PROPX Fair and Efficient Allocation of Indivisible Chores

Haris Aziz
UNSW
Sydney, Australia
haris.aziz@unsw.edu.au

Bo Li
The Hong Kong Polytechnic University
Hong Kong, China
comp-bo.li@polyu.edu.hk

Herve Moulin
University of Glasgow
Glasgow, UK
herve.moulin@glasgow.ac.uk

Xiaowei Wu
University of Macau
Macau, China
xiaoweiwu@um.edu.mo

Xinran Zhu
UNSW
Sydney, Australia
zxrzita@gmail.com

ABSTRACT

We consider the problem of fair allocation of indivisible items under negative additive valuations, where fairness is measured by the proportionality up to any item (PROPX). The compatibility of PROPX with economic efficiency concepts such as Pareto optimality was raised as a research problem by Moulin (2019), but it is still poorly understood. In this work, we investigate the conditions under which allocations that are both efficient and PROPX exist and provide new insights and algorithmic results.

KEYWORDS

Fair Division, PROPX, Pareto Optimal

ACM Reference Format:
Haris Aziz, Bo Li, Herve Moulin, Xiaowei Wu, and Xinran Zhu. 2018. PROPX Fair and Efficient Allocation of Indivisible Chores. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX).* ACM, New York, NY, USA, 8 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Fair division is a fundamental problem in multi-agent systems that involves the allocation of a set of items (goods or chores) among self-interested agents [28]. The formal study of fair allocation is initiated by Steinhaus [31] under the context of the proportional allocation of land. An allocation is regarded as proportionally fair (PROP) if every agent's utility is at least $1/n$ fraction of the utility when all items are allocated to her, where n is the number of agents. A stronger fairness concept is envy-freeness (EF) [17] which requires every agent prefers her own allocated items to the items allocated to any other agent. When items are indivisible, proportionality and envy-freeness are hard to achieve and recent works have been dedicated to studying the extent to which their relaxed versions can be satisfied. Among various ways to relax the fairness requirements, "up to one item" (denoted by PROP1 and EF1) and "up to any item" (denoted by PROPX and EFX) are widely accepted, which require the fairness requirement to be satisfiable if *some* or *any* item is removed. For both goods and chores, PROP1 and EF1 are easy to satisfy [7, 15, 27], but the existence of EFX allocations is still unknown. It is shown by Aziz *et al.* [7] that a PROPX allocation may not exist for goods. In contrast, for chores, a PROPX allocation always exists even when the agents are asymmetric [26, 29].

Whether or not a certain fairness property is compatible with economic efficiency is of critical importance to its merit. The most common formulation of efficiency is the *Pareto optimality* (PO), which requires that no alternative allocation can make an agent better off without making anyone worse off. A stronger version of PO is the fractional Pareto optimality (fPO) where the requirement is tightened to "no alternative fractional allocation". It is known that for goods, both PROP1 and EF1 can be satisfied together with PO [14, 15]. For chores, PROP1 and fPO are still compatible as shown by Aziz *et al.* [7], however, the compatibility between PROPX and PO on the one hand, and between EF1 and PO on the other hand, are both poorly understood. The former appears as an open question in the survey paper of Moulin [29]:

> *Can we always divide the bads efficiently and meet FSX (an alias for PROPX)?*

Very recently, there are some progresses on the compatibility of EF1 and PO. It is shown by Ebadian *et al.* [16] and Garg *et al.* [19] respectively that when agents have bi-valued valuations, an EF1 and PO allocation exists and can be found in polynomial time. We quote the following argument from [19] regarding the allocation of chores:

> *Settling the existence of EF1+PO allocations (and developing algorithms for computing them) has turned out to be a challenging open problem.*

In contrast to the existence of EF1+PO allocation under bi-valued valuations, nothing is known regarding the compatibility of PROPX and PO prior to our work. In summary, we aim at understanding the compatibility of PROPX with PO or its weaker or stronger versions. We allow the agents to have asymmetric weights that indicate the proportion of the burdens put on them.

1.1 Our Contributions

Similar to EFX [14, 30], the definition of PROPX has two versions. The weaker version requires the removed item brings non-zero utility to the agent, and the stronger version does not have this

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

requirement. To distinguish the two definitions, we continue to denote the weaker version by PROPX but the stronger version by PROPX*. We first show that PROPX* and PO are not compatible for any number of agents with additive valuations. Then we focus on the problem of PROPX and PO.

Secondly, we show that existing algorithms for computing PROPX allocations do not additionally satisfy PO. Moreover, a Pareto improvement over a PROPX allocation may not be PROPX any more. In fact, if we start from an arbitrary PROPX allocation, it can be the case that neither the allocation is PO nor any of its Pareto improvements is PROPX. However, we identify some conditions under which the Pareto improvements preserve the property of PROPX.

Thirdly, we show that PROPX and fPO are incompatible even for two agents with the same weight. This result contrasts with the one by Aziz *et al.* [7] who showed that fPO is compatible with PROP1. On the positive side, we show that for any number of agents who may have asymmetric weights, PROPX is compatible with SD-efficiency. SD stands for *stochastic dominance*, a partial ordering relation of subsets of chores that only depends upon the ordinal preferences of a given agent. SD-efficiency is then weaker than PO that depends upon the agents' cardinal utilities. Furthermore, an allocation that is both PROPX and SD-efficient can be computed in polynomial time.

Finally, we prove that in the following cases a PROPX and PO allocation exists: (1) two agents with any additive valuations and symmetric weights, (2) any number of agents with lexicographic or bi-valued valuations, even if agents may have asymmetric weights. The instances when agents have lexicographic valuations have been widely considered in the literature [5, 11, 21, 22]. The case when agents have bi-valued utility functions [1] is also one of the most well studied restricted cases of the fair division problem. Our result aligns with [16, 19] which show the compatibility of EF1 and PO under bi-valued valuations. As justified therein, the study of bi-valued valuations is of practical interest, since reporting exact numerical utilities can be cumbersome for agents in many real-word scenarios.

