Fair Allocation of Two Types of Chores

Haris Aziz*[1], Jeremy Lindsay†[1], Angus Ritossa‡[1], and Mashbat Suzuki§[1]

[1]UNSW Sydney

Abstract

We consider the problem of fair allocation of indivisible chores under additive valuations. We assume that the chores are divided into two types and under this scenario, we present several results. Our first result is a new characterization of Pareto optimal allocations in our setting and a polynomial-time algorithm to compute an envy-free up to one item (EF1) and Pareto optimal allocation. We then turn to the question of whether we can achieve a stronger fairness concept called envy-free up any item (EFX). We present a polynomial-time algorithm that returns an EFX allocation. Finally, we show that for our setting, it can be checked in polynomial time whether an envy-free allocation exists or not.

1 Introduction

How to make allocation decisions fairly is a fundamental question that has been examined in many fields including computer science, economics, operations research and mathematics. We consider this question in the context of allocating indivisible chores among agents where each agent has additive valuations over the chores.

There are several formal criteria of fairness (see e.g., [8, 20]). Among the criteria, envy-freeness is referred to as the 'gold-standard' [10]. It requires that no agent prefers another agent's bundle to their own bundle. Although envy-freeness is a highly-desirable fairness concept, it poses several challenges. An envy-free allocation may not exist, and furthermore, it is NP-complete to check whether an envy-free allocation exists under additive valuations [3, 7]. For this reason, a major focus on fair allocation is to find relaxations of envy-freeness. A particularly attractive relaxation of envy-freeness is called *envy-freeness up to any item (EFX)* [10, 2]. However, the existence of EFX is a major open problem for goods and for chores. EFX requires that if an agent is envious of another agent, ignoring any item that lessens the envy results in envy disappearing. A weaker concept is *envy-freeness up to one item (EF1)* that requires that if an agent is envious of another agent, then

*haris.aziz@unsw.edu.au

†j.lindsay@student.unsw.edu.au

‡a.ritossa@student.unsw.edu.au

§mashbat.suzuki@unsw.edu.au

there exists some item such that ignoring the item results in envy disappearing. It is open whether an EF1 and *Pareto optimal (PO)* allocation always exists for chores.

In view of the open problem concerning the existence of EFX as well as EF1+PO allocations and the absence of positive algorithmic results regarding envy-free allocations, we turn our attention to a natural scenario of chore allocation in which there are at most two types of chores. We assume that the items can be divided into two groups A and B. Chores within the same group are identical and hence a given agent has the same value for the identical items. A natural motivating example could be a group of 4 housemates allocating monthly household chores consisting of 18 room cleaning chores and 15 cooking chores.

There are several reasons for considering the case of two chore types. Firstly, it is natural to consider restrictions on the general chore allocation under which we can achieve positive algorithmic results. For example, there are many papers that assume that agents have binary valuations for items (see, e.g., [6, 12, 5]): 0 or 1 in the case of goods and 0 and -1 in the case of chores.[1] There are also some recent papers where agents have exactly two values in the valuation functions (bi-valued utilities) [15] . In contrast, we allow the set of all agents to possibly have $2n$ different values for the set of items. Finally, two chore-types is a natural subclass of *personalized bi-valued instances* (see, e.g., [13]) in which each agent subjectively divides the items into two classes and has a corresponding value for items in each of the classes.

Contributions

We give a polynomial time algorithm for computing an EF1+fPO allocation for two chore type instances (Theorem 4.7) where fPO (fractional Pareto optimal) is a property stronger than Pareto optimality and requires Pareto optimality among all fractional outcomes. Since there are very few results known on the existence of EF1+PO allocation for chores - as the general additive valuation setting is a major open problem - we make concrete progress towards the problem by providing an affirmative answer in a restricted case. En route to our result, we also give a novel characterization of all fPO allocations in our setting.

We prove that for two chore type instances an EFX allocation exists and can be computed in polynomial time (Theorem 5.1). Our algorithm differs significantly from the natural adaptation of the goods algorithm of Gorantla et al. [15] and other existing approaches as they fail to produce an EFX allocation in our setting. Since the existence of EFX allocations for chores is not known even in the restricted setting of three agents with additive valuations, we remark that our work contributes towards the body of literature which explores this question in restricted settings.

We show that there exists a polynomial-time algorithm to check whether an envy-free allocation exists in the two chore types setting (Theorem 6.1). Note that this problem is NP-hard for general additive instances of indivisible chores [7]. Table 1 summarizes existence and complexity results under additive valuations and Figure 1 summarizes the

[1] Our assumption of two chore types does not assume that agents have zero as one of the two valuations. Zero valuations make many problems considerably easier.

logical relations and compatibility of the key concepts that we consider.

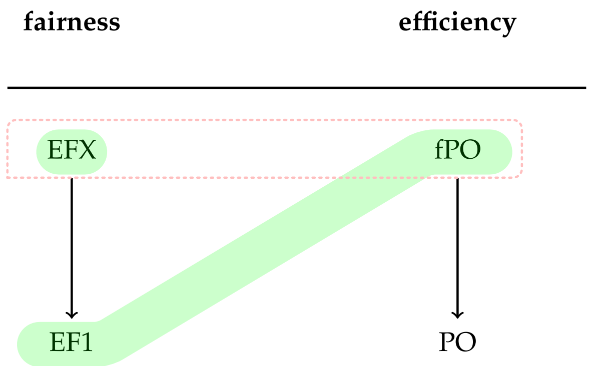

Figure 1: Logical relations between fairness and efficiency concepts. An arrow from (A) to (B) denotes that (A) implies (B). For our setting of 2 chore types, the properties in a connected solid green shape can be simultaneously satisfied, and the combined properties in connected dotted pink are impossible to simultaneously satisfy.

	EF1 & PO	EFX
Chores: general	existence open	existence open
Chores: personalised bi-valued	existence open	existence open
Chores: bi-valued	in P, exists [13, 14]	existence open
Chores: binary	in P, exists	in P, exists
Chores: 2 item types	in P, exists (Theorem 4.7)	in P, exists (Theorem 5.1)

Table 1: Existence and complexity results under additive valuations

2 Related Work

Given that an envy-free allocation may not exist, Budish [9] proposed a relaxation of envy-freeness called *envy-free up to one item (EF1)*. An allocation satisfies EF1 if it is envy-free or any agent's envy for another agent can be removed if some item is ignored. Under additive utilities, EF1 can be achieved by a simple algorithm called the round-robin sequential allocation algorithm. Agents take turn in a round-robin manner and pick their most preferred unallocated item. The interest in EF1 was especially piqued when Caragiannis et al. [10] proved that for positive additive utilities, a rule based on maximizing Nash social welfare finds an allocation that is both EF1 and Pareto optimal.

For negative additive valuations, the existence of a PO and EF1 allocation is a major open problem. Moulin [20] highlighted the problem in his survey (page 436). Except for binary utilities and bi-valued utilities ([13, 14]), the guaranteed existence of PO and EF1 allocations has not been established for other classes of valuations.

In their paper Caragiannis et al. [10] also presented the concept of EFX for goods which is strictly stronger than EF1. EFX requires that if an agent i is envious of another agent j, the envy can be removed by removing any item of j that is desirable to i. The concepts have been adapted for the case of chores or generalized to the case of mixed goods and chores (see e.g., [2, 4]). Procaccia [21] writes that the existence of EFX allocations is the biggest problem in fair division.

There are several papers that have explored the question concerning the existence of EFX allocations and have provided partial results. It is well-understood that EFX allocations exists for identical valuations. Chaudhury et al. [11] proved that an EFX allocation exists for the case 3 agents and goods. Mahara [18] showed that when items are goods and the agents have at most 2 types of valuation functions, then there exists an EFX allocation. Some of the results on sufficient conditions for the existence of EFX allocations have been extended to more general valuations [19]. On the other hand, Hosseini et al. [16] showed that when there are mixed goods and chores, then an EFX allocation may not exist. In this paper, we focus on EFX allocation of chores and identify conditions under which an EFX allocation exists. Zhou and Wu [22] presented algorithms that provide approximation of EFX for chores. Li et al. [17] considered PROPX which is a weaker property than EFX in the context of chores and they proposed algorithms for PROPX allocation of chores. One particular paper [15] focusses on positive valuations and among other results, presents an algorithm to compute an EFX allocation when there are at most two item types. The approach does not extend to the case of chores and our corresponding result requires a different approach and argument.

Garg et al. [14] and Ebadian et al. [13] examine problems in which agents have negative bi-valued valuations[2], and they both present a polynomial-time algorithm to compute an EF1 and Pareto optimal allocation. Ebadian et al. [13] also showed that for a subclass of personalised bi-valued allocations an MMS fair allocation can always be computed. Previously, Aziz et al. [1] characterized Pareto optimal allocations for positive bi-valued valuations.

3 Preliminaries

Let M be a set of m indivisible chores, and N be a set of n agents. Each agent $i \in N$ has a valuation function $v_i : M \to \mathbb{R}_{\leq 0}$, where $v_i(r)$ indicates i's value for chore $r \in M$. Throughout the paper we assume that the valuation functions are additive, i.e., for each agent $i \in N$ and for each set of chores $S \subseteq M$, $v_i(S) = \sum_{r \in S} v_i(r)$. Our main focus is to study the following class of instances:

Definition 3.1. A fair division instance $I = (N, M, v)$ is *two chore types* if the item set can be partitioned into two sets A and B with $M = A \cup B$, such that for each $i \in N$ we have $v_i(r) = v_i(r')$ for all $r, r' \in A$, and $v_i(h) = v_i(h')$ for all $h, h' \in B$.

