $V \in S_{n,m}$, N (set of agents), M (set of goods)
Output: Allocation of items such that agent i receives a bundle worth at least $V_n(\alpha_i)$, $\forall i$.

```
1: for $i = 1$ to $n$ do
2:     Set $S_i = \emptyset$
3: end for
4: while $\nexists\, i$ with $v_i(S_i) \geq V_n(\alpha_i)$ do
5:     for every player $i$ do
6:         $S_i$ = $S_i \cup$ {next highest item in agent $i$'s order}
7:     end for
8: end while
9: Pick an agent $i$ with $v_i(S_i) \geq V_n(\alpha_i)$  //pick arbitrarily in case of ties
10: Allocate $S_i$ to agent $i$
11: if $|N| = 2$ then
12:     Allocate all other items to remaining agent
13: else
14:     for every row $k \neq i$ do
15:         row $k$ = (row $k$) * $1/(1 - v_k(S_i))$  //normalization before going to next round
16:     end for
17:     $V'$ = new normalized matrix after also removing row $i$ and columns corresponding to $S_i$
18:     run $ALLOCATE(V', N \setminus \{i\}, M \setminus S_i)$
19: end if
```

Proof of Theorem 3 :

We are now ready to prove the main result of this Section. The previous lemmas give rise to a simple algorithm seen above. In short, the algorithm satisfies first the player i that needs the least number of items to achieve $V_n(\alpha_i)$. This is done by maintaining for each agent, a decreasing ordering of the goods, as induced by the agent's valuation. Once the algorithm finds such an agent i, the corresponding items are given to i, who is then removed from this process. We then perform a normalization so that the remaining items add to 1 for all the remaining players, and we start again, trying to find an agent j that needs the least number of items to achieve $V_{n-1}(\tilde{\alpha_j})$, where $\tilde{\alpha_j}$ is the highest value of j, after the normalization. The algorithm continues in the same fashion until everybody is satisfied. Our algorithm is similar to a discrete analog of the procedure by Banach and Knaster in [13]. One difference however is that we need to maintain for each agent a different ordering of the goods according to their preferences, instead of using a common ordering during the process of allocating the goods.

The algorithm clearly terminates in a polynomial number of steps. In fact it can be implemented in a more efficient way without recursion, but we present it in this way, so as to be in line with the inductive proof of correctness that we provide. Finally, regarding the implementation, one also needs to ensure that the value $V_n(\alpha)$ is efficiently computable for rational inputs. This can be easily established.

Claim 7 *For any rational number $\alpha \in [0,1]$, $V_n(\alpha)$ can be computed in polynomial time.*

Proof : Given α, we need to find the integer k, for which $\alpha \in I(n,k) \cup NI(n,k)$. When $\alpha \in I(n,k) \cup NI(n,k)$, for some $k \geq 1$, then $\alpha \leq r(n,k)$. By setting $k = \lfloor \frac{1+\alpha}{\alpha n} \rfloor$, which is the solution of $r(n,k) = \alpha$, it holds that $\alpha \in (r(n,k+1), r(n,k)]$. Knowing this, finding $V_n(\alpha)$ is trivial. $\square$

The proof of correctness of the algorithm is by induction on the number of players, n:

• *Induction Basis:* $n = 2$. Without loss of generality we can assume that the first player receives the first bundle allocated by the algorithm, and that it consists of t items. Then we know that this bundle is worth at least $V_2(\alpha_1)$ for agent 1 and he is settled. We now argue about the second agent.

Suppose that $\alpha_1 < V_2(\alpha_1)$, which implies that $t > 1$ and $\alpha_2 < V_2(\alpha_2)$, otherwise the algorithm would have allocated first to agent 2 his most desirable item. Let s_2 be the value of the t items allocated to agent 1, according to the valuation of agent 2. We know that even with his $t-1$ most desirable items, agent 2 cannot exceed $V_2(\alpha_2)$, otherwise the algorithm would have chosen him first. Hence Lemma 6 can be applied.

Case 1: $s_2 \geq V_2(\alpha_2)$. s_2 is at most as much as the value of the t most desirable items of agent 2. If $\alpha_2 \in I(n,k)$, for some $k \geq 1$, then Lemma 6 gives $s_2 \leq 1 - V_2(\alpha_2)$, i.e., $1 - s_2 \geq V_2(\alpha_2)$. If $\alpha_2 \in NI(n,k)$, for some $k \geq 1$, then we have that $s_2 \leq k \cdot l(n,k)$. Hence

$$1 - s_2 \geq 1 - k \cdot l(n,k) = \frac{(k+1)(n-1) - 1}{(k+1)n - 1}$$

It is now an easy calculation, given the definition of $V_n(\cdot)$ for $\alpha \in NI(n,k)$ to show that $1 - s_2 \geq V_2(\alpha_2)$. Hence, for all possible values of α_2 under this case, we have $1 - s_2 \geq V_2(\alpha_2)$. Since the bundle allocated to agent 1 is worth s_2 to agent 2, the remaining items are worth to him at least $V_2(\alpha_2)$.