1.2 Related Work

Fair and efficient allocation of indivisible items has attracted significant effort from both fields of computer science and economics. For some recent overviews, we refer to the surveys of Moulin [29], Aziz [8] and Walsh [32]. When the items are goods, it is shown by Caragiannis *et al.* [14] that the allocation that maximizes Nash social welfare is EF1 and PO. However, the computation of Nash social welfare maximizers may need exponential time. Later, Barman *et al.* [10] showed that an EF1 and PO allocation can be computed in pseudo-polynomial time. Conitzer *et al.* [15] proved the existence of a PROP1 and PO allocation, and Barman and Krishnamurthy [9] designed the first polynomial-time algorithm to compute such an allocation. When the item are chores, Brânzei and Sandomirskiy [12] showed that a PROP1 and PO allocation exists even when the agents have asymmetric weights and such an allocation can be computed in polynomial time if the number of agents or items is constant. Recently, Aziz *et al.* [7] designed a strongly polynomial-time algorithm to compute a PROP1 and PO allocation. Moreover, their results holds even if the agents have asymmetric weights and the items are mixture of goods and chores.

The works most closely related to ours are that of Moulin [29] and Li *et al.* [26]. Although it is shown by Aziz *et al.* [7] that a PROPX allocation may not exist for goods, Moulin [29] proved the existence of PROPX allocations for chores under the alias of FSX (fair share up to all items). Li *et al.* [26] also studied the existence and computation of PROPX allocations for chores while allowing for asymmetric weights. They present several results including two algorithms for PROPX. Unlike us, they do not additionally consider efficiency concepts such as Pareto optimality. Although the algorithm of Aziz *et al.* [7] ensures PROP1 and PO, it does not achieve the stronger property PROPX.

Besides the "up to one item" and "up to any item" relaxations, maximin share (MMS) fairness is another widely studied relaxation of proportionality, which is proposed by [13]. It is shown by Kurokawa *et al.* [25] and Aziz *et al.* [4] respectively that an MMS allocation may not exist for goods and chores. Thereafter, many approximation algorithms have been proposed (see, e.g., [4, 18, 23, 25]). Note that given any approximate MMS allocation, a Pareto improvement preserves the approximation ratio of MMS, and thus the compatibility of approximate MMS and PO is settled. However, whether there exist polynomial time algorithms to compute such allocations remains largely open.

2 PRELIMINARIES

We consider the allocation of m indivisible chores in set O to n (asymmetric) agents in set N, where each agent $i \in N$ has a weight $b_i > 0$ and $\sum_{i \in N} b_i = 1$. If $b_1 = \cdots = b_n = \frac{1}{n}$, the agents are called symmetric. Each agent's utility function is denoted by $u_i : 2^O \to \mathbb{R}^- \cup \{0\}$. Let $u_i(o) = u_i(\{o\})$ be agent i's utility for receiving the whole item o. In this work, we assume the utility functions are additive, i.e., for any $S \subseteq O$, $u_i(S) = \sum_{o \in S} u_i(o)$. In case we meet fractional allocations, let $u_i(x_i) = \sum_{o \in O} u_i(o) x_{i,o}$.

Each item is allocated to a single agent, and we call the allocation *integral* or simply an allocation. We typically denote an integral allocation by $X = (X_1, \cdots, X_n)$ where X_i denotes the allocated set of items to agent i. If some item is allocated to more than one agent, the allocation is called *fractional* and is denoted by $x = (x_1, \ldots, x_n)$ where $x_i = (x_{i,1}, \cdots, x_{i,m})$ is the allocation of agent i and $0 \leq x_{i,o} \leq 1$ is the fraction of item o given to agent i. Note that it is required that $\sum_{i \in N} x_{i,o} = 1$ for all item $o \in O$. A fractional allocation y Pareto improves a fractional allocation x if $u_i(y_i) \geq u_i(x_i)$ for all $i \in N$ and for some i the inequality is strict. We will call an allocation *Pareto optimal* (PO), if no integral allocation Pareto improves it. An allocation that cannot be Pareto improved by any fractional allocation is called *fractional Pareto optimal* (fPO). Clearly, an fPO allocation is PO as well.

2.1 Fairness Concepts

Next, we present our main fairness concepts. An allocation X is *proportional* (PROP) if for each agent $i \in N$, $u_i(X_i) \geq u_i(O) \cdot b_i$. It is not hard to see that a PROP allocation may not exist even when there are two agents and a single item. Thus we consider the following relaxations.

Definition 2.1 (PROPX and PROP1). Given a chore allocation instance, an allocation $X = (X_1, \cdots, X_n)$ is *proportional up to any item* (PROPX) if for each agent $i \in N$,

$$\forall o \in X_i \text{ s.t., } u_i(o) < 0, \quad u_i(X_i \setminus \{o\}) \geq u_i(O) \cdot b_i. \tag{1}$$

The allocation X is *proportional up to one item* (PROP1) if for each agent $i \in N$,

$$\exists o \in X_i, \quad u_i(X_i \setminus \{o\}) \geq u_i(O) \cdot b_i.$$

Similar to EFX [14, 30], the definition of PROPX has two versions and we have the weaker version in Equation (1) which requires that the removed item makes non-zero utility. A stronger version, denoted by PROPX*, is to drop the requirement of non-zero utility, i.e.,

$$\forall o \in X_i, \quad u_i(X_i \setminus \{o\}) \geq u_i(O) \cdot b_i.$$

Next we show that PROPX* and PO are not compatible.

Proposition 2.2. *PROPX* and PO are not compatible for any number of agents with additive valuations.*

Proof. Consider the following instance with $n \geq 2$ agents and $m = n + 1$ items. Let $0 < \epsilon < \frac{1}{n^2}$.

	o_1	o_2	$\dots$	o_n	o_{n+1}
1	$\mathbf{0}$	$-\epsilon$	$\dots$	$-\epsilon$	$(n-1)\epsilon - 1$
2	$-\epsilon$	$\mathbf{0}$	$\dots$	$-\epsilon$	$(n-1)\epsilon - 1$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
n	$-\epsilon$	$-\epsilon$	$\dots$	$\mathbf{0}$	$(n-1)\epsilon - 1$

Note that in this instance, any PO allocation must allocate item i to agent i, for all $i = 1, \cdots, n$. Otherwise reallocating item i from its owner to agent i does not hurt anyone but strictly increases the original owner's utility. However, in such allocations, the agent who receives item $n+1$ does not satisfy PROPX*, since by removing the item with utility 0, the remaining utility is $(n-1)\epsilon - 1 < \frac{1}{n}$. □

Thus in the following of this work, we only focus on the weaker definition of PROPX as defined in Equation (1). Moreover, we always assume $u_i(o) \neq 0$ for all $i \in N$ and $o \in O$. This is without loss of generality since if some agent has zero utility for an item, we can assign the item to this agent without incurring any additional load. Under this assumption, PROPX and PROPX* do not make any difference.