In plain English, an instance is two chore types if there are at most two item types such that each agent is indifferent among items of the same type. Denote v_i^A as agent i's value

[2]Each agent i and item o, the valuation is either some value a or b.

for an item of type A, and v_i^B as value for an item of type B. For notational convenience, we order the agents so that $\frac{v_i^A}{v_i^B} \le \frac{v_{i+1}^A}{v_{i+1}^B}$ for all $1 \le i < n$, where we consider $\frac{v_i^A}{0}$ to be ∞.[3] More formally, this condition can be restated as $v_i^A v_{i+1}^B \le v_{i+1}^A v_i^B$. Informally, this means that agents who prefer type A items have smaller indices, and agents who prefer type B items have larger indices. We divide the agents into two sets N_A and N_B, where agents in N_A prefer type A items and agents in N_B prefer type B items. In particular, if $v_i^A \ge v_i^B$ then $i \in N_A$, and otherwise $i \in N_B$. We say that an agent $i \in N_A$ *strongly prefers A* if $2v_i^A \ge v_i^B$, and define it similarly for agents in N_B.

A valuation function is called *bi-valued* if there exist $a, b \in \mathbb{R}$ such that $v_i(h) \in \{a, b\}$ for all $i \in N$ and $h \in M$. There have been several works which focus on bi-valued valuations [14, 13]. We remark that bi-valued valuations are incomparable to two chore types valuations. Two chore type instances allow the set of agents to have $2n$ different values across agents and items whereas bi-valued instances allow for exactly two. A generalization of both bi-valued and two chore type instances is called *personalized bi-valued*, where for each agent $i \in N$ there exist $a_i, b_i \in \mathbb{R}$ such that $v_i(h) \in \{a_i, b_i\}$ for all $h \in M$. For personalized bi-valued instances, the existence of EF1+PO or EFX allocations are not known.

Allocation: An *allocation* is a partition $X = (X_1, ..., X_n)$ of the item set M, where $X_i \subseteq M$ is the bundle allocated to agent $i \in N$. An allocation is called *partial* if $\bigcup_{i \in N} X_i \ne M$. We say that the allocation is *fractional* if items are allocated (possibly) fractionally such that no more than one unit of each chore is allocated. Observe that for two chore type instances any bundle can be succinctly represented by the number of items of each type in the bundle. Thus we denote $X_i = (\alpha_i, \beta_i)$ where α_i is the number of type A items and β_i is the number of type B items in agent i's bundle. We write $(\alpha, \beta) \uplus (\alpha', \beta')$ to denote the set $(\alpha + \alpha', \beta + \beta')$ for convenience.

Fairness Notions: An allocation $X = (X_1, ..., X_n)$ is *envy-free (EF)* if for any agents $i, j \in N$, we have $v_i(X_i) \ge v_i(X_j)$. It is easy to see that EF allocations may not exist in general[4]. As a result weaker fairness notions EF1 and EFX have been introduced. An allocation X is *envy-free up to one chore* (EF1) if for any agents $i, j \in N$, where $X_i \ne \emptyset$, there exists a chore $h \in X_i$ such that $v_i(X_i \setminus h) \ge v_i(X_j)$. An allocation X is *envy-free up to any chore* (EFX) if for any agents $i, j \in N$, and for any chore $h \in X_i$ with $v_i(h) < 0$, we have $v_i(X_i \setminus h) \ge v_i(X_j)$.

Observe that EFX implies EF1, but not vice versa. We say that an agent i *EF1-envies* (respectively *EFX-envies*) another agent j if i envies j and this envy is not EF1 (respectively EFX).

Efficiency Notions: An allocation Y *Pareto dominates* another allocation X if $v_i(Y_i) \ge v_i(X_i)$ for all agents i and there exists an agent j such that $v_j(Y_i) > v_j(X_i)$. An allocation is *Pareto optimal* (PO) if it is not Pareto dominated by any allocation. An allocation is *fractionally Pareto optimal* (fPO) if it is not Pareto dominated by any fractional allocation.

[3] We assume that no agent values both item types at 0, as otherwise we can simply allocate all the chores to that agent.

[4] Consider an instance where there is one chore and two agents who have negative values for the chore

Note that an fPO allocation is also PO, but a PO allocation is not necessarily fPO.

For the remainder of the paper, we assume that all agents have strictly negative valuations for both item types. We make this assumption since if there is at least one agent who values a chore at zero then both EF1+fPO and EFX allocations can be found in a straightforward way. To see this, observe that if there is an agent i with $v_i^A = 0$ and an agent j with $v_j^B = 0$, then we can give all type A items to agent i and all type B items to agent j. In this case, every agent values their bundle at 0 and so this is trivially EF1+fPO and also EFX. On the other hand, without loss of generality, if there exists an agent i with $v_i^A = 0$, but $v_j^B < 0$ for all agents j then we assign all type A items to agent i and we assign the type B items in a round-robin way to all the agents. This gives an EFX allocation because each agent has at most one more type B item than any other agent. Additionally, this allocation is fPO since all type A items were allocated to an agent who values them at zero, and so redistributing these items cannot lead to a Pareto improvement. Furthermore, if any agent were to receive fewer type B items (possibly fractionally), a different agent must receive more type B items, and hence no Pareto improvements are possible.

4 EF1+ fPO

In this section, we present a polynomial-time algorithm which computes an EF1 and fPO allocation for the fair division problem with two chore type instances. En route, we give a novel characterization of fPO allocations in our setting.

Characterization of fPO Allocations

We begin by providing a new characterization of structure of fPO allocations by showing Lemma 4.1.

Lemma 4.1. *Given a two chore types instance where all agents have strictly negative valuations, an allocation $X = (X_1, ..., X_n)$ is fPO if and only if there exists an agent i such that:*

- *For all agents j where $\frac{v_j^A}{v_j^B} < \frac{v_i^A}{v_i^B}$, the bundle X_j only contains type A items.*
- *For all agents j where $\frac{v_j^A}{v_j^B} > \frac{v_i^A}{v_i^B}$, the bundle X_j only contains type B items.*

Proof. We first prove that any allocation which does not satisfy this criteria is not fPO. In particular, consider some (potentially fractional) allocation X which does not satisfy the criteria of the lemma. Since the criteria is not met, there must exist two agents j and k satisfying $\frac{v_j^A}{v_j^B} < \frac{v_k^A}{v_k^B}$, where X_j has a nonzero fraction of a type B item and X_k has a nonzero fraction of a type A item. Let $X_j = (\alpha_j, \beta_j)$ and $X_k = (\alpha_k, \beta_k)$.

Now, consider a sufficiently small $0 < \epsilon \leq \alpha_k$ such that $\epsilon \frac{v_j^A}{v_j^B} \leq \beta_j$. Consider the fractional allocation $X' = (X'_1, \cdots, X'_n)$, where $X'_j = (\alpha_j + \epsilon, \beta_j - \epsilon \frac{v_j^A}{v_j^B})$, $X'_k = (\alpha_k - \epsilon, \beta_k +$

$\epsilon \frac{v_j^A}{v_j^B})$ and $X'_l = X_l$ for all other agents l. Note that $v_j(X'_j) = v_j(X_j) + \epsilon v_j^A - \epsilon \frac{v_j^A}{v_j^B} v_j^B = v_j(X_j)$. Additionally, $v_k(X'_k) = v_k(X_k) - \epsilon v_k^A + \epsilon \frac{v_j^A}{v_j^B} v_k^B > v_k(X_k) - \epsilon v_k^A + \epsilon \frac{v_k^A}{v_k^B} v_k^B = v_k(X_k)$. Hence, the allocation X' is a fractional Pareto improvement over X, and so X is not fPO.

We now prove that any allocation which satisfies the criteria of Lemma 4.1 is fPO. We prove by contradiction. Consider some allocation $X = (X_1, ..., X_n)$ which satisfies the criteria with some agent i. Additionally, assume that X is fractionally Pareto dominated by some allocation $X' = (X'_1, ..., X'_n)$. From the previous paragraph, we can assume that X' also satisfies the criteria of Lemma 4.1 with some agent i': if it did not, we could apply fractional Pareto improvements until it did. Let $X_j = (\alpha_j, \beta_j)$ and $X'_j = (\alpha'_j, \beta'_j)$ for all agents j.