Case 2: If $s_2 < V_2(\alpha_2) \leq 1/2$, the remaining items will surely be worth at least $V_2(\alpha_2)$.

Suppose now that $\alpha_1 \geq V_2(\alpha_1)$. Then $t = 1$, hence it suffices to show that we can satisfy agent 2 with the remaining items. If $\alpha_2 \geq V_2(\alpha_2)$, this implies that $\alpha_2 \geq 1/3$, and $V_2(\alpha_2) \leq 1/3$, since the equation $\alpha_2 = V_2(\alpha_2)$ has a solution at $1/3$, by Fact 1. If $\alpha_2 \in [1/3, 2/3]$, all the remaining items are worth to him at least $1 - \alpha_2 \geq \frac{1}{3} \geq V_2(\alpha_2)$. If $\alpha_2 > \frac{2}{3}$, then $V_2(\alpha_2) = 1 - \alpha_2$, because $\alpha_2 \in I(n,1)$, see Figure 1. Hence again all the remaining items are worth to him at least $V_2(\alpha_2)$. Finally, if $\alpha_2 < V_2(\alpha_2) \leq 1/n = 1/2$, the remaining items will surely be worth more than $V_2(\alpha_2)$.

• *Induction Step:* Suppose the algorithm is correct for $n-1$ agents and consider an instance with n agents. Without loss of generality we can assume that player 1 gets the first bundle of t items, which is worth to him at least $V_n(\alpha_1)$.

Case 1: $\alpha_1 < V_n(\alpha_1)$

Since player 1 gets the items first, $\alpha_i < V_n(\alpha_i)$ holds for all i. The recursion of the algorithm for the remaining $n-1$ players guarantees by the induction hypothesis that for $i = 2, ..., n$, agent i will end up with an allocation worth at least $V_{n-1}(\tilde{\alpha_i})$, where $\tilde{\alpha_i}$ is the new highest value of agent i after the normalization in Line 15. Let s_i be the total value agent i has for the t items that agent 1 received. Due to the normalization, it suffices to show that $(1 - s_i)V_{n-1}(\tilde{\alpha_i}) \geq V_n(\alpha_i)$. But $\tilde{\alpha_i} \leq \frac{\alpha_i}{1-s_i}$, and since V_n is nonincreasing, we need to show that

$$(1 - s_i)V_{n-1}(\frac{\alpha_i}{1 - s_i}) \geq V_n(\alpha_i) \tag{6}$$

The agents $2, ..., n$ may belong to one of the following two groups:

(i) Consider an agent i with $\alpha_i \in I(n,k)$, for some $k \geq 1$. Since at the end of the $(t-1)$-th round, no player was selected by the algorithm, the $t-1$ most desirable goods for agent i do not exceed $V_n(\alpha_i)$. Hence Lemma 6 can be applied for the t highest items of i, which in turn implies that $s_i \leq k\alpha_i$. Substituting and by using Lemma 5, we see that (6) holds.

(ii) Consider an agent i with $\alpha_i \in NI(n,k)$, for some $k \geq 1$. Then by Lemma 6, we have $s_i \leq k \cdot l(n,k)$. Also, $\alpha_i \leq l(n,k)$, and V_n is nonincreasing. Hence to show (6), it suffices to show

$$(1 - k \cdot l(n,k))V_{n-1}(\frac{l(n,k)}{1 - k \cdot l(n,k)}) \geq V_n(\alpha_i)$$

Since $l(n,k) \in I(n,k)$, by Lemma 5, and by Fact 4 we have

$$(1 - k \cdot l(n,k))V_{n-1}(\frac{l(n,k)}{1 - k \cdot l(n,k)}) \geq V_n(l(n,k)) = V_n(\alpha_i)$$

Case 2: $\alpha_1 \geq V_n(\alpha_1)$. Then agent 1 receives only one good, his most desirable item. Consider an agent $i \in \{2, ..., n\}$. Suppose $\alpha_i \in I(n,k) \cup NI(n,k)$, for some $k \geq 1$.