2.2 Relation between EFX and PROPX

Before investigating the compatibility of PROPX and PO, we explore the relations between PROPX and the EFX property that is based on a relaxation of envy-freeness. An allocation X is *envy-free (EF)* if for any agents $i, j \in N$, we have $u_i(X_i) \geq (b_i/b_j) \cdot u_i(X_j)$. An allocation X is EFX if for any agents $i, j \in N$,

$$\forall o \in X_i, \quad u_i(X_i \setminus \{o\}) \geq \frac{b_i}{b_j} \cdot u_i(X_j).$$

Next, we show that EFX implies PROPX even for asymmetric weights. The proof of the following proposition is included in the appendix. The statement contrasts with the fact that EFX does not imply PROPX for the case of goods (see, e.g., [7, 20]).

Proposition 2.3. *Any EFX allocation of chores is also PROPX even for agents with asymmetric weights.*

For $n = 2$, it is known that PROP1 and EF1 are equivalent [6]. Interestingly, for $n = 2$, PROPX and EFX are not equivalent.

Proposition 2.4. *A PROPX allocation may not be EFX even for two agents with equal weights.*

Proof. Consider the following instance with a PROPX allocation shown in squares, i.e., $X_1 = \{o_1, o_2\}$ and $X_2 = \{o_3\}$.

	o_1	o_2	o_3
1	$\boxed{-2}$	$\boxed{-4}$	-3
2	-5	-1	$\boxed{-2}$

However, by removing item o_1 from X_1, we have

$$u_1(X_1 \setminus \{o_1\}) = -4 < u_1(X_2) = -3,$$

which means this allocation is not EFX to agent 1. □

3 CHALLENGES

Before presenting our algorithmic results, we first discuss the challenges existing algorithm faces, and some failed attempts.

3.1 Lack of PO of Existing Algorithms

Li *et al.* [26] proved that a PROPX allocation always exists for chore allocation and can be computed in polynomial time even for agents with asymmetric weights. We present a self-contained specificaton of algorithm that captures the essential features of the Bid-and-Take algorithm of Li *et al.* [26] and the algorithm of Moulin [29]. The algorithms are identical because they adhere to the following guidelines: (1) items are allocated from lowest to highest utility; (2) each item is allocated to the agent that has the highest utility for it; (3) each agent is turned off when her utility becomes smaller than her share. We will show further results of these algorithms so it is useful to a formal specification in our notation. In particular, we show that the algorithm may not return a Pareto optimal allocation. For completeness, we prove the following proposition in the appendix.

Proposition 3.1. *For chores, Algorithm 1 terminates, and allocates all the items, and returns a PROPX allocation.*

We now show that Algorithm 1 does not guarantee PO even for three symmetric agents by providing a counter example.

Example 3.2 (Algorithm 1 does not guarantee PO). Consider an instance with three agents $N = \{1, 2, 3\}$ and four items $O = \{o_1, o_2, o_3, o_4\}$ at the beginning as per the algorithm. The utilities are shown in the following table.

	o_1	o_2	o_3	o_4
1	$\boxed{-0.2}$	$\boxed{-0.2}$	-0.15	-0.45
2	-0.3	-0.25	$\boxed{-0.3}$	-0.15
3	-0.3	-0.3	-0.35	$\boxed{-0.05}$

The allocation X returned by Algorithm 1 is shown in the squares. However, agents 1 and 2 can exchange items o_2 and o_3 to get an allocation X', where $u_1(X'_1) = u_1(\{o_1, o_3\}) = -0.35 > -0.4 = u_1(X_1)$ and $u_2(X'_2) = u_2(\{o_2\}) = -0.25 > -0.3 = u_2(X_2)$. Hence X' Pareto dominates X and X is not PO.

Algorithm 1: Computing PROPX allocations

Input: An instance $I = (N, O, u)$
Output: A PROPX allocation

Initialize $X_i \leftarrow \emptyset$ for all $i \in N$
Normalize utilities so that $u_i(O) = -1, \forall i \in N$.
while $O \neq \emptyset$ **do**
  For each item $o \in O$, let $m(o) \leftarrow \max_{i \in N} u_i(o)$.
  Pick some item $o \leftarrow \arg\min_{o' \in O} m(o')$.
  Let $i_o \in N$ be the agent with lowest index such that $u_{i_o}(o) = m(o)$.
  $X_i \leftarrow X_i \cup \{o\}$
  $O \leftarrow O \setminus \{o\}$
  if $u_i(X_i) < -b_i$ **then**
    $N \leftarrow N \setminus \{i\}$
return Allocation $X = (X_1, \cdots, X_n)$

Li *et al.* [26] also presented *Top-trading Envy Cycle Elimination* algorithm to find PROPX allocations for symmetric agents. However, this cannot ensure PO as well, and we provide a detailed discussion in the appendix.

3.2 The Failure of Pareto Improvement

To find an allocation that is both PROPX and PO, a naive approach would be that Pareto improvements can preserve PROPX. However, there are several challenges. First, checking whether a given allocation is PO is coNP-hard.

Proposition 3.3. *Under additive negative utilities, testing whether a given allocation is PO or not is weakly coNP-complete, even for $n = 2$ with identical weights.*

Proof. The proof is an adaptation of a similar result by Aziz *et al.* [5] for positive utilities. First, given an allocation $X = (X_1, \cdots, X_n)$, testing whether X is PO is in coNP since for any allocation X' one can test whether X is Pareto dominated by X' in linear time by comparing every agent's utility under the two allocations.