Note that for all allocations which satisfy the criteria of Lemma 4.1, there exists a range of possible agents i for which the lemma holds. In particular, there are two (possibly equal) agents i_L and i_R such that X satisfies the conditions of Lemma 4.1 for all $i \in [i_L, i_R]$, and does not satisfy the conditions for all $i \notin [i_L, i_R]$. Similarly, there exists such agents i'_L and i'_R for X'. We consider two cases:

First, assume there exists some agent $i \in [i_L, i_R] \cap [i'_L, i'_R]$. Then, let N_1 be the set of agents j with $\frac{v_j^A}{v_j^B} < \frac{v_i^A}{v_i^B}$, N_2 be the agents j with $\frac{v_j^A}{v_j^B} = \frac{v_i^A}{v_i^B}$ and N_3 be the agents j with $\frac{v_j^A}{v_j^B} > \frac{v_i^A}{v_i^B}$. Then, agents in N_1 receive only type A items in both X and X', and agents in N_3 receive only type B items in both X and X'. Let $X_{N_1} = \biguplus_{j \in N_1} X_j$, and define X_{N_2}, X_{N_3}, X'_{N_1}, X'_{N_2} and X'_{N_3} similarly. Since X' Pareto dominates X, it follows that $|X'_{N_1}| \leq |X_{N_1}|$ and $|X'_{N_3}| \leq |X_{N_3}|$. However, since $X_{N_1} \uplus X_{N_2} \uplus X_{N_3} = X'_{N_1} \uplus X'_{N_2} \uplus X'_{N_3}$, we know that $X_{N_2} \subseteq X'_{N_2}$. These constraints can only be satisfied if $|X'_{N_1}| = |X_{N_1}|$, $|X'_{N_2}| = |X_{N_2}|$ and $|X'_{N_3}| = |X_{N_3}|$. Therefore X' cannot Pareto dominate X: at best, all agents receive the same valuation in both allocations, which is a contradiction.

Otherwise, assume that $[i_L, i_R] \cap [i'_L, i'_R] = \emptyset$. Without loss of generality, assume that $i'_R < i_L$. Note there must exist an agent in $[i_L, i_R]$ who received a type A item in X: otherwise, X would satisfy the conditions of *Lemma* 4.1 for $i = i_L - 1$. Hence, it follows that, in X, not all of the type A items are allocated to agents in the range $[1, i'_R]$. However, in X', all the type A items are allocated to agents in the range $[1, i'_R]$. Therefore there must exist an agent $j \in [1, i'_R]$ who receives a worse bundle in X' than they do in X, which is a contradiction. □

We remark that Lemma 4.1 allows us to restrict our attention to allocations that obey the structure outlined in the lemma. In Figure 2, we give a visualisation of this structure.

$$\underbrace{\frac{v_1^A}{v_1^B} \leq ... \leq \frac{v_{i-1}^A}{v_{i-1}^B}}_{\text{Only type } A} < \underbrace{\frac{v_i^A}{v_i^B} = ... = \frac{v_j^A}{v_j^B}}_{\text{No restrictions}} < \underbrace{\frac{v_{j+1}^A}{v_{j+1}^B} \leq ... \leq \frac{v_n^A}{v_n^B}}_{\text{Only type } B}$$

Figure 2: The general form of allocations which satisfy Lemma 4.1.

Algorithm for EF1+fPO

To find an EF1 and fPO allocation, it is sufficient to consider only a subset of the allocations which satisfy Lemma 4.1. In particular, we consider a set of allocations with the following structure.

Definition 4.2. An allocation $X = (X_1, ..., X_n)$ is *ordered with respect to agent i* (or *ordered* for short) if there exists some agent i where:

- For all agents j where $j < i$, the bundle X_j only contains type A items.
- For all agents j where $j > i$, the bundle X_j only contains type B items.

We remark that all ordered allocations satisfy Lemma 4.1, but the converse does not necessarily hold (in particular, it does not always hold when there are multiple agents with identical preferences).

First, we consider an even more restricted class of allocations, namely *split-round-robin*.

Definition 4.3. Let i be an agent such that $1 \leq i < n$. The allocation *split-round-robin(i)* is the allocation formed by distributing the type A items to agents 1 through i in a round-robin way, and distributing the type B items to agents $i+1$ through n in a round-robin way. In both cases, we allocate to agents with smaller indices first.

By Lemma 4.1, the allocation *split-round-robin(i)* is fPO for all i. We introduce terminology to describe whether a *split-round-robin* allocation is EF1. Let i be an agent such that $1 \leq i < n$. We say that the allocation *split-round-robin(i)* has *A-envy* if there is an agent $j \leq i$ who has EF1-envy towards another agent $k > i$. Similarly, we say that the allocation *split-round-robin(i)* has *B-envy* if there is an agent $j > i$ who has EF1-envy towards another agent $k \leq i$.

Observe that *split-round-robin(i)* is EF1 if and only if it does not have A-envy nor B-envy. We can now begin describing our algorithm for finding an EF1 and fPO allocation. Algorithm 1 begins by checking whether *split-round-robin(i)* is EF1 for any $1 \leq i < n$. If so, then the algorithm has found an EF1 and fPO allocation. Otherwise, we create an allocation which is ordered with respect to a carefully chosen agent, which we call a *split-agent*.

Definition 4.4. An agent i is a *split-agent* if both of the following conditions hold:

- Either $i = 1$ or *split-round-robin(i-1)* has A-envy, and
- Either $i = n$ or *split-round-robin(i)* has B-envy.

Lemma 4.5. *If split-round-robin(i) is not EF1 for all* $1 \leq i < n$*, then there exists a split-agent.*

Proof. Observe that if *split-round-robin(i)* is not EF1 (for any $1 \leq i < n$), it must have A-envy or B-envy. If neither 1 nor n are split-agents, then *split-round-robin(1)* has A-envy and *split-round-robin(n-1)* has B-envy. Hence, there must exist some $1 < i < n$ such that *split-round-robin(i-1)* has A-envy and *split-round-robin(i)* has B-envy. □

We select a split-agent i^*, and will create an instance that is ordered with respect to i^*. We now explore a useful property of ordered allocations.

Lemma 4.6. *Let $I = (N, M, v)$ be a two chore types instance and X be an allocation that is ordered with respect to agent i^*. Consider a modified valuation profile $\tilde{v}$, where $\tilde{v}_j = v_{i^*}$ for all $j \in N$. If X is EF1 with respect to the modified valuation profile $\tilde{v}$ then it is EF1 in the original valuation profile v.*

Proof. As X is ordered with respect to agent i^*, any agent $j < i^*$ has only type A items i.e., $X_j = (\alpha_j, 0)$. Consider now some other agent $k \in N$. We show that if agent j does not EF1-envy k under a modified valuation $\tilde{v}_j = v_{i^*}$, then j does not EF1-envy k in the original instance.

Observe that if $\alpha_j = 0$, then agent j is not allocated any chores, and thus she does not have envy towards any other agent. Hence we assume that $\alpha_j > 0$. Since X is EF1 under the modified valuation profile, agent j does not EF1-envy k when $\tilde{v}_j = v_{i^*}$. It follows that,

$$\begin{aligned} \tilde{v}_j(\alpha_j - 1, 0) &= (\alpha_j - 1)v^A_{i^*} \\ &\geq \alpha_k v^A_{i^*} + \beta_k v^B_{i^*} \end{aligned} \tag{1}$$

Recalling $j < i^*$, we have $\frac{v^A_j}{v^B_j} \leq \frac{v^A_{i^*}}{v^B_{i^*}}$, rearranging we have that $\frac{v^A_j}{v^A_{i^*}} \leq \frac{v^B_j}{v^B_{i^*}}$. As both sides of Equation (1) are non-positive, it follows that $(\alpha_j - 1)v^A_j \geq \alpha_k v^A_j + \beta_k v^B_j$, and hence j does not EF1-envy k under the original valuation function.

We can apply a similar argument for agents $j > i^*$. □

Theorem 4.7. *Given a two chore types instance, Algorithm 1 finds an allocation that is EF1 and fPO in polynomial-time.*

Proof. First observe that the algorithm only outputs an ordered allocation and thus fPO by Lemma 4.1. Furthermore if *split-round-robin(i)* is EF1 for some i then the algorithm returns an allocation that is both EF1 and fPO immediately. Thus the main challenge is to analyse the algorithm on instances where *split-round-robin(i)* is not EF1 for any $1 \leq i < n$. In the remainder of the proof we restrict our attention to these instances.

Recall that if *split-round-robin(i)* is not EF1 for any $1 \leq i < n$ then there exists a split agent i^* by Lemma 4.5. At a high level the algorithm transfers items from the split agent to other agents until the allocation becomes EF1 whilst maintaining the allocation is ordered with respect to i^*.

Consider now a modified valuation $\tilde{v}_j = v_{i^*}$ for all $j \in N$. We show that the algorithm outputs an EF1 allocation with respect to the modified instance. By Lemma 4.6, the same allocation is also EF1 with respect to the original instance. In the modified instance, there is no EF1-envy among all agents other than i^* since their bundles are formed by repeatedly transferring an item to the agent with the highest valuation. In particular, if X is not EF1, this must be due to EF1-envy that agent i^* has for another agent, or EF1-envy that another agent has towards agent i^*.