Since agent 1 receives only one good, $s_i \leq a_i$, where s_i is, as in Case 1, the value of i for the item allocated to agent 1. Hence $s_i \leq k\alpha_i$. The recursive call for the $n-1$ players guarantees to i, $V_{n-1}(\tilde{\alpha_i})$, and $\tilde{\alpha_i} \leq \alpha_i/(1-s_i) \leq \alpha_i/(1-k\alpha_i)$. By (6), all we need to show is

$$(1 - k\alpha_i)V_{n-1}(\frac{\alpha_i}{1 - k\alpha_i}) \geq V_n(\alpha_i)$$

But this holds because of Lemma 5. □

The examples provided in [7] with all agents having the same α_i show that Theorem 3 is tight.

Remark 2 *Our algorithm can be applied to the more general model of valuations described in [7]. This can be done by using Lemma 2.2 of [7].*

4 Mechanism Design

4.1 Deterministic Algorithms

We now focus on designing truthful algorithms, where reporting the true valuation is a dominant strategy. Given an instance V, let $A(V)$ be the outcome of the algorithm. Let also V_i denote the i-th row of V, i.e., the declaration of player i, and V_{-i} denote the remaining matrix, excluding V_i. Let (V_i', V_{-i}) be the matrix that is produced from V, when we replace V_i with V_i'.

Definition 3 *A mechanism is truthful if for any instance V, and any other possible declaration V_i' of i:*

$$v_i(A(V)) \geq v_i(A(V_i', V_{-i}))$$

It is easy to see that Algorithm 1 is not truthful. For example, consider the instance

$$\begin{bmatrix} 0.6 & 0.1 & 0.1 & 0.1 & 0.1 & 0 & 0 \\ \frac{1}{7} & \frac{1}{7} & \frac{1}{7} & \frac{1}{7} & \frac{1}{7} & \frac{1}{7} & \frac{1}{7} \end{bmatrix}$$

We have that $V_2(0.6) = \frac{1}{3}$ and $V_2(\frac{1}{7}) = \frac{3}{7}$, hence Algorithm 1 gives item $\{1\}$ to player 1 and the rest of the items to player 2. Player 1 has an incentive to lie, by reporting $(0.32, 0.3, 0.28, 0.1, 0, 0, 0)$. The values are already sorted, and player 1 gets satisfied first again, getting items $\{1, 2\}$.

In fact as we show below, even if we ask for a large enough approximation to the value of $V_n(\alpha_i)$, no truthful algorithm can satisfy this.

Theorem 8 *Even for two agents, there is no deterministic truthful algorithm that guarantees a total value that is strictly better than $2/3 \cdot V_n(\alpha_i)$, for every player i.*

To prove the above theorem we will first argue about the impossibility of truthfulness for a restricted class of algorithms. Given an instance I, and a permutation $\pi \in S^m$, we will denote by $\pi(I)$ the instance where every row of the matrix is permuted by the permutation π. We call a deterministic algorithm *permutation-respecting with regard to instance I*, if, for every permutation π, the allocation produced by the algorithm at $\pi(I)$ is $(\pi(S_1), ..., \pi(S_n))$, where $(S_1, ..., S_n)$ is the allocation produced by the algorithm at I. Thus, such an algorithm does not depend on the identities of the goods for the specific instance but only on their values. We call an algorithm simply *permutation-respecting* if this holds for any instance I.

To prove Theorem 8, we first prove Lemma 9. Lemma 9 and more generally the approach of proving first the impossibility result for permutation-respecting mechanisms, may be of independent interest in proving other impossibility results regarding truthful algorithms in fair division (since the requirement to produce allocations with a value of more than $2/3 \cdot V_n(\alpha_i)$, simply demands that the algorithm is "reasonably" fair to all players).

Lemma 9 *If a permutation-respecting algorithm A guarantees a value that is strictly better than $2/3 \cdot V_n(\alpha_i)$ to each player i, then A cannot be truthful.*

Proof : We will argue about instances with 2 players and 4 goods. Suppose there is a permutation-respecting algorithm A that is truthful and always guarantees a value better than $2/3 \cdot V_n(\alpha_i)$, for every player i. Suppose that both players report the values

$$V' = \begin{bmatrix} \frac{2}{3} & \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{2}{3} & \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}$$

Since $V_2(2/3) = 1/3$, and $2/3 \cdot V_2(2/3) = 2/9$, the only choices for algorithm A is to give the first item to one player and all other items to the other player. Without loss of generality, we can assume that A gives the set $\{1\}$ to the first player and $\{2, 3, 4\}$ to the second player.