Next, we design a polynomial-time reduction from PARTITION problem, which is a well known NP-complete problem [24]. An instance of PARTITION is described by a set of t elements $E = \{e_1, \ldots, e_t\}$ where each $e_j \in E$ has integer weight $w(e_j)$ such that $\sum_{e_i \in E} w(e_i) = 2M$. The question is to decide whether there is a balanced partition of E i.e., $S \subseteq E$ such that $\sum_{e_i \in S} w(e_i) = \sum_{e_i \in E \setminus S} w(e_i) = M$. Given any PARTITION instance we construct a fair allocation instance with $t + 1$ items $\{o^+, o_1, \ldots, o_t\}$ and two agents $\{1, 2\}$. Agent 1's utility function is: $u_1(o^+) = -M$ and $u_1(o_i) = -w(e_i)$ for all $i \in \{1, \ldots t\}$. Agent 2' utility function is: $u_2(o^+) = -M + \varepsilon$, with $0 < \varepsilon < 1$, and $u_2(o_i) = -w(e_i)$ for all $i \in \{1, \ldots, t\}$. Consider allocation X with $X_1 = \{o^+\}$ and $X_2 = O \setminus \{o^+\}$. Then X is PO if and only if there is a balanced partition of E. □

The second difficulty is that given an arbitrary PROPX allocation X which is not PO, any Pareto improvement of X makes X not PROPX, as shown by the following example.

Example 3.4. Consider an instance with four items and two agents with utilities shown in the following table.

	o_1	o_2	o_3	o_4
1	−0.32	**−0.27**	**−0.11**	−0.3
2	**−0.13**	−0.05	−0.38	**−0.44**

The allocation X shown in squares, i.e., $X_1 = \{o_2, o_3\}$ and $X_2 = \{o_1, o_4\}$, is PROPX but is not PO because it is Pareto dominated by allocation X' as shown in the following table, i.e., $X'_1 = \{o_4\}$ and $X'_2 = \{o_1, o_2, o_3\}$. Allocation X' is not PROPX since $u_2(X'_2 \setminus \{o_2\}) = -0.51 < -0.5$.

	o_1	o_2	o_3	o_4
1	−0.32	−0.27	−0.11	**−0.3**
2	**−0.13**	**−0.05**	**−0.38**	−0.44

Actually, X' is the only allocation that Pareto dominates X. Suppose any allocation Y that Pareto dominates X. Agent 1's utility can be increased by either giving one of her item to agent 2 without receiving any other item, or giving out both o_2 and o_3 in exchange for either o_1 or o_4 with a total higher utility. The first case is not acceptable since it strictly decreases agent 2's utility. For the second case, if agent 1 gets o_1, then o_2, o_3 and o_4 are assigned to agent 2 which decreases her utility since

$$u_2(Y_2) = u_2(\{o_2, o_3, o_4\}) = -0.87 < -0.57 = u_2(X_2).$$

Then the only option is that agent 1 gets o_4, with

$$u_1(Y_1) = u_1(\{o_4\}) = -0.3 > u_1(X_1), \text{ and}$$

$$u_2(Y_2) = u_2(\{o_1, o_2, o_3\}) = -0.56 > u_2(X_2).$$

Hence X' is the only allocation that Pareto dominates X.

Therefore, for the allocation X, there does not exist a Pareto improvement over it that preserves PROPX.

Next, we identify a condition under which Pareto improvements over a PROPX allocation preserves PROPX. We proved the following lemma in the appendix.

Lemma 3.5. *Consider a PROPX allocation X such that there is a Pareto improvement Y over X such that*

$$\forall i \in N, \quad \max_{o \in X_i} u_i(o) = \max_{o \in Y_i} u_i(o)$$

Then Y is PROPX.

4 PROPX AND VARIANTS OF PO

In this section, we consider fPO and SD-efficiency which are stronger and weaker versions of PO, respectively.

4.1 PROPX and fPO

It is shown by Aziz *et al.* [7] that an fPO and PROP1 allocation exists for chores. In contrast, we show that PROPX and fPO are not compatible any more.

Proposition 4.1. *There may not exist any PROPX and fPO allocation even for 2 agents.*

Proof. Consider an instance with 2 agents and 3 items. The utilities are shown in the following table.

	o_1	o_2	o_3
1	−0.1	−0.7	−0.2
2	−0.2	−0.7	−0.1

In the above instance, it is straightforward that any PROPX allocation cannot allocate all items to a single agent. By the characterization of fPO allocations for 2 agents [2], there are two fPO integral allocations, both of which allocate o_1 to agent 1 and o_3 to agent 2. However, neither of the two allocations is PROPX because the agent who receives item o_2 has utility -0.7 after removing the item with maximum utility. □

The proof of Proposition 4.1 also shows that no rounding of an fPO and PROP fractional allocation ensures PROPX.

4.2 PROPX and SD-efficiency

Next, we consider *SD-efficiency* which is an ordinal notion of efficiency and is weaker than PO. Given two allocations X and Y, we say that agent i *SD prefers* allocation X_i to allocation Y_i (denoted by $X_i \gtrsim_i^{SD} Y_i$) if for all $o \in O$,

$$|\{o' \in X_i \mid u_i(o') \leq u_i(o)\}| \leq |\{o' \in Y_i \mid u_i(o') \leq u_i(o)\}|.$$

Denote $X_i \succ_i^{SD} Y_i$ if $X_i \gtrsim_i^{SD} Y_i$ but not $Y_i \gtrsim_i^{SD} X_i$. An allocation X is SD-efficient is there is no other allocation Y such that $Y_i \gtrsim_i^{SD} X_i$ for all $i \in N$ and $Y_i \succ_i^{SD} X_i$ some i.

Given an allocation X, we can build a *trading graph* $G(X) = (V(X), E(X))$. In $G(X)$, the set of vertices $V(X)$ contains one vertex per item in O. Furthermore, for any two vertices o and o', there is a directed edge from o to o' if $u_i(o') \geq u_i(o)$, where i is the agent who receives item o in X. If $u_i(o') > u_i(o)$, the edge is called *strict*. We say that $G(X)$ admits a *Pareto trading cycle* C if there is a cycle C in $G(X)$ that contains at least one strict edge. We say that allocation Y is a result of resolving trading cycle C if for each edge $(o, o') \in C$ with $o \in X_i$, it holds that $Y_i = (X_i \setminus \{o\}) \cup \{o'\}$. Note that given an allocation, the construction of $G(X)$ and finding a cycle therein can be done in polynomial time.

Lemma 4.2. *For any allocation X, $G(X)$ contains at most $O(m^2)$ edges, and after resolving a trading cycle, the number of edges in $G(X)$ strictly decreases.*

Lemma 4.3. *An allocation X is not SD-efficient if and only if there exists a cycle in $G(X)$ which contains a strict edge.*

Lemma 4.4. *Resolving a Pareto trading cycle in a PROPX allocation with asymmetric weights preserves PROPX.*

We prove Lemmas 4.2, 4.3 and 4.4 in the appendix.