Let X^L be the allocation when $v_{i^*}(X^L_{i^*}) \neq \min_{j \in N} v_{i^*}(X^L_j)$ holds for the first time (assuming that the Algorithm 1 does not terminate prior to this). If Algorithm 1 terminates

Algorithm 1: Computing an EF1 and fPO allocation

Input : A fair allocation instance with two chore types, where all agents have strictly negative valuations

Output: An allocation which is EF1 and fPO

for $i \leftarrow 1$ **to** $n-1$ **do**
    if *split-round-robin(i) is EF1* **then**
        return *split-round-robin(i)*

$i^* \leftarrow$ a *split agent* ; ▷ Note that such an agent is guaranteed to exist by Lemma 4.5.
$X = (X_1, ..., X_n) \leftarrow$ an allocation where $X_{i^*} = M$ and $X_j = \emptyset$ for all $j \neq i^*$.
while *X is not EF1* **do**
    $j = \arg\max_{j \in N \setminus i^*} v_{i^*}(X_j)$
    if $j < i^*$ **then**
        Transfer a type A item from X_{i^*} to X_j
    else
        Transfer a type B item from X_{i^*} to X_j
return X

prior to X^L, for simplicity we say that every allocation in the algorithm is prior to X^L. We show that for all allocations X prior to (and including) X^L, no agent has EF1-envy towards i^*. The statement holds for all allocations prior to X^L from the definition of X^L. We now prove that in the allocation X^L, no agent has EF1-envy towards i^*. Let X' be the allocation immediately prior to X^L. By definition of X', we have $v_{i^*}(X'_{i^*}) = \min_{j \in N} v_{i^*}(X'_j)$, so it follows that X' is not EF1 if and only if agent i^* has EF1-envy towards agent j, where $j = \arg\max_{j \in N \setminus \{i^*\}} v_{i^*}(X'_j)$. Note that j is the agent who was transferred an item to create X^L. Since agent i^* had EF1-envy towards j when the allocation was X', it follows that $v_{i^*}(X^L_{i^*}) < v_{i^*}(X'_j)$. For all agents $k \neq j, i^*$, their bundle is unchanged between X'_k and X^L_k. Because agent k does not EF1-envy the bundle X'_j they do not EF1-envy the even worse bundle $X^L_{i^*}$.

Claim 1: For all allocations X prior to and including X^L, we have that X_{i^*} contains at least one item of each type.

Proof of Claim 1. Recall that no agent has EF1 envy towards i^* and thus every agent other than i^* has no EF1-envy towards any agent.

We first prove that X_{i^*} has at least one type A item. If $i^* = 1$, then this is immediately true. Otherwise, assume $i^* > 1$. We proceed by contradiction. Assume that X_{i^*} has no type A items. Then, agents 1 through $i^* - 1$ have all the type A items, and agents i through n have all the type B items. However, because i^* is a split-agent, it follows that *split-round-robin*$(i^* - 1)$ has A-envy. Thus there must exist some agent $j < i^*$ which has EF1-envy towards another agent $k \geq i^*$ in X which is a contradiction.

We now prove that there is at least one type B item. If $i^* = n$, it follows immediately. Otherwise, if $i^* < n$, we can use a symmetrical argument to the type A item case. □

We show that throughout the algorithm, allocation X remains ordered with respect to i^*. This follows from Claim 1 as whenever Line 9 is reached, X_{i^*} has at least one type A item, and whenever Line 11 is reached, X_{i^*} has at least one type B item.

Finally we show that the algorithm terminates (returns an EF1 allocation) prior to or at allocation X^L. If the algorithm terminates prior to X^L then we are done. On the other hand if every allocation before X^L is not EF1 then we show that X^L must be EF1. By the definition of X^L, there exists some agent k such that $v_{i^*}(X_k^L) < v_{i^*}(X_{i^*}^L)$. Since no agent in $N \setminus i^*$ has any EF1-envy towards any other agent in $N \setminus i^*$, it follows that k does not EF1-envy any agent i.e., there exists some chore $r \in X_k^L$ such that $v_{i^*}(X_k^L \setminus r) \geq v_{i^*}(X_l^L)$ for all agents l. By Claim 1, $X_{i^*}^L$ contains at least one item of each type, and therefore contains an item r' of the same type as r. Therefore $v_{i^*}(X_{i^*}^L \setminus r') > v_{i^*}(X_k^L \setminus r) \geq v_{i^*}(X_l^L)$ for all $l \in N$. Hence indeed X^L is EF1 with respect to the modified instance.

As for time complexity, the algorithm runs in polynomial-time since the while loop on Line 6 can only run at most m times. □

EFX and fPO are not always compatible

A natural extension of Theorem 4.7 is to ask whether an allocation always exists that is EFX and fPO. Here, we disprove this by providing an instance with no allocation that is both EFX and fPO.

Consider an instance with 3 agents, where $v_1^A = -10$, $v_2^A = -11$, $v_3^A = -12$, and $v_1^B = v_2^B = v_3^B = -1$. There are 3 type A items and 2 type B items. For the allocation to be EFX, each agent must receive one type A item. Otherwise, one agent would receive at least 2 type A items and another agent would receive no type A items, which cannot be EFX. However, if the allocation is fPO it must satisfy Lemma 4.1 and so agent 3 must receive both type B items. However, this is not EFX. Hence, in this instance, there does not exist any allocation that is both EFX and fPO.

Due to this nonexistence result, we next instead consider the question of whether an EFX allocation always exists.

5 EFX

In this section, we give an algorithm to compute an EFX allocation of chores when there are two item types. Our first observation is that important algorithms for chore allocation as well natural adaptations for fair allocation of goods to the case of chores do not give EFX guarantees even for two item types. These include two algorithms ("The Top-trading Envy Cycle Elimination Algorithm" and "The Bid-and-Take Algorithm") for PROPX allocations by Li et al. [17] as well as an adaptation the algorithm of Gorantla et al. [15] to the case of chores. This is detailed in Appendix A.

The main result of this section is Theorem 5.1, which we dedicate the remainder of this section to proving.

Theorem 5.1. *For two chore type instances, an EFX allocation always exists and can be found in polynomial-time.*

We now present our algorithm for finding an EFX allocation.

5.1 Allocation algorithm when $|A| \leq |N_A|$ or $|B| \leq |N_B|$

The main case described in Section 5.2 requires $|A| > |N_A|$ and $|B| > |N_B|$, and so we begin with an algorithm for when this does not hold. Assume without loss of generality that $|A| \leq |N_A|$. Let $k = \left\lfloor \frac{|B|}{|N|} \right\rfloor$. We allocate k type B items to all agents, and let b be the number of unallocated type B items. Note that $0 \leq b < |N|$. Consider two cases, depending on b.

Case 1: $b \leq |N_B|$ We allocate 1 more type B item to b agents from N_B and allocate up to 1 type A item to agents in N_A, which gives an EFX allocation.

Case 2: $b > |N_B|$ We allocate one more type B item to all agents in N_B and to the $b - |N_B|$ agents from N_A with the largest $\frac{v_i^A}{v_i^B}$ (call these agents N'_A).

Now, let $l \geq 1$ be the largest integer such that $l v_i^A \geq v_i^B$ for all $i \in N_A \backslash N'_A$. We assign type A items to agents in $N_A \backslash N'_A$ in a round-robin way until no type A items remain or all agents in $N_A \backslash N'_A$ have $l+1$ type A items. Note that this (potentially partial) allocation is EFX due to the selection of l. Now, let $a \leq |N'_A|$ be the number of unallocated type A items. If $a > 0$, then agents in N'_A have no envy since $(l+1)v_i^A < v_i^B$ for all $i \in N'_A$, and so we allocate a type A item to a agents in N'_A, which gives an EFX allocation.

5.2 Allocation algorithm when $|A| > |N_A|$ and $|B| > |N_B|$

In this section, we prove that Algorithm 2 will always find an EFX allocation in polynomial time. We assume without loss of generality that $|N_A| \geq |N_B|$.

Algorithm 2 begins by computing an EFX partial allocation X^* on Line 3. In this initial allocation, all type B (and potentially some type A) items are allocated. However, X^* must be chosen carefully, otherwise Algorithm 2 will fail to allocate all the type A items. Computing X^* is the main challenge of the algorithm, and most of the section is dedicated to this.

Once X^* is computed, Algorithm 2 applies one of following two update rules until all the remaining type A items are allocated:

- Rule 1 (Line 8). Let a be the number of unallocated type A items and let $X' = (X'_1, ..., X'_n)$ be an allocation where $X'_i = X_i$ for all $i \in N_A$ and $X'_j = X_j \uplus (1, 0)$ for all $j \in N_B$. If $a \geq |N_B|$ and X' is EFX, then set X to be X'. We refer to the condition "X' is EFX" as the "EFX condition of Rule 1".

- Rule 2 (Line 11). If Rule 1 does not apply, then let $i \in N_A$ be an agent who does not envy any other agent (we will prove that such an agent always exists under our choice of X^*). We allocate a type A item to i.

Algorithm 2: Computing an EFX allocation

Input : A fair allocation instance with two chore types, where all agents have strictly negative valuations and $|N_A| \geq |N_B|$

Output: An EFX allocation

1 **if** $|A| \leq |N_A|$ *or* $|B| \leq |N_B|$ **then**

2   **return** the allocation described in Section 5.1

3 $X = (X_1, ..., X_n) \leftarrow X^*$, an initial partial EFX allocation, described in Section 5.3

4 **while** X *is a partial allocation* **do**

5     $a \leftarrow$ the number of unallocated type A items

6     $X' = (X'_1, ..., X'_n) \leftarrow$ an allocation where $X'_i = X_i$ for all $i \in N_A$ and $X'_j = X_j \uplus (1, 0)$ for all $j \in N_B$

7   **if** $a \geq |N_B|$ *and* X' *is EFX* **then**

8         $X \leftarrow X'$ ▷ Rule 1

9   **else**

10         $i \leftarrow$ an agent in N_A where $v_i(X_i) \geq v_i(X_j)$ for all $j \in N$

11         $X_i \leftarrow X_i \uplus (1, 0)$ ▷ Rule 2

12 **return** X

Note that both rules preserve EFX. In particular, Rule 1 preserves EFX by definition, and Rule 2 preserves EFX because any envy that agent i has will disappear if a single type A item is removed from their bundle. Hence, if Algorithm 2 returns, then the returned allocation will be EFX. Additionally, Algorithm 2 runs in polynomial time because the update rules can be applied at most m times.