If the players declare

$$V = \begin{bmatrix} \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \end{bmatrix}$$

then $V_2(1/4) = 0.4$, and $2/3 \cdot V_2(1/4) > 1/4$. Hence each player should receive exactly two items, therefore the choices for A is to give player 1 one of the sets $\{1, 2\}$, $\{1, 3\}$, $\{1, 4\}$, $\{2, 3\}$, $\{2, 4\}$, $\{3, 4\}$. We will show that at least one player has an incentive to lie in every case. We analyze all cases below.

- A gives one of $\{1, 2\}$,$\{1, 3\}$,$\{1, 4\}$ to player 1 in instance V. Consider then the instance

  $$\begin{bmatrix} \frac{2}{3} & \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \end{bmatrix}$$

  Notice that in this new instance, since A must guarantee more than $2/3 \cdot V_2(\frac{1}{4}) > 0.25$ to player 2, it has to give at most two items to player 1. If the first item is not given to player 1, then player 2 would have an incentive to lie when his valuation is as in V', and report $(1/4, 1/4, 1/4, 1/4)$. Hence A has to give player 1 at least the first item. If A gives to player 1 only the first item, he can get more if he reports $(1/4, 1/4, 1/4, 1/4)$ and switches to V. If it gives him one of $\{1, 2\}$,$\{1, 3\}$,$\{1, 4\}$, then player 2 can raise his utility by reporting $(\frac{2}{3}, \frac{1}{9}, \frac{1}{9}, \frac{1}{9})$, and switching to V'. Hence there is always an incentive for lying by one of the players.

- A gives $\{2, 3\}$ or $\{2, 4\}$ to player 1 in instance V. Consider the instance

  $$\begin{bmatrix} \frac{1}{9} & \frac{2}{3} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \end{bmatrix}$$

  In analogy to the previous case, A can give to player 1 at most two items. Furthermore, player 1 has to receive the second item, as otherwise he will not achieve more than $2/3 \cdot V_n(2/3) = 2/9$. If A gives to player 1 only the second item, he has an incentive to lie, and report $(1/4, 1/4, 1/4, 1/4)$ so as to switch to instance V. If A gives one of $\{2, 1\}$, $\{2, 3\}$, or $\{2, 4\}$, to player 1, then player 2 has an incentive to lie and report $(1/9, 2/3, 1/9, 1/9)$. In this case, since A is permutation-respecting, player 2 would receive the set $\{1, 3, 4\}$. Hence in all possible events, one of the two players has an incentive to lie.

- A gives $\{3,4\}$ to player 1. Consider the instance

  $$\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{2}{3} & \frac{1}{9} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \end{bmatrix}$$

  Here, again, player 1 can receive at most two items. If player 1 does not receive the third item, then he will not achieve more than $2/3 \cdot V_n(2/3) = 2/9$. If A gives to player 1 only the third item, he has an incentive to lie, and report $(1/4, 1/4, 1/4, 1/4)$. Finally, if A gives him one of $\{3,1\}$,$\{3,2\}$,$\{3,4\}$, then player 2 has an incentive to lie. By reporting $(\frac{1}{9}, \frac{1}{9}, \frac{2}{3}, \frac{1}{9})$, since A is permutation-respecting, it will give player 2 the set $\{1,2,4\}$.

This shows that in all cases, regarding the possible allocations of A in instance V, someone has an incentive to lie. Hence algorithm A cannot be truthful. □

Proof of Theorem 8 :

Suppose the statement of the theorem is not true and let A be such a truthful algorithm. We can then prove the following:

Lemma 10 *A is permutation-respecting with regard to instance V'.*

Proof : Suppose without loss of generality that in V', A produces the allocation $(\{1\}, \{2,3,4\})$. The only other choice is to produce the reverse allocation, which is handled symmetrically. The properties of A guarantee that it must produce the same allocation in the instance

$$V^1 = \begin{bmatrix} \frac{2}{3} & \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{2}{3} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}$$

To see this, suppose player 2 does not receive the second item in V^1. Then he would have an incentive to switch from V^1 to V'. If player 2 is allocated the first item in V^1, then he would have an incentive to switch from V' to V^1. Hence he has to receive a subset of $\{2,3,4\}$. If he receives a strict subset of this set, then again he would prefer to lie when his real valuation is as seen in V^1 and switch to V'. The only choice left for A is the allocation $(\{1\}, \{2,3,4\})$.