Proposition 4.5. *There exists a polynomial-time algorithm that computes an allocation that is both PROPX and SD-efficient even for agents with asymmetric weights.*

Proof. We first use any existing polynomial-time algorithm to compute a PROPX allocation for agents with asymmetric weights (see, e.g., [26]). By Lemma 4.3, we can check in linear time whether the allocation is SD-efficient. If not, by Lemma 4.3, the corresponding trading graph admits a Pareto trading cycle. By Lemma 4.2 and 4.4, resolving this cycle preserves PROPX and reduces the number of edges by one. Since there are at most $O(m^2)$ edges in the trading graph, the process terminates with an allocation that is both PROPX and SD-efficient in polynomial time. □

Remark. We have identified two conditions (Lemmas 3.5 and 4.4) that preserve PROPX under Pareto improvements. In the appendix, we show that it is not guaranteed that a PROPX and PO allocation can be reached by Pareto improving any given PROPX allocation as per Lemmas 3.5 and 4.4 even if such an allocation exists.

5 PROPX AND PO IN RESTRICTED CASES

In this section we show that a PROPX and PO allocation always exists for the case of two agents. We further identify some natural classes of utilities under which a PROPX and PO allocation can be computed in polynomial time.

5.1 Two Agents

For the case of 2 agents, we show that a PROPX and PO is guaranteed to exist, if they have symmetric weights. Actually we prove a stronger statement that there exists an EFX and PO allocation for two symmetric agents. Since PROPX is implied by EFX, the result follows.

Proposition 5.1. *For 2 symmetric agents, there exists an EFX and PO allocation.*

Proof. We normalize the utilities of the two agents $\{1, 2\}$ so that $u_1(O) = u_2(O) = -1$. Let (X_1, X_2) be the leximin allocation. In other words, among all allocations that maximize $\min\{u_1(X_1), u_2(X_2)\}$, (X_1, X_2) has the maximum value of $\max\{u_1(X_1), u_2(X_2)\}$. In the following we show that (X_1, X_2) is EFX and PO.

The Pareto optimality follows straightforwardly from the fact that the allocation that Parato improves (X_1, X_2) must have a higher lexicographical order, which contradicts with (X_1, X_2) being Leximin allocation. Next, we argue that the allocation is EFX. If the allocation is PROP, e.g., both agents $i \in \{1, 2\}$ have $u_i(X_i) \geq -0.5$, then it is clearly EFX because $u_i(X_j) = -1 - u_i(X_i) \leq -0.5$ for $j \neq i$. Now suppose that the allocation is not PROP. In other words, there is an agent, say agent 1, that has $u_1(X_1) < -0.5$. It follows that $u_1(X_2) = -1 - u_1(X_1) > -0.5$.

We claim that we have $u_2(X_2) > -0.5$. Because otherwise $u_2(X_1) \geq -0.5$, and thus swapping the two bundles X_1, X_2 gives a PROP allocation, which has $\min\{u_1(X_2), u_2(X_1)\} \geq -0.5$ and contradicts with (X_1, X_2) being Leximin. Therefore agent 2 does not envy agent 1, and the allocation is EFX to agent 2.

Next we prove that the allocation is also EFX to agent 1. That is, for any $o \in X_1$ such that $u_1(o) < 0$, we have $u_1(X_1 \setminus \{o\}) \geq u_1(X_2)$. Suppose otherwise, e.g., $u_1(X_1 \setminus \{o\}) < u_1(X_2)$. Then we have $u_1(X_1) < u_1(X_2 \cup \{o\})$. In other words, both $X_1 \setminus \{o\}$ and $X_2 \cup \{o\}$ offer a better utility to agent 1, compared to X_1. Now suppose we let agent 2 picks her preferred bundle between $X_1 \setminus \{o\}$ and $X_2 \cup \{o\}$, and assign the remaining one to agent 1. Then the utility of agent 2 is at least -0.5 while the utility of agent 1 is strictly larger than $u_1(X_1)$, which contradicts with (X_1, X_2) being the Leximin allocation. □

The above result immediately implies the following.

Corollary 5.2. *For 2 symmetric agents, there exists a PROPX and PO allocation.*

Note that when agents have asymmetric weights, the Leximin allocation (which is PO) might not be PROPX due to the following instance.

Example 5.3. Consider the following instance with 2 agents with asymmetric weights $b_1 = 0.7$ and $b_2 = 0.3$.

	o_1	o_2	o_3	o_4
1	$\boxed{-0.79}$	-0.18	$\boxed{-0.01}$	-0.02
2	-0.58	$\boxed{-0.39}$	-0.02	$\boxed{-0.01}$

The only allocation (which is the Leximin allocation) that maximizes $min\{u_1(X_1) + b_1, u_2(X_2) + b_2\}$ has $X_1 = \{o_1, o_3\}$ and $X_2 = \{o_2, o_4\}$. Unfortunately, this allocation is not PROPX.

Our result (Proposition 5.1) complements that of [30], who showed that for positive utilities and symmetric weights, the leximin is EFX and PO. Since the computation of Leximin allocation is NP-hard, our result only shows the existence of PROPX and PO allocations for two symmetric agents. A natural open question is whether there exist polynomial-time algorithms for the computation of such allocations.

One may also wonder whether the Leximin allocation is PROPX and PO for $n \geq 3$ agents. Unfortunately, via the following example we show that even for three symmetric agents, no Leximin allocation is PROPX and PO.

Example 5.4. Consider the following instance.

	o_1	o_2	o_3	o_4
1	$\boxed{-0.34}$	$\boxed{-0.05}$	-0.31	-0.3
2	-0.4	-0.4	$\boxed{-0.09}$	-0.11
3	-0.4	-0.4	-0.11	$\boxed{-0.09}$

To maximize the minimum utility, the Leximin allocation has to assign items o_1, o_2 to agent 1. Thus the unique Leximin allocation assigns the items as we indicated using the rectangles. However, this allocation is not PROPX since

$$u_1(X_1 \setminus \{o_2\}) = u_1(o_1) = -0.34 < -1/3.$$

5.2 Restricted Utility Functions

In this section we show that for the cases when agents have *lexicographic utilities*, or *bi-valued utilities*, PROPX and PO allocations always exist and can be computed efficiently.