Therefore, to prove the correctness of Algorithm 2 and hence Theorem 5.1, it is sufficient to show that whenever Line 10 is reached, there exists such an agent i. In this case, we say that "Rule 2 can be applied". If it is possible to apply Rule 2 k times consecutively, then we say that "Rule 2 can be applied k times". It is not true in general that Rule 2 can be applied whenever Rule 1 cannot be applied, however we will give certain conditions for X^* under which it is true.

We begin by presenting some useful properties of allocations.

Lemma 5.2. *Let i and j be two agents. If $i > j$ and X_i has at least as many type B items as X_j, then i and j cannot both envy each other. That is, if i envies j, then j does not envy i.*

Proof. First, note that $\frac{v_i^A}{v_i^B} \geq \frac{v_j^A}{v_j^B}$. Let $X_i = (\alpha_i, \beta_i)$ and $X_j = (\alpha_j, \beta_j)$. Assume that i envies j, and so

$$\alpha_i v_i^A + \beta_i v_i^B < \alpha_j v_i^A + \beta_j v_i^B.$$

Rearranging this gives

$$v_i^B(\beta_i - \beta_j) < v_i^A(\alpha_j - \alpha_i).$$

Since $\frac{v_j^B}{v_i^B} \geq \frac{v_j^A}{v_i^A}$ and the left side of the above equation is negative, it follows that

$$v_j^B(\beta_i - \beta_j) < v_j^A(\alpha_j - \alpha_i),$$

and so

$$\alpha_i v_j^A + \beta_i v_j^B < \alpha_j v_j^A + \beta_j v_j^B.$$

Hence j does not envy i. □

Lemma 5.3. *Let $i \in N_A$ and $j \in N_B$ be two agents, and let X_i and X_j be their bundles. If X_j has strictly more type B items than X_i and j EFX-envies i, then $|X_i| < |X_j| - 1$.*

Proof. let $X_i = (\alpha_i, \beta_i)$ and $X_j = (\alpha_j, \beta_j)$. Since j EFX-envies i,

$$v_j((\alpha_i, \beta_i)) > v_j((\alpha_j, \beta_j - 1)),$$

and hence

$$\alpha_i v_j^A + \beta_i v_j^B > \alpha_j v_j^A + (\beta_j - 1)v_j^B.$$

Rearranging this gives

$$(\alpha_i - \alpha_j)v_j^A > ((\beta_j - 1) - \beta_i)v_j^B.$$

Noting that $(\beta_j - 1) - \beta_i \geq 0$ and $0 > v_j^B \geq v_j^A$, it follows that

$$\alpha_i - \alpha_j < (\beta_j - 1) - \beta_i,$$

and so $\alpha_i + \beta_i < \alpha_j + \beta_j - 1$, implying that $|X_i| < |X_j| - 1$. □

We now turn our attention to finding conditions for X^* under which the update rules can always be applied.

Lemma 5.4. *Let X^* be an EFX partial allocation where all type B items are allocated. If X^* satisfies the following conditions, then the update rules can be applied until all items are allocated:*

1. *The EFX condition of Rule 1 does not hold for X^*, and*
2. *Consider a partial allocation Y formed by applying the update rules 0 or more times to X^*. Whenever the EFX condition of Rule 1 does not hold for Y, Rule 2 can be applied $|N_B|$ times to Y.*

Proof. It is sufficient to show that Rule 2 can be applied whenever Rule 1 cannot be applied.

Let a_t be the number of unallocated type A items after the update rules have been applied t times to the allocation X^*. If $a_0 < |N_B|$, then since the EFX condition of Rule 1 does not hold for X^*, it follows from the second condition of the lemma that we can apply Rule 2 until all items are allocated.

Otherwise, assume that $a_0 \geq |N_B|$. We begin by showing that immediately after Rule 1 is applied, the EFX condition of Rule 1 will no longer hold. We prove by contradiction. Assume there is a situation where Rule 1 is applied, and then the EFX condition of Rule 1 still holds, and consider the first such occurrence of this. Since Rule 2 will always be the first update rule applied (due to the first condition of the lemma), it follows that Rule 2 must have been applied immediately before Rule 1 was applied. Additionally, when we applied Rule 2, the EFX condition of Rule 1 did not hold. However this gives a contradiction: if the EFX condition of Rule 1 holds after applying Rule 2 followed by Rule 1, then it must have held prior to Rule 2 being applied, which implies that Rule 2 would not have been applied.

Now, we show that Rule 2 can be applied whenever Rule 1 cannot. Consider a situation where the update rules have been applied t times, and Rule 1 cannot be applied. If $a_t \geq |N_B|$, then Rule 2 can be applied because of the second condition of the lemma. If $a_t < |N_B|$, then consider the last update applied when $a_{t'} \geq |N_B|$ still held. If it was Rule 1, then after this update the EFX condition of Rule 1 did not hold (as we proved in the previous paragraph) and so Rule 2 can be applied until every item is allocated. If it was Rule 2, then the EFX condition of Rule 1 did not hold and so by the second condition of the lemma, Rule 2 can be applied until every item is allocated. □

We now show the following result which gives a set of conditions under which the second condition of Lemma 5.4 is satisfied.

Lemma 5.5. *Let X be an EFX partial allocation. If X satisfies the following conditions, then Rule 2 can be applied $|N_B|$ times:*

1. *For all agents $i \in N_A$ and $j \in N_B$, X_j has strictly more type B items than X_i,*
2. *For all agents $i \in N_A$ and $j \in N_B$, i does not envy j, and*
3. *Consider a partial allocation Y formed by applying the update rules 0 or more times to X. For any such allocation Y and any nonempty subset $S \subseteq N_A$, there exists some agent $i \in S$ who does not envy any other agent in S.*

Proof. We will show that we can apply Rule 2 $|N_A|$ times. Let T be the set of agents in N_A to which we haven't applied Rule 2 yet. Initially, $T = N_A$. Assume that we have applied Rule 2 less than $|N_A|$ times, and so T is non-empty. We will show that we can apply Rule 2 again.

Let $t \in T$ be an agent who does not envy any other agents in T. We can apply Rule 2 to t unless t envies some $i \in N_A \setminus T$. Therefore, assume that t envies some $i \in N_A \setminus T$. Let $X_i = (\alpha_i, \beta_i)$. Now, we know that t does not EFX-envy the bundle $(\alpha_i - 1, \beta_i)$, as this was X_i prior to Rule 2 being applied to i. However, if t envies (α_i, β_i) but does not EFX-envy $(\alpha_i - 1, \beta_i)$, then X_t must contain only type B items. However, this implies that no agent in N_A envies any agent $j \in N_B$: if they did, then they would EFX-envy t since $X_t \subset X_j$. Hence, we can apply Rule 2 to an agent in N_A who does not envy any other agent in N_A. □

Finally, we prove a result which gives an alternate set of conditions under which the update rules can always be applied.

Lemma 5.6. *Let X^* be an EFX partial allocation where all type B items are allocated. If X^* satisfies the following conditions, then the update rules can be applied until all items are allocated:*

1. *The EFX condition of Rule 1 does not hold for X^*,*
2. *$|X_i^*| = |X_{i'}^*|$ for all $i, i' \in N_B$,*
3. *For all agents $i \in N_A$ and $j \in N_B$, X_j^* has strictly more type B items than X_i^*, and*
4. *Consider a partial allocation Y formed by applying the update rules 0 or more times to X^*. For any such allocation Y and any nonempty subset $S \subseteq N_A$, there exists some agent $i \in S$ who does not envy any other agent in S.*

Proof. We use Lemma 5.4. The first condition of Lemma 5.4 is the same as the first condition of Lemma 5.6, and so we just need to show that the second condition of Lemma 5.4 is met.

Consider a partial allocation Y formed by applying the update rules 0 or more times to X^*, and assume that the EFX condition of Rule 1 does not apply to Y. We use Lemma 5.5 to show that Rule 2 can be applied $|N_B|$ times to Y. The first and third conditions of Lemma 5.5 immediately hold because they are shared with Lemma 5.6. Hence, we just need to show that the second condition of Lemma 5.5 holds.

Let $i \in N_A$ and $j \in N_B$ be agents such that j would EFX-envy i if Rule 1 was applied to Y. By Lemma 5.3, $|Y_i| < |Y_j|$.

Now, consider any $i' \in N_A$ and $j' \in N_B$. Since Y is EFX, we know that $v_{i'}(Y_{i'}) - v_{i'}^B \geq v_{i'}(Y_i)$. Because $|Y_i| < |Y_j| = |Y_{j'}|$ and Y_i has less type B items than $Y_{j'}$, it follows that $v_{i'}(Y_i) + v_{i'}^B \geq v_{i'}(Y_{j'})$. Therefore $v_{i'}(Y_{i'}) \geq v_{i'}(Y_i) + v_{i'}^B \geq v_{i'}(Y_{j'})$ and so i' does not envy j'.

Thus Lemma 5.5 holds for Y and so Lemma 5.4 holds for X^*. □

5.3 Computing X^*

In this section, we describe X^* using the results we proved in the previous section. We consider several cases, depending on the input instance.

Let a and b be the number of unallocated type A and B items respectively. Initially, $a = |A|$ and $b = |B|$.