In the instance

$$V^2 = \begin{bmatrix} \frac{1}{9} & \frac{2}{3} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{2}{3} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}$$

A has to produce the allocation $(\{2\}, \{1,3,4\})$, or else player 1 would have an incentive to lie. To see this, since the algorithm guarantees more than $2/3 \cdot V_2(2/3) = 2/9$ to both players, the only possible allocations are either to give the second item to player 1 and the remaining items to player 2, or the other way around. If player 1 does not receive the second item, then he receives items $\{1,3,4\}$. But then he would have an incentive to lie if his real valuation was as in V^1 and declare $(1/9, 2/3, 1/9, 1/9)$.

Using similar arguments we see that A must produce the allocation $(\{2\}, \{1,3,4\})$ in

$$V^3 = \begin{bmatrix} \frac{1}{9} & \frac{2}{3} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{2}{3} & \frac{1}{9} \end{bmatrix}$$

or else player 2 would have an incentive to lie.

Continuing in this manner, A must output the allocation $(\{3\}, \{1,2,4\})$ in

$$V^4 = \begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{2}{3} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{2}{3} & \frac{1}{9} \end{bmatrix}$$

otherwise player 1 would have an incentive to lie, when his real valuation is as in V^3.

Finally, it is by now straightforward that with the same arguments we can handle the remaining permutation where both players assign a value of $2/3$ to the 4th item. □

The proof of Lemma 9 only required that an algorithm is permutation-respecting with regard to instance V'. Hence by Lemma 10 the proof of Theorem 8 is complete. □

Clearly the lower bound of Theorem 8 holds when we have more than two players, since in that case we can add more players and goods to the instances used in the proof so that the added players are interested only in the added goods. Below we state a very simple algorithm that shows that the bound of Theorem 8 is almost tight when we have two agents and four goods. The same algorithm shows that we cannot have the same lower bound for the case of two or three goods. We need to have at least 4 goods for such constructions to work.

Algorithm 2

Given the reported input V, allocate to player 1 his most desirable good and allocate the remaining goods to player 2.

The mechanism is truthful, since player 1 cannot gain anything from lying, and neither can player 2.

Theorem 11 *For instances with* 2 *agents and* m *goods, Algorithm* 2 *is truthful and always returns an allocation worth at least* $\rho V_2(\alpha_i)$ *for* $i = 1, 2$, *where* $\rho = \min\{1, \frac{1}{mV_2(1/m)}\}$.

Proof : We look first at player 2. Since player 1 receives the first item, the total value of the remaining goods for player 2 will be at least $1 - \alpha_2$. We know that for $n = 2$, $V_n(\alpha_2) = 1 - k(n-1)\alpha_2 = 1 - k\alpha_2$, for some integer $k \geq 1$. But then we see trivially that $1 - \alpha_2 \geq 1 - k\alpha_2 = V_2(\alpha_2)$.

We look now at player 1. Since player 1 receives a value of exactly α_1, the ratio that is achieved is $\frac{\alpha_1}{V_2(\alpha_1)}$. For instances with m goods, we know that $\alpha_1 \geq 1/m$. Also since $V_n(\alpha)$ is non-increasing, we have that $V_2(\alpha_1) \leq V_2(1/m)$. Hence $\frac{\alpha_1}{V_2(\alpha_1)} \geq \frac{1}{mV_2(1/m)}$. □

In Table 4.1, we can see the ratio ρ that Algorithm 2 achieves as the number of goods increases. We see that this ratio equals 0.625 when $m = 4$, which is very close to the lower bound of $2/3$. The worst case of Algorithm 2 when $m = 4$, is achieved when the valuation of player 1 is $(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4})$. In general, the worst case scenario for this mechanism happens when player 1 values all items equally.

Number of goods	2	3	4	5	...	m
ratio ρ	1	1	0.625	$1/2$	...	$\frac{1}{mV_2(1/m)}$

Table 1: The guarantee provided by Algorithm 2, for various values of m.

It still remains an open problem to determine whether a constant factor approximation to $V_n(\alpha_i)$ can be achieved when the number of goods is large, even for two agents. The difficulties arise from the fact that we have a multi-parameter domain (each player submits m numbers), which makes it more challenging to argue about truthfulness, i.e., it allows for more flexibility concerning the possible ways of lying.

4.2 Randomized Algorithms

Given the negative results of the previous subsection, we initiate the study of randomized algorithms and would like to investigate whether randomization allows for truthful algorithms that provide each player with a total utility of at least $V_n(\alpha_i)$. Once we allow randomization, we need to decide on the quality of the solution that we want the algorithm to return (should it be in expectation or not?) as well as on the notion of truthfulness.