Definition 5.5 (Lexicographic Utilities). We say that utility function u is *lexicographic* if there is a partition $(L_1, \ldots, L_k)$ of the items O such that

(1) $\forall i \in [k]$ and $o, o' \in L_i$, we have $u(o) = u(o')$, and
(2) $\forall i \in [k]$ and $o \in L_i$, we have $u(o) < u(\cup_{j=i+1}^{k} L_j)$.

Under lexicographic utilities, we say that an agent prefers item o' to item o if there exist $i < j$ such that $o \in L_i$ and $o' \in L_j$. In other words, the agent hates each chore more than all more preferred chores combined.

Definition 5.6 (Bi-valued Utilities). We say that utility function u is *bi-valued* if there exist constants $0 > \alpha > \beta$ such that $u(o) \in \{\alpha, \beta\}$ for all items $o \in O$.

For instances with lexicographic or bi-valued utilities, the following lemmas have been established for efficient checking of whether an allocation is PO or not.

Lemma 5.7 (Aziz *et al.* [5]; Ebadian *et al.* [16]). *An allocation X is not PO with respect to lexicographic preferences if and only if there exists a cycle in $G(X)$ which contains at least one edge corresponding to a strict preference.*

Lemma 5.8 (Ebadian *et al.* [16]). *An allocation X is not PO with respect to bi-valued preferences if and only if there exists a cycle in $G(X)$ which contains at least one edge corresponding to a strict preference.*

Utilizing these results, we show that there exists efficient algorithms for the computation of PROPX and PO allocation for instances when agents have lexicographic or bi-valued utilities, even if they have asymmetric weights.

Proposition 5.9. *For lexicographic utilities, there exists a polynomial-time algorithm that computes an allocation that is both PROPX and PO even for asymmetric weights.*

Proof. We first use any existing polynomial-time algorithm to compute a PROPX allocation for a group of agents with asymmetric weights (see, e.g., [26]). From Lemma 5.7, we can check in $O(nm)$ time whether the allocation in PO or not by constructing $G(X)$ and checking whether there is a cycle in $G(X)$ which contains at least one edge corresponding to a strict preference. If it is not, from Lemma 5.7, we know that the allocation's trading graph admits a Pareto trading cycle. If we resolve such a cycle, it follows from Lemma 4.4 that the new allocation is also PROPX. By Lemma 4.2, there can be at most nm such Pareto improvements until the process terminates with an allocation that is both PROPX and PO. □

The proof of the following proposition is identical to that of Proposition 5.9 except that we use Lemma 5.8 instead of Lemma 5.7, and hence is omitted.

Proposition 5.10. *For bi-valued utilities, there exists a polynomial-time algorithm that computes an allocation that is both PROPX and PO even for asymmetric weights.*

6 CONCLUSION AND OPEN PROBLEMS

In this work, we studied the compatibility of PROPX and Pareto optimality. As we have seen, in contrast to PROP1, the issue for PROPX poses considerably different challenges. Our study raises several questions. For example,

- Does there always exist a PROPX and PO allocation for general additive valuations?
- What is the complexity of computing an allocation that is both PROPX and PO?
- What is the complexity of computing an allocation that is PO among all PROPX allocations (i.e., not Pareto dominated by any PROPX allocation)?

REFERENCES

[1] Georgios Amanatidis, Georgios Birmpas, Aris Filos-Ratsikas, Alexandros Hollender, and Alexandros A. Voudouris. Maximum nash welfare and other stories about EFX. *Theoretical Computer Science*, 863:69–85, 2021.
[2] H. Aziz, Simina Brânzei, S. Frederiksen, and A. Filos-Ratsikas. The adjusted winner procedure: Characterizations and equilibria. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 454–460, 2015.
[3] H. Aziz, S. Gaspers, S. Mackenzie, and T. Walsh. Fair assignment of indivisible objects under ordinal preferences. *Artificial Intelligence*, 227:71–92, 2015.