Let $k = \left\lfloor \frac{b - |N_B|}{|N|} \right\rfloor$. We assign k type B items to all agents in N_A and $k+1$ to all agents in N_B. In particular,

$$X_i^* = \begin{cases} (0, k) & \text{for } i \in N_A, \\ (0, k+1) & \text{for } i \in N_B. \end{cases}$$

Now, $0 \leq b < |N|$. We consider two cases, depending on b.

5.3.1 Case 1: $b \geq |N_B|$

We assign one more type B item to all agents in N_B and to the $b - |N_B|$ agents from N_A who have the greatest $\frac{v_i^A}{v_i^B}$ (call these agents N'_A), and we assign one type A item to all agents in $N_A \backslash N'_A$. In particular, the partial allocation is:

$$X_i^* = \begin{cases} (1, k) & \text{for } i \in N_A \backslash N'_A, \\ (0, k+1) & \text{for } i \in N'_A, \\ (0, k+2) & \text{for } i \in N_B. \end{cases}$$

We use Lemma 5.6 to show that the update rules can be applied until all items are allocated. Note that the partial allocation is EFX and the first three conditions of Lemma 5.6 clearly hold. For the fourth condition, consider a partial allocation Y as described in the lemma, and some nonempty subset $S \subseteq N_A$. If $S \subseteq N'_A$ or $S \subseteq N_A \setminus N'_A$, then the fourth condition holds as the agent $i = \arg\min_{j \in S} |Y_j|$ does not envy any other agents in S. Otherwise, let $i = \arg\min_{j \in S \cap N'_A} |Y_j|$ and $i' = \arg\min_{j \in S \cap (N_A \backslash N'_A)} |Y_j|$. By Lemma 5.2 these agents cannot both envy each other, and so assume without loss of generality that i does not envy i'. Then, i does not envy any agents in S. Hence this allocation satisfies all the conditions of Lemma 5.6.

5.3.2 Case 2: $b < |N_B|$

We assign one more type B item to the b agents from N_B who have the greatest v_i^B (call these agents N'_B).

This gives us the following partial allocation which is not EFX:

$$X_i^* = \begin{cases} (0, k) & \text{for } i \in N_A, \\ (0, k+1) & \text{for } i \in N_B \backslash N'_B, \\ (0, k+2) & \text{for } i \in N'_B. \end{cases}$$

First, if $|A| \leq 2|N_A|$, then we allocate the type A items to agents in N_A in a round-robin way. Note that each agent in N_A will receive 1 or 2 type A items. In particular, let N'_A be the agents who receive 1 type A item. Then we will have the following EFX allocation:

$$X_i^* = \begin{cases} (1, k) & \text{for } i \in N'_A, \\ (2, k) & \text{for } i \in N_A \backslash N'_A, \\ (0, k+1) & \text{for } i \in N_B \backslash N'_B, \\ (0, k+2) & \text{for } i \in N'_B. \end{cases}$$

Otherwise, $|A| > 2|N_A|$. We consider three more subcases.

Case 2.1: Every agent $j \in N_B \backslash N'_B$ does not strongly prefer B In this case, we allocate one type A item to all agents in $N_A \cup N_B \backslash N'_B$, resulting in the following EFX partial allocation:

$$X_i^* = \begin{cases} (1, k) & \text{for } i \in N_A, \\ (1, k+1) & \text{for } i \in N_B \backslash N_B', \\ (0, k+2) & \text{for } i \in N_B'. \end{cases}$$

This partial allocation is EFX because agents in $N_B \backslash N_B'$ prefer 1 type A item over 2 type B items. We use Lemma 5.6 to show that the update rules can be applied until all items are allocated. The first three conditions of Lemma 5.6 clearly hold. For the fourth condition, consider a partial allocation Y as described in the lemma and a nonempty subset $S \subseteq N_A$. Then, the agent $i = \arg\min_{j \in S} |Y_j|$ does not envy any other agents in S. Hence this allocation satisfies all the conditions of Lemma 5.6.

Case 2.2: There are at least $|N_B|$ agents $i \in N_A$ who strongly prefer A In this case, we give one type A item to all agents in N_A, resulting in the following EFX partial allocation:

$$X_i^* = \begin{cases} (1, k) & \text{for } i \in N_A, \\ (0, k+1) & \text{for } i \in N_B \backslash N_B', \\ (0, k+2) & \text{for } i \in N_B'. \end{cases}$$

We use Lemma 5.4 to show that the update rules can always be applied. The first condition clearly holds. For the second condition, consider a partial allocation Y as described in the lemma, and assume that the EFX condition of Rule 1 does not hold for Y. Then, by Lemma 5.3 there exists some $i \in N_A$ and $j \in N_B$ such that $|Y_i| < |Y_j|$, and so $|Y_i| \leq |Y_{j'}|$ for all $j' \in N_B$. Since all agents in N_A have the same number of type B items and Y is EFX, it follows that $|Y_{i'}| \leq |Y_i| + 1 \leq |Y_{j'}| + 1$ for all $i' \in N_A$ and $j' \in N_B$. We can apply Rule 2 at least $|N_B|$ times to Y as follows:

- While there exists an agent $i \in N_A$ where $|Y_i| \leq |Y_j|$ for all $j \in N_B$, apply Rule 2 to such an agent with the smallest $|Y_i|$. This maintains EFX as i did not EFX any agent prior to the rule being applied.
- After doing the above 1 or more times, all agents $i \in N_A$ have identical bundles (with $|Y_i| \leq |Y_j| + 1$ for all $j \in N_B$). We can apply Rule 2 once to all agents who strongly prefer A. This maintains EFX as these agents will not EFX-envy any $j \in N_B$ because they prefer two type A items over a type B item.

Case 2.3: Cases 2.1 and 2.2 do not hold Since Case 2.2 does not hold, there are less than $|N_B|$ agents $i \in N_A$ who strongly prefer A. Since Case 2.1 does not hold, all agents $j \in N_B'$ strongly prefer B.

Let N_A' be the $|N_B|$ agents in N_A with the minimum $\frac{v_i^A}{v_i^B}$. We transfer one type B item from all agents in N_A' to the agents in N_B, allocate 2 type A items to all agents in N_A' and 1 type A item to all agents in $N_A \backslash N_A'$. In particular,

$$X_i^* = \begin{cases} (2, k-1) & \text{for } i \in N'_A, \\ (1, k) & \text{for } i \in N_A \backslash N'_A, \\ (0, k+2) & \text{for } i \in N_B \backslash N'_B, \\ (0, k+3) & \text{for } i \in N'_B. \end{cases}$$

First, note that the above partial allocation is EFX, since all agents in $N_A \backslash N'_A$ do not strongly prefer A, and all agents in N'_B strongly prefer B.

We use Lemma 5.4. The first condition clearly holds. For the second condition, consider a partial allocation Y as described in the lemma, and assume that the EFX condition of Rule 1 does not apply to Y. We use Lemma 5.5.

1. The first condition of Lemma 5.5 clearly holds for Y.

2. For the second condition, note that there exists some $i \in N_A$ and $j \in N_B$ such that j would EFX-envy i if Rule 1 was applied. Hence, by Lemma 5.3, $|Y_i| < |Y_j|$.

   If $j \in N_B \backslash N'_B$, this implies that all $i' \in N_A$ do not envy any agents in N_B, using the same proof as Lemma 5.6.

   If $j \in N'_B$, then we show that $|Y_i| < |Y_j| - 1$, by proving that $|Y_i| \neq |Y_j| - 1$. Let $Y_j = (\alpha_j, k+3)$ and assume $|Y_i| = |Y_j| - 1$. Then,

   $$Y_i = \begin{cases} (\alpha_j + 3, k-1) & \text{if } i \in N'_A, \\ (\alpha_j + 2, k) & \text{if } i \in N_A \backslash N'_A. \end{cases}$$

   If Rule 1 was applied, Y_j would be $(\alpha_j + 1, k+3)$. However, if this occurred, j would not EFX-envy i in either case (since j strongly prefers B) and so $|Y_i| < |Y_j| - 1$. This implies that $|Y_i| < |Y_{j'}|$ for all $j' \in N_B$, meaning that all $i' \in N_A$ do not envy any agents in N_B using the same proof as Lemma 5.6.

3. For the third condition, we can use the same argument as Section 5.3.1.

This completes our proof of Theorem 5.1.

6 Algorithm for Checking Existence of EF Allocations

For negative additive valuations, checking whether an envy-free allocation exists is NP-complete [7]. Under our scenario of two chore types, we propose a polynomial-time algorithm to solve the problem. In particular, we prove the following result.

Theorem 6.1. *When there are two chore types, an envy-free allocation can be found in polynomial time (with respect to the number of agents and items) whenever one exists.*

Before proving this theorem in full, let us first deal with a trivial case. If $v_i^A = 0$ and $v_j^B = 0$ for some (not necessarily distinct) agents i and j, then we can allocate all chores of types A and B to agents i and j respectively. Since all other agents are not given any chores, the resulting allocation is trivially envy-free. It suffices therefore to only consider cases where at most one chore type is valued at zero by at least one agent.

To further simplify the problem, we also do the following: if $v_i^B = 0$ for some agent i, swap the chore types—that is, rename them—so that $v_i^A = 0$ instead. Then, without loss of generality, we may assume $v_i^A \leq 0$ and $v_i^B < 0$. To prove Theorem 6.1, we first present a result about the structure of any envy-free allocation.