We first motivate the notions that seem more appropriate for our problem. Given an instance V reported by the players, let $\mathcal{F}(V)$ be the set of all allocations in which every player receives a bundle worth at least $V_n(\alpha_i)$, according to the reported valuations. An allocation that belongs to $\mathcal{F}(V)$ will be called *feasible*. We say that a randomized algorithm A is *universally feasible* if it always outputs a feasible allocation. Similarly, we say that an algorithm is *universally truthful* if an agent never has an incentive to lie, regardless of the randomness used by the algorithm. A mechanism that is universally truthful is a distribution over deterministic truthful mechanisms.

If we go for the most strict requirements, we would like an algorithm that is universally feasible and universally truthful. But this would not allow us to exploit the randomization of the algorithm, and given the negative results of Section 4.1, we cannot have such mechanisms. If we decide to relax the quality of the solution and ask for an algorithm that produces allocations that give each player a value of $V_n(\alpha_i)$ only in expectation, then there is a trivial way to achieve this: simply allocate all the goods to a player i uniformly at random. This is a universally truthful mechanism, where everybody in expectation receives a value of $1/n \geq V_n(\alpha_i)$.

Given the above, the most appropriate setting is to insist on universally feasible mechanisms but relax the notion of truthfulness. See also [3] for an analogous discussion in the continuous version of cake-cutting. An algorithm is *truthful in expectation* if no agent can improve his expected payoff by misreporting. Hence, we want to investigate the possibility of truthful in expectation algorithms, that are probability distributions on $\mathcal{F}(V)$, for any instance V, i.e., for any realization of the coin flips of the algorithm, a feasible allocation is produced. Note that the randomized version of Algorithm 2 (choosing a player at random), does not satisfy this.

In this subsection we exhibit that certain "reasonable" algorithms cannot achieve the goals we want, even for two agents. The family of algorithms we focus on, satisfy Pareto-efficiency and some symmetry requirements. In particular, we consider the following properties:

(P1) The algorithm outputs only Pareto-efficient outcomes, i.e., if a feasible allocation S is dominated by another feasible allocation S', in the sense that no player is worse off in S' and at least one is strictly better off, the algorithm has zero probability for S.

(P2) Two feasible outcomes that give exactly the same utilities to all players have the same probability.

(P3) Two feasible outcomes, where the utility vector of the first is a permutation of the utility vector of the second, have the same probability.

For example, in the two player case, if in some instance there are three feasible allocations, A,B and C, giving utilities $(1/2, 1/4)$, $(1/4, 1/2)$, and $(1/8, 1/2)$ to the two players respectively, then by (P1) the probability that the output is C is $Pr[C] = 0$, whereas by (P3), $Pr[A] = Pr[B]$.

An example of an algorithm with such properties is to find all feasible allocations, eliminate those that are Pareto-inefficient and select one of those left uniformly at random. Another example is again to find first all Pareto-efficient feasible allocations, then among them, select for each player an allocation that maximizes his utility (without repeating it, if it is the best allocation for two or more players) and then output one of these selected allocations uniformly at random. In general, any uniform distribution on a subset of Pareto-efficient allocations satisfies the above properties. It is however easy to construct examples showing that such algorithms are not truthful in expectation. In fact, we have the following:

Theorem 12 *Even for two agents, there is no truthful in expectation algorithm that satisfies (P1)-(P3) and guarantees, for every instance, a value of $V_n(\alpha_i)$, for every player i.*

Proof : Suppose there was such an algorithm, say A, for two-player instances. To simplify the notation, we denote by $\{x_1, x_2, \ldots, x_k\}$ the outcome where player 1 receives items $\{x_1, x_2, \ldots, x_k\}$ and player 2 receives the rest. Consider the instance

$$V = \begin{bmatrix} \frac{1}{2} & \frac{1}{4} & \frac{1}{4} & 0 \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \end{bmatrix}$$

We know that $V_2(\frac{1}{2}) = \frac{1}{3}$ and $V_2(\frac{1}{4}) = 0.4$. The set of feasible outcomes is $\mathcal{F}(V) = \{\{1\}, \{1, 2\}, \{1, 3\}, \{1, 4\}, \{2, 3\}\}$. Note that player 1 has the same utility for $\{1\}$ and $\{1, 4\}$, but since player 2 has strictly less utility for $\{1,4\}$ than for $\{1\}$, by property (P1) the outcome $\{1,4)\}$ has a zero probability by A. The same argument also holds for the outcome $\{2,3\}$. Let p be the probability assigned to the outcome $\{1,2\}$. By property (P2), A assigns the same probability to $\{(1,3\}$ as well. Finally, note that in outcome $\{1\}$, the utilities of the two players are $(1/2, 3/4)$ whereas in outcome $\{1,2\}$, these are $(3/4, 1/2)$. Hence by property (P3), $\{1\}$ has the same probability as $\{1,2\}$ and $\{(1,3\}$, implying $p = 1/3$.