[4] H. Aziz, G. Rauchecker, G. Schryen, and T. Walsh. Algorithms for max-min share fair allocation of indivisible chores. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI)*, pages 335–341, 2017.
[5] H. Aziz, P. Biro, J. Lang, J. Lesca, and J. Monnot. Efficient reallocation under additive and ordinal preferences. *Theoretical Computer Science*, 2019.
[6] H. Aziz, I. Caragiannis, A. Igarashi, and T. Walsh. Fair allocation of combinations of indivisible goods and chores. In *Proceedings of the 28h International Joint Conference on Artificial Intelligence (IJCAI)*, 2019.
[7] H. Aziz, H. Moulin, and F. Sandomirskiy. A polynomial-time algorithm for computing a pareto optimal and almost proportional allocation. *Operations Research Letters*, 48(5):573–578, 2020.
[8] H. Aziz. Developments in multi-agent fair allocation. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI)*, 2020.
[9] S. Barman and S. K. Krishnamurthy. On the proximity of markets with integral equilibria. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, pages 1748–1755, 2019.
[10] S. Barman, S. K. Krishnamurthy, and R. Vaish. Finding fair and efficient allocations. In *Proceedings of the 2018 ACM Conference on Economics and Computation, Ithaca, NY, USA, June 18-22, 2018*, pages 557–574, 2018.
[11] Dorothea Baumeister, Sylvain Bouveret, Jérôme Lang, Nhan-Tam Nguyen, Trung Thanh Nguyen, Jörg Rothe, and Abdallah Saffidine. Positional scoring-based allocation of indivisible goods. *Autonomous Agents and Multi-Agent Systems*, 31(3):628–655, 2017.
[12] S. Brânzei and F. Sandomirskiy. Algorithms for competitive division of chores. *arXiv:1907.01766*, 2019.
[13] E. Budish. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Journal of Political Economy*, 119(6):1061–1103, 2011.
[14] I. Caragiannis, D. Kurokawa, H. Moulin, A. D. Procaccia, N. Shah, and J. Wang. The unreasonable fairness of maximum Nash welfare. *ACM Transactions on Economics and Computation*, 7(3), 2019.
[15] V. Conitzer, R. Freeman, and N. Shah. Fair public decision making. In *Proceedings of the 2017 ACM Conference on Economics and Computation, EC '17*, pages 629–646. ACM, 2017.
[16] S. Ebadian, D. Peters, and N. Shah. How to fairly allocate easy and difficult chores, 2021.
[17] D. Foley. Resource allocation and the public sector. *Yale Econ Essays*, 7:45—98, 1967.
[18] J. Garg and S. Taki. An improved approximation algorithm for maximin shares. *Artificial Intelligence*, page 103547, 2021.
[19] J. Garg, A. Murhekar, and J. Qin. Fair and efficient allocations of chores under bivalued preferences, 2021.
[20] L. Gourvès, J. Lesca, and A. Wilczynski. On fairness via picking sequences in allocation of indivisible goods. In *Algorithmic Decision Theory - 7th International Conference, ADT'21*, volume 13023, pages 258–272. Springer, 2021.
[21] Hadi Hosseini and Kate Larson. Multiple assignment problems under lexicographic preferences. In *AAMAS*, pages 837–845. International Foundation for Autonomous Agents and Multiagent Systems, 2019.
[22] Hadi Hosseini, Sujoy Sikdar, Rohit Vaish, and Lirong Xia. Fair and efficient allocations under lexicographic preferences. In *AAAI*, pages 5472–5480. AAAI Press, 2021.
[23] X. Huang and P. Lu. An algorithmic framework for approximating maximin share allocation of chores. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, pages 630–631, 2021.
[24] Richard E. Korf. A complete anytime algorithm for number partitioning. *Artificial Intelligence*, 106(2):181–203, 1998.
[25] D. Kurokawa, A. Procaccia, and J. Wang. Fair enough: Guaranteeing approximate maximin shares. *Journal of the ACM*, 65(2):1–27, 2018.
[26] B. Li, Y. Li, and X. Wu. Almost proportional allocations for indivisible chores. Technical Report arXiv:2103.11849, arXiv.org, 2021.
[27] R. J. Lipton, E. Markakis, E. Mossel, and A. Saberi. On approximately fair allocations of indivisible goods. In *Proceedings of the 5th*, pages 125–131. ACM Press, 2004.
[28] H. Moulin. *Fair Division and Collective Welfare*. The MIT Press, 2003.
[29] H. Moulin. Fair division in the internet age. *Annual Review of Economics*, 11:1–37, 2019.
[30] B. Plaut and T. Roughgarden. Almost envy-freeness with general valuations. In *Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2018.
[31] H. Steinhaus. The problem of fair division. *Econometrica*, 16:101–104, 1948.
[32] T. Walsh. Fair division: The computer scientist's perspective. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 4966–4972. IJCAI, 2020.

APPENDIX

A PROOF OF PROPOSITION 2.3

Proof of Proposition 2.3. Suppose the allocation $X = (X_1, \cdots, X_n)$ is EFX. For any fixed agent $i \in N$, let o be a lightest chore to agent i, i.e., $o \in \arg\max_{o \in X_i} u_i(o)$. By EFX, for any j,

$$b_j \cdot u_i(X_i \setminus \{o\}) \geq b_i \cdot u_i(X_j).$$

Summing the above inequality over all j gives us

$$\begin{aligned} u_i(X_i \setminus \{o\}) &= u_i(X_i \setminus \{o\}) \cdot \sum_{j \in N} b_j \\ &\geq b_i \cdot \sum_{j \in N} u_i(X_j) = b_i \cdot u_i(O), \end{aligned}$$

which implies that X is PROPX. □

B PROOF OF PROPOSITION 3.1

Proof of Proposition 3.1. We prove by induction over the number of agents that if a fractional allocation satisfying PROP exists, then Algorithm 1 returns a PROPX allocation. The base case for $n = 1$ is trivial. Suppose we know that a PROPX allocation exists for any $1 \leq k' \leq k$ and Algorithm 1 computes such an allocation. We now assume the induction hypothesis for $k' \leq n - 1$ and prove the statement for n agents.

We run the algorithm on the n agents who are to be allocated items from set O'. If no agent is removed and all the agents are allocated, then each agent has PROP guarantee. Otherwise, we consider the first agent who is removed from N'. Suppose the agent is agent i. When agent i is removed, $u(X_i) < -b_i$. Note the last item added to X_i is the smallest chore for agent i. If we remove this chore, the utility of agent i is more than $-b_i$. Hence, if we remove any chore from X_i, the utility of agent i is more than $-b_i$. Hence the bundle of i provides a PROPX guarantee.

For each item $o \in X_i$ and $j \in N' \setminus \{i\}$, $u_j(o) \leq u_i(o)$. Therefore for each $j \in N' \setminus \{i\}$, $u_j(X_i) \leq b_i(u_i(O'))$. It implies that $u_j(O' \setminus X_i) \geq (1 - b_i)(u_j(O'))$. Therefore, $\frac{b_j}{1-b_i} u_j(O' \setminus X_i) \geq b_j u_j(O')$. Hence by getting $\frac{b_j}{1-b_i}$ share of $O' \setminus X_i$, agent j can get b_j share of O'. In other words, a fractional allocation for the k agents provides a proportional share guarantee with respect to all items and all agents. By the induction hypothesis, Algorithm 1 computes a PROPX allocation for the remaining agents. □

C THE FAILURE OF TOP-TRADING ENVY CYCLE ELIMINATION ALGORITHM

Li *et al.* [26] also presented an alternative algorithm called Top-trading Envy Cycle Elimination Algorithm that finds a PROPX outcome for the case of symmetric agents. The algorithm assumed that all agents have the same ordinal preference for all items. The algorithm is based on considering a graph where the nodes correspond to agents. An agent i points to agent j if i is most envious of j. The algorithm works by assigning, at each step, an unassigned item with minimum utility to an agent who does not envy others. If the corresponding graph admits a cycle for the updated allocation, the cycle is resolved by exchanging the bundles of items (an agent in the cycle gets the bundle of the agent she points to). However,

this algorithm does not produce PO Allocations even for 2 agents with symmetric weights.

Example C.1 (Top-trading Envy Cycle Elimination Algorithm does not produce PO Allocations even for 2 agents). Consider an instance with two agents and three items. The utilities are shown in the following table. The agents have identical ordering on the items as per the restriction.

	o_1	o_2	o_3
1	−0.3	−0.3	**−0.4**
2	**−0.2**	**−0.4**	−0.4

The top-trading envy cycle elimination algorithm returns an allocation X as shown in squares. Note that X is not Pareto optimal since it is Pareto dominated by allocation X' with $X'_1 = \{o_2\}$ and $X'_2 = \{o_1, o_3\}$.