Lemma 6.2. *Consider an envy-free allocation where $v_i^A \leq 0$ and $v_i^B < 0$ for all $i \in N$. Let i and j be two agents with bundles $X_i = (\alpha_i, \beta_i)$ and $X_j = (\alpha_j, \beta_j)$. If $\frac{v_i^A}{v_i^B} < \frac{v_j^A}{v_j^B}$, then $\alpha_i \geq \alpha_j$ must hold.*

Proof. Assume by contradiction that $\frac{v_i^A}{v_i^B} < \frac{v_j^A}{v_j^B}$ but $\alpha_i < \alpha_j$. Agent i does not envy agent j, so $v_i(X_i) \geq v_i(X_j)$, i.e.,

$$\alpha_i v_i^A + \beta_i v_i^B \geq \alpha_j v_i^A + \beta_j v_i^B.$$

Rearranging,

$$(\alpha_i - \alpha_j) v_i^A \geq (\beta_j - \beta_i) v_i^B.$$

Similarly, since agent j does not envy agent i, we have

$$(\alpha_j - \alpha_i) v_j^A \geq (\beta_i - \beta_j) v_j^B.$$

Since v_i^B, v_j^B, and $\alpha_i - \alpha_j$ are strictly negative, we have

$$\frac{v_i^A}{v_i^B} \geq \frac{\beta_j - \beta_i}{\alpha_i - \alpha_j} \qquad \text{and} \qquad \frac{v_j^A}{v_j^B} \leq \frac{\beta_i - \beta_j}{\alpha_j - \alpha_i} = \frac{\beta_j - \beta_i}{\alpha_i - \alpha_j}.$$

By assumption,

$$\frac{\beta_j - \beta_i}{\alpha_i - \alpha_j} \leq \frac{v_i^A}{v_i^B} < \frac{v_j^A}{v_j^B} \leq \frac{\beta_j - \beta_i}{\alpha_i - \alpha_j},$$

which is absurd. □

Corollary 6.3. *Consider an envy-free allocation where, for all $i \in N$, we have $v_i^A \leq 0$ and $v_i^B < 0$, and each agent i is allocated the bundle $X_i = (\alpha_i, \beta_i)$. We can reorder the agents so that $\frac{v_j^A}{v_j^B} \leq \frac{v_{j+1}^A}{v_{j+1}^B}$ and $\alpha_j \geq \alpha_{j+1}$ for all $1 \leq j < n$.*

For the remainder of this section, we assume the agents are reordered as in Corollary 6.3. The following result provides an easy method for checking envy-freeness.

Lemma 6.4. *Consider an instance where $v_i^A \leq 0$ and $v_i^B < 0$ for all $i \in N$, and $\frac{v_i^A}{v_i^B} \leq \frac{v_{i+1}^A}{v_{i+1}^B}$ for all $1 \leq i < n$. Suppose each agent i receives the bundle $X_i = (\alpha_i, \beta_i)$, and $\alpha_i \geq \alpha_{i+1}$ for all $1 \leq i < n$. If agents i and $i+1$ do not envy each other for all $1 \leq i < n$, then the allocation is envy-free.*

Proof. Let i, j, k be three agents where $i < j < k$. It is sufficient to prove that non-envy between these agents is transitive: that is, whenever agents i and j do not envy each other, and agents j and k do not envy each other, then i and k do not envy each other. Let us show this is the case.

By assumption, $\alpha_i \geq \alpha_j \geq \alpha_k$. Since agents i and j do not envy each other, this implies $\beta_i \leq \beta_j$. Similarly, since agents j and k do not envy each other, $\beta_j \leq \beta_k$.

Let us first consider the case where at least one of i, j and k is indifferent towards type A chores. By the ordering assumption, this forces $v_i^A = 0$. Then, $v_i(X_i) = v_i^B \beta_i \geq v_i^B \beta_k = v_i(X_k)$. Hence agent i never envies agent k in this case. It remains to show agent k does not envy agent i: to do this, let us consider three subcases.

1. If $v_i^A = v_j^A = v_k^A = 0$, then $\beta_i = \beta_j = \beta_k$. Then $v_k(X_k) = v_k^B \beta_k = v_k^B \beta_i = v_k^B(X_i)$, so agent k does not envy agent i.
2. Suppose $v_i^A = v_j^A = 0$, but $v_k^A < 0$. This implies $\beta_i = \beta_j$. Then $v_k(X_k) \geq v_k(X_j) = v_k((\alpha_j, \beta_j)) \geq v_k((\alpha_i, \beta_i)) = v_k(X_i)$. Hence agent k does not envy agent i.
3. The last subcase, where $v_i^A = 0$ but v_j^A and v_k^A are strictly negative, is deferred: this is handled by the method detailed below.

Therefore, when at least one of agents i, j, k is indifferent towards type A chores, agents i and k do not envy each other. To complete the proof, let us now consider the remaining case where v_i^A, v_j^A, v_k^A are strictly negative.

Since agent j does not envy agent k, we have

$$\alpha_j v_j^A + \beta_j v_j^B \geq \alpha_k v_j^A + \beta_k v_j^B.$$

Rearranging,

$$(\alpha_j - \alpha_k) v_j^A \geq (\beta_k - \beta_j) v_j^B. \tag{2}$$

Since $\beta_j \leq \beta_k$, both sides of the inequality are non-positive. Also, since $\frac{v_i^A}{v_i^B} \leq \frac{v_j^A}{v_j^B}$, we have $\frac{v_i^A}{v_j^A} \leq \frac{v_i^B}{v_j^B}$. Multiplying the left side of inequality (2) by $\frac{v_i^A}{v_j^A}$ and the right side by $\frac{v_i^B}{v_j^B}$ yields

$$(\alpha_j - \alpha_k) v_i^A \geq (\beta_k - \beta_j) v_i^B,$$

so $v_i(X_j) \geq v_i(X_k)$. By assumption, agent i does not envy agent j, i.e., $v_i(X_i) \geq v_i(X_j)$; hence agent i does not envy agent k either.

We now use a similar approach to show that agent k does not envy agent i; this method also deals with case 3 from earlier, since it applies even if $v_i^A = 0$. Because agent j does not envy agent i, we have

$$(\alpha_j - \alpha_i) v_j^A \geq (\beta_i - \beta_j) v_j^B.$$

In this case, both sides of the inequality are non-negative. Since $\frac{v_k^A}{v_j^A} \geq \frac{v_k^B}{v_j^B}$, it follows that

$$(\alpha_j - \alpha_i) v_k^A \geq (\beta_i - \beta_j) v_k^B,$$

Algorithm 3: Dynamic programming function to determine whether a partial allocation can be extended into an envy-free allocation.

```
Function f(a, b, i, α, β):
    if i == N then                          ▷ Base case: there are no agents remaining
        if a + b == 0 then
            return YES
        else
            return NO
    ▷ Try every possible bundle (α′, β′) for agent i such that α′ ≤ α
    for α′ ← 0 to min(a, α) do
        for β′ ← 0 to b do
            if If (α′, β′) can be assigned to agent i + 1 such that agent i and i + 1 are
              envy-free then
                if f(a − α′, b − β′, i + 1, α′, β′) is YES then
                    return YES
    return NO
```

so $v_k(X_j) \geq v_k(X_i)$. By assumption, $v_k(X_k) \geq v_k(X_j)$, so agent k does not envy agent i either. □

We use Lemma 6.4 to create a dynamic programming algorithm, Algorithm 3, to help us determine whether an envy-free allocation exists. Let $f(a, b, i, \alpha, \beta)$ be the result of a subproblem that represents a state where we have assigned bundles to the first i agents. In particular, the state (a, b, i, α, β) represents the following:

- Items have been assigned to the first i agents such that they are envy-free,
- Agent i received the bundle (α, β), and
- There are a type A items and b type B items to allocate to the remaining $n - i$ agents.

The result of $f(a, b, i, \alpha, \beta)$ is YES if the remaining items can be allocated in an envy-free way to agents $i + 1$ through n, and NO otherwise.

To compute $f(a, b, i, \alpha, \beta)$, every valid bundle for agent $i + 1$ is considered. In particular, we consider every bundle (α', β') satisfying $\alpha' \leq a$, $\beta' \leq b$ and $\alpha' \leq \alpha$. If there exists such a bundle (α', β') that can be extended into an envy-free allocation, then the result is YES. Otherwise, the result is NO. Envy-freeness is checked using Lemma 6.4. Correctness of Algorithm 3 holds because it considers every assignment satisfying the structure of Corollary 6.3.

Since there are polynomial many states and each state takes polynomial time to compute (with respect to the number of agents and items), the dynamic programming algorithm runs in polynomial time.

We use Algorithm 3 to find an envy-free allocation whenever one exists. In particular, for each agent starting from agent 1, we try every possible bundle until one is found that can be extended into an envy-free allocation. If this procedure succeeds, then we have found an envy-free allocation in polynomial time. If this procedure fails, then by Corollary 6.3 and Lemma 6.4 we know that there does not exist any envy-free allocation.

Remark. We note that the same approach can be used to prove the equivalent result of Theorem 6.1 for two *good* types.

7 Discussion

The existence of EF1 and PO allocations or EFX allocations for the case of chores are major open problems in fair division. In this paper, we identified a natural setting or valuation restriction under which not only can we guarantee the existence of allocations that satisfy EF1 and PO, and EFX respectively, but such allocations can be computed in polynomial time. A related question is the complexity of checking whether there exists an envy-free allocation. Whereas this problem is NP-complete for chores in general, we showed that there exists a dynamic program for two chore types instances that can solve the problem in polynomial time. There are several relevant problems that remain open. The existence and complexity of EF1 and PO allocations or EFX allocations is open for personalized bi-valued utilities. It is also open whether there always exists a PO and EFX allocation for our setting.