Consider now the instance

$$V' = \begin{bmatrix} \frac{1}{2} & \frac{1}{4} & \frac{1}{4} & 0 \\ 0 & \frac{1}{4} & \frac{1}{4} & \frac{1}{2} \end{bmatrix}$$

The set of feasible outcomes, $\mathcal{F}(V')$, consists of $\{1\}$,$\{1,2,3\}$, $\{1,2\}$, $\{1,3\}$, $\{2,3\}$ and $\{1,4\}$. The outcomes $\{2,3\}$ and $\{1,4\}$ are not Pareto-efficient, hence they are not realized by A. Furthermore, by property (P2) and (P3), the outcomes $\{1,2\}$ and $\{1,3\}$ have the same probability, say r, and by property (P3), the outcomes $\{1\}$, and $\{1,2,3\}$ have the same probability, say r', where $2r + 2r' = 1$. The expected utility of player 2 is

$$E_2' = 2r \cdot 3/4 + r'(1 + 1/2) = \frac{3}{4}$$

Since A is truthful in expectation, E_2' must be at least the utility of player 2 if he reports a uniform valuation over the items (and hence switch to instance V). In V, the expected utility of player 2, if his real valuation is as in V', equals $1 \cdot 1/3 + (\frac{3}{4} + \frac{3}{4}) \cdot 1/3$, therefore,

$$3/4 \geq 1 \cdot 1/3 + (\frac{3}{4} + \frac{3}{4}) \cdot 1/3 = \frac{5}{6}$$

which is a contradiction. Hence such an algorithm A cannot exist.

□

5 On the complexity of finding better allocations

In this Section, we study a slightly different question. In instances where there is not much conflict, one might be able to produce allocations that exceed the worst case guarantee of $V_n(\alpha_i)$. The question that arises is whether we can compute such improved allocations on instances that admit them or even decide when do they exist. A particularly interesting case is to determine which instances admit proportional allocations, where everybody can receive a value of at least $1/n$. With this in mind, Demko and Hill [4] proved the following:

Theorem 13 *[4] It is NP-hard to decide if there exists a proportional allocation.*

This is done via a reduction from the PARTITION problem, as for two identical agents, any such allocation would imply a partitioning of the goods in two sets of equal value. Clearly there must be some value $\beta \in [V_n(\alpha), 1/n]$ ($\alpha = \max \alpha_i$), for which we can decide in polynomial time if an allocation where everybody receives at least β exists. So far, we only know that $\beta \geq V_n(\alpha)$. Below we exhibit that this value cannot be close to $1/n$. For this we use a different reduction, which is based on the reduction of [8] for the inapproximability of makespan in job scheduling.

Theorem 14 *For any constant $c \geq 1$, it is NP-hard to decide if there exists an allocation where every player receives a bundle worth at least $1/cn$.*

Proof : We give a reduction from 3-Dimensional Matching (3DM). An instance of 3DM consists of 3 disjoint sets $A = \{a_1, ..., a_m\}$, $B = \{b_1, ..., b_m\}$, $C = \{c_1, ..., c_m\}$ and a family of triples $F = \{T_1, ..., T_n\}$, where each triple contains exactly one element from A, B and C. We denote a triple of the form (a_i, b_j, c_k) by T_{ijk}. The 3DM problem is to decide whether F contains a matching, which is a collection of m triples such that each element of $A \cup B \cup C$ is covered by exactly one triple.

Given an instance I of 3DM, we construct an instance of our problem as follows: We have n agents, one for every triple of F. If an agent corresponds to the triple (a_i, b_j, c_k), then we say that he is of *type* i. We have $2m$ goods corresponding to the $2m$ elements of $B \cup C$. We also have some additional dummy goods. In particular, if there are t_i agents of type i, we have $t_i - 1$ dummy goods of type i. In total we have $n - m$ dummy goods, since $\sum t_i = n$. We now need to describe the valuations. If an agent is of type i and corresponds to the triple $T_{ijk} = (a_i, b_j, c_k)$, then his valuation for b_j and c_k is $v_i(b_j) = v_i(c_k) = 1/2cn$. The left over valuation of $1 - 1/cn$ is split evenly across the dummy goods of type i, hence his utility for each dummy good of type i is $\frac{1-1/cn}{t_i-1}$. The agents of type i have a zero value for all other goods. Note that we can assume that $t_i \geq 2$ for every type i.