D PROOF OF LEMMA 3.5

Proof of Lemma 3.5. Consider any agent i. Since $\max_{o\in X_i} u_i(o) = \max_{o\in Y_i} u_i(o)$, it follows that

$$u_i(Y_i) - \max_{o\in Y_i} u_i(o) = u_i(Y_i) - \max_{o\in X_i} u_i(o).$$

We also know that $u_i(Y_i) \geq u_i(X_i)$ by Pareto improvement. Since X is PROPX, $u_i(Y_i) - \max_{o\in X_i} u_i(o) \geq b_i$. Hence,

$$u_i(Y_i) - \max_{o\in Y_i} u_i(o) = u_i(Y_i) - \max_{o\in X_i} u_i(o) \geq b_i.$$

It follows that allocation Y is PROPX to agent i. □

E PROOF OF LEMMA 4.2

Proof of Lemma 4.2. $G(X)$ contains at most $O(m^2)$ edges since there are in total m items corresponding to m vertices and each item has degree at most m. For any agent i and $o \in X_i$, let $\delta_i(o)$ be the out-degree of o in $G(X)$. Then the number of edges in $G(X)$ equals $\sum_{i\in N}\sum_{o\in X_i}\delta_i(o)$. For any agent i and item $o \in X_i$, if o is involved in the trading cycle, denote by $r(o)$ the item following o, i.e., $o \to r(o)$ is an edge in the trading cycle; if o is not involved, $r(o) = o$. Then $\delta_i(o) \geq \delta_i(r(o))$ since the items can only be exchanged with weakly better ones. Note that since the trading cycle involves a strict edge, there exists an agent i^* and an item $o^* \in X_{i^*}$ such that $\delta_{i^*}(o^*) > \delta_{i^*}(r(o^*))$. Thus after resolving the trading cycle, the number of edges (i.e., $\sum_{i\in N}\sum_{o\in X_i}\delta_i(r(o))$) is strictly decreased. □

F PROOF OF LEMMA 4.3

Proof of Lemma 4.3. The lemma above is well-known for the case of goods (see, e.g. [3]) and works in the same way for the case of chores. We give the argument below.

Note that when allocating goods, the SD relation is written in the following way (see, e.g., [3]). Given two allocations X and Y, $X(i) \succsim_i^{SD} Y(i)$ that is, an agent i *SD prefers* allocation $X(i)$ to allocation $Y(i)$ if $|\{o_k \succsim_i o\} \cap X(i)| \geq |\{o_k : o_k \succsim_i o\} \cap Y(i)|$ for all $o \in O$. We show that the two SD-domination definitions for goods and chores give rise to the same SD-efficiency concept. When $X(i) \succsim_i^{SD} Y(i)$ for all $i \in N$, it follows that $|X(i)| = |Y(i)|$ for all $i \in N$ so the two definitions coincide. □

G PROOF OF LEMMA 4.4

Proof of Lemma 4.4. Let X be a PROPX allocation and Y be the result after resolving one trading cycle in $G(X)$. Let the agents involved in the trading cycle C be N_C. Since X is PROPX, for any $o \in X_i$

$$u_i(X_i \setminus \{o\}) \geq u_i(O) \cdot b_i.$$

For any $i \in N_C$ where $Y_i = X_i \cup \{\alpha\} \setminus \{\beta\}$ and $\beta \in X_i$, considering $\alpha \in Y_i$ gives

$$u_i(Y_i \setminus \{\alpha\}) = u_i(X_i \setminus \{\beta\}) \geq u_i(O) \cdot b_i.$$

For any $o \in Y_i \setminus \{\alpha\}$, we have $o \in X_i$. Since $u_i(\alpha) \geq u_i(\beta)$,

$$\begin{aligned} u_i(Y_i \setminus \{o\}) &= u_i((X_i \cup \{\alpha\} \setminus \{\beta\}) \setminus \{o\}) \\ &= u_i(X_i \setminus \{o\}) + (u_i(\alpha) - u_i(\beta)) \\ &\geq u_i(X_i \setminus \{o\}). \end{aligned}$$

Hence

$$u_i(Y_i \setminus \{o\}) \geq u_i(X_i \setminus \{o\}) \geq u_i(O) \cdot b_i.$$

For any $i \notin N_C$ where $Y_i = X_i$, we have

$$u_i(Y_i) = u_i(X_i) \geq u_i(O) \cdot b_i.$$

Therefore Y is PROPX. □

H FAILURE OF COMBINATION OF LEMMAS 3.5 AND 4.4

Example H.1 (Combination of Lemmas 3.5 and 4.4 terminates before reaching a Pareto allocation.). Consider an instance with two agents $\{1, 2\}$ and four items $\{o_1, o_2, o_3, o_4\}$. The utilities are shown in the following table. Consider a PROPX allocation X as shown by squares, and we Pareto improve X in a way as described in Lemmas 3.5 and 4.4.

	o_1	o_2	o_3	o_4
1	**−0.1**	**−0.2**	−0.25	−0.45
2	−0.2	−0.2	**−0.4**	**−0.2**

(1) If we Pareto improve X in a way as described in Lemma 3.5, agent 1 needs to keep o_1. Then for agent 1's utility to strictly increase, it has to give out o_2 to agent 2 without receiving any other item. This makes agent 2's utility strictly decrease and will hence not obtain any allocation Pareto dominating X. (2) Lemma 4.4 Pareto improves an allocation by implementing trading cycles within it. However, there is no trading cycle in X since $|\{o_k \in X_1 \mid u_1(o_k) \leq u_1(o)\}| = 0$ for all $o \in O$. Therefore, the series of Pareto improvements by implementing trading cycles and exchanging items while keeping the lightest item in each bundle will terminate at X, not being able to obtain any further allocation that Pareto dominates X. However, X is Pareto dominated by the following PROPX and PO allocation, meaning that a combination of Lemma 3.5 and Lemma 4.4 is not adequate for obtaining a PROPX and Pareto optimal allocation under the general case of PROPX allocations without any restrictions.

	o_1	o_2	o_3	o_4
1	−0.1	−0.2	**−0.25**	−0.45
2	**−0.2**	**−0.2**	−0.4	**−0.2**