Acknowledgment

Mashbat Suzuki was partially supported by the ARC Laureate Project FL200100204 on ”Trustworthy AI”.

References

[1] H. Aziz, P. Biro, J. Lang, J. Lesca, and J. Monnot. 2019. Efficient Reallocation under Additive and Ordinal Preferences. *Theoretical Computer Science* (2019).

[2] H. Aziz, I. Caragiannis, A. Igarashi, and T. Walsh. 2022. Fair allocation of indivisible goods and chores. *Journal of Autonomous Agents and Multi-Agent Systems* (2022).

[3] H. Aziz, S. Gaspers, S. Mackenzie, and T. Walsh. 2014. Fair Assignment Of Indivisible Objects Under Ordinal Preferences. In *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. 1305–1312.

[4] H. Aziz, H. Moulin, and F. Sandomirskiy. 2020. A polynomial-time algorithm for computing a Pareto optimal and almost proportional allocation. *Operations Research Letters* 48, 5 (2020), 573–578.

[5] H. Aziz and S. Rey. 2020. Almost Group Envy-free Allocation of Indivisible Goods and Chores. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI)*. 39–45.

[6] S. Barman, S. K. Krishnamurthy, and R. Vaish. 2018. Greedy algorithms for maximizing Nash social welfare. In *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.

[7] Umang Bhaskar, AR Sricharan, and Rohit Vaish. 2021. On Approximate Envy-Freeness for Indivisible Chores and Mixed Resources. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

[8] S. Bouveret, Y. Chevaleyre, and J. Lang. 2016. Fair Allocation of Indivisible Goods. In *Handbook of Computational Social Choice*, F. Brandt, V. Conitzer, U. Endriss, J. Lang, and A. D. Procaccia (Eds.). Cambridge University Press, Chapter 12, 284–311.

[9] E. Budish. 2011. The Combinatorial Assignment Problem: Approximate Competitive Equilibrium from Equal Incomes. *Journal of Political Economy* 119, 6 (2011), 1061–1103.

[10] I. Caragiannis, D. Kurokawa, H. Moulin, A. D. Procaccia, N. Shah, and J. Wang. 2019. The Unreasonable Fairness of Maximum Nash Welfare. *ACM Transactions on Economics and Computation (TEAC)* 7, 3 (2019).

[11] B. R. Chaudhury, J. Garg, and K. Mehlhorn. 2020. EFX Exists for Three Agents. (2020). Working paper.

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

[13] S. Ebadian, D. Peters, and N. Shah. 2022. How to Fairly Allocate Easy and Difficult Chores. In *21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2022, Auckland, New Zealand, May 9-13, 2022*. 372–380.

[14] J. Garg, A. Murhekar, and J. Qin. 2022. Fair and Efficient Allocations of Chores under Bivalued Preferences. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*. AAAI Press, 5043–5050.

[15] Pranay Gorantla, Kunal Marwaha, and Santhoshini Velusamy. 2022. Fair allocation of a multiset of indivisible items. *CoRR* abs/2202.05186 (2022).

[16] H. Hosseini, S. Sikdar, R. Vaish, and L. Xia. 2022. Fairly Dividing Mixtures of Goods and Chores under Lexicographic Preferences. *CoRR* abs/2203.07279 (2022). arXiv:2203.07279

[17] B. Li, Y. Li, and X. Wu. 2021. *Almost Proportional Allocations for Indivisible Chores*. Technical Report arXiv:2103.11849. arXiv.org.

[18] R. Mahara. 2020. Existence of EFX for Two Additive Valuations. *CoRR* abs/2008.08798 (2020). arXiv:2008.08798 https://arxiv.org/abs/2008.08798

[19] R. Mahara. 2021. Extension of Additive Valuations to General Valuations on the Existence of EFX. In *29th Annual European Symposium on Algorithms, ESA 2021, September 6-8, 2021, Lisbon, Portugal (Virtual Conference) (LIPIcs, Vol. 204)*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 66:1–66:15.

[20] H. Moulin. 2019. Fair Division in the Internet Age. *Annual Review of Economics* 11 (2019), 1–37.

[21] A. D. Procaccia. 2020. An answer to fair division's most enigmatic question: technical perspective. *Commun. ACM* 63, 4 (2020), 118.

[22] S. Zhou and X. Wu. 2021. Approximately EFX Allocations for Indivisible Chores. *CoRR* abs/2109.07313 (2021).

A EFX: Failure of Existing Approaches

In this section, we explore important algorithms for chore allocation as well natural adaptations for fair allocation of goods to the case of chores. Our finding is that the algorithms do not give the EFX guarantee even for the case of two item types, which suggests that a different approach is required to find an EFX allocation.

A.1 Goods algorithm of Gorantla et al. [15]

In the paper [15], an EFX algorithm is presented for goods with two item types. In their algorithm, they begin by allocating each agent their most preferred item in a round-robin way. This process stops once there are not enough items remaining to continue this. They then describe how to allocate the remaining items.

We show that there exists a case with chores where this approach cannot produce an EFX allocation. The case has 4 agents, numbered from 1 to 4, and 6 items (3 of type A and 3 of type B). The valuations are as follows, where ϵ is a sufficiently small positive constant. Note that $N_A = \{1\}$ and $N_B = \{2, 3, 4\}$.

Agents	**Valuation of type A items**	**Valuation of type B items**
1	$-\frac{1}{6} + \epsilon$	$-\frac{1}{6} - \epsilon$
2, 3, 4	$-\frac{1}{6} - \epsilon$	$-\frac{1}{6} + \epsilon$

Table 2: Two chore type instance where the goods algorithm of Gorantla et al. [15] fails to find an EFX allocation.

If we allocate each agent their most preferred item in a round robin way, this creates a partial allocation where $X_1 = (1, 0)$ and $X_2 = X_3 = X_4 = (0, 1)$, with 2 unallocated type A items. We cannot allocate both of these to one agent, as this would not be EFX. Hence, at least one agent in N_B must receive one of the unallocated chores, and at least one agent from N_B must not receive one of the unallocated chores. However, this is not EFX as an agent in N_B with $X_i = (1, 1)$ would EFX-envy an agent with $X_j = (0, 1)$.

A.2 PROPX Algorithms of Li et al. [17]

Li et al. [17] provide two algorithms which produce a PROPX allocation. They begin by transforming any instance into an instance with identical ordering. An instance has identical ordering (IDO) if all agents agree on the ordering of the items. In particular, let $c_1, c_2, ..., c_m$ be the chores. Then, $v_i(c_1) \leq v_i(c_2) \leq ... \leq v_i(c_m)$ for all agents i. They then use one of two algorithms, "The Top-trading Envy Cycle Elimination Algorithm" and "The Bid-and-Take Algorithm", to create a PROPX allocation for the IDO instance. They then provide a mechanism to transform this into a PROPX allocation for the original non-IDO instance. We show that there exists a case with two item types where both algorithms create an allocation that is not EFX.

The case has 3 agents, numbered from 1 to 3, and 6 items (3 of type A and 3 of type B).

The valuations are in Table 3, where ϵ is a sufficiently small positive constant. Note that $N_A = \{1\}$ and $N_B = \{2, 3\}$.

Agent	Valuation of type A items	Valuation of type B items
1	-3ϵ	$-\frac{1}{3} + 3\epsilon$
2	$-\frac{1}{3} + 2\epsilon$	-2ϵ
3	$-\frac{1}{3} + \epsilon$	$-\epsilon$

Table 3: Two chore type instance where PROPX algorithms fail to find an EFX allocation.

This instance is transformed into an instance with identical ordering, as shown in Table 4.

Agent	Valuation of type A items	Valuation of type B items
1	$-\frac{1}{3} + 3\epsilon$	-3ϵ
2	$-\frac{1}{3} + 2\epsilon$	-2ϵ
3	$-\frac{1}{3} + \epsilon$	$-\epsilon$

Table 4: The instance from Table 3, transformed into an instance with identical ordering.

The Top-trading Envy Cycle Elimination Algorithm: In this algorithm, items are allocated from the least valuation to the greatest valuation (according to the IDO instance) to an agent who does not envy any other agent. This leads to an allocation where each agent receives one type A and one type B item.

This allocation is then transformed into a PROPX allocation for the non-IDO case. This leads to one of the following two allocations, depending on the tiebreaking used:

- $X_1 = (2, 0)$, $X_2 = (1, 1)$ and $X_3 = (0, 2)$.
- $X_1 = (2, 0)$, $X_2 = (0, 2)$ and $X_3 = (1, 1)$.

Neither allocation is EFX. In the first case, this is due to the envy that agent 2 has for agent 3, and in the second case this is due to the envy that agent 3 has for agent 2.

The Bid-and-Take Algorithm: In this algorithm, items are allocated from the least valuation to the greatest valuation (according to the IDO instance) to an agent which has the greatest valuation for this item, as long as this satisfies PROPX. This leads to the following allocation:

- $X_1 = (2, 0)$, $X_2 = (1, 1)$ and $X_3 = (0, 2)$.

This allocation is then transformed into a PROPX allocation for the non-IDO case, which leaves the allocation unchanged. This is not EFX due to the envy that agent 2 has for agent 3.