We first want to ensure that every dummy good of type i has value at least $1/cn$ for agents of type i. Hence we need to show:

$$\frac{1-1/cn}{t_i-1} \geq \frac{1}{cn}$$

But this follows easily by the fact that $t_i \leq n$.

We now claim that there is an allocation where every agent has a total value of at least $1/cn$ if and only if the instance I has a 3-D matching. Suppose I has a matching. For each triple (a_i, b_j, c_k) in the matching, assign the goods b_j and c_k to the agent corresponding to this triple. These agents receive then a total value of $1/cn$. There are now $t_i - 1$ remaining agents of type i, which we can satisfy with one dummy good each.

Conversely suppose there is an allocation, where everybody has a value of at least $1/cn$. Then each dummy good of type i is allocated to an agent of the same type, otherwise we can produce a better allocation. There is also no agent receiving more than one dummy good. Hence for every type i, exactly $t_i - 1$ agents can receive a utility of $1/cn$ via the dummy goods. This means there will be exactly one agent from each type who is not covered by the dummy goods. But each such agent can only receive a value of $1/cn$ if he is allocated the goods b_j and c_k that correspond to its triple. Hence these agents are assigned the $2m$ goods of $B \cup C$. Each good of $B \cup C$ is assigned to exactly one agent and therefore the triples corresponding to the agents that do not have dummy goods form a 3-D matching. □

Remark 3 *Note that this is not in any conflict with our positive result of Theorem 3, because it can be easily shown that for the family of instances produced in the reduction, $V_n(\alpha_i) = 0$.*

6 Conclusions and Future Work

We have studied a fair division problem with indivisible goods and obtained an algorithmic version of the results of [7]. Furthermore we have investigated mechanism design aspects of this problem and provided some impossibility results for achieving such allocations with truthful algorithms. The results of Section 3 are in the same spirit as the bounds on maximum envy provided in [9] for instances with indivisible goods. However, in [9] the bounds on the envy had a much simpler form than the function $V_n(\cdot)$ here.

There are many interesting questions for future work. The most important one is to obtain a better understanding of truthful mechanisms even for restricted valuations. Surprisingly, even for two agents we are not yet aware if there exists a deterministic truthful algorithm that provides a constant factor approximation to $V_n(\alpha_i)$, when the number of goods is not $O(1)$. One of the main challenges for resolving this, is the fact that we have a multi-parameter domain (each player submits m numbers). As is the case in other contexts, arguing about truthfulness is harder in multi-parameter domains, as players have plenty of flexibility in finding possible ways of lying.

Acknowledgements

We would like to thank Ted Hill for many valuable discussions on this topic. We would also like to thank Jarek Byrka for suggesting to us the question of strengthening the guarantee in Section 3 from $V_n(\alpha)$ to $V_n(\alpha_i)$, for every agent i.

References

[1] A. Asadpour and A. Saberi. An approximation algorithm for max-min fair allocation of indivisible goods. In *ACM Symposium on Theory of Computing (STOC)*, pages 114–121, 2007.

[2] S. J. Brams and A. D. Taylor. *Fair Division: from Cake Cutting to Dispute Resolution*. Cambrige University press, 1986.

[3] Y. Chen, J. Lai, D. Parkes, and A. Procaccia. Truth, justice, and cake cutting. In *AAAI*, pages 756–761, 2010.

[4] S. Demko and T. Hill. Equitable distribution of indivisible items. *Mathematical Social Sciences*, 16:145–158, 1988.

[5] L. Dubins and E. Spanier. How to cut a cake fairly. *Amer. Math. Monthly*, 68:1–17, 1961.

[6] J. Edmonds and K. Pruhs. Balanced allocations of cake. In *Symposium on Foundations of Computer Science (FOCS)*, pages 623–634, 2006.

[7] T. Hill. Partitioning general probability measures. *The Annals of Probability*, 15(2):804–813, 1987.

[8] J. K. Lenstra, D. B. Shmoys, and E. Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Mathematical Programming*, 46:259–271, 1990.

[9] R. J. Lipton, E. Markakis, E. Mossel, and A. Saberi. On approximately fair allocations of indivisible goods. In *ACM Conference on Electronic Commerce (EC)*, pages 125–131, 2004.

[10] E. Mossel and O. Tamuz. Truthful fair division. In *Symposium on Algorithmic Game Theory (SAGT)*, pages 288–299, 2010.

[11] N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani, editors. *Algorithmic game theory*. Springer-Verlag, 2007.

[12] J. M. Robertson and W. A. Webb. *Cake Cutting Algorithms: be fair if you can*. AK Peters, 1998.

[13] H. Steinhaus. The problem of fair division. *Econometrica*, 16:101–104, 1948.