Project Report **(IITK CS656, 2023-24 II)**

Fairly Dividing Mixtures of Goods and Chores under Lexicographic Preferences

Aditya Tanwar 200057
Akanksha Singh 200070
Soham Samaddar 200990

1 Introduction and Context

Fair division of indivisible items among agents has been an important field of study, primarily owing to its significance in real life. For example, allocation of courses to students, distribution of medical supplies to hospitals and allocation of presentation slides to group members of a project are all examples of division of items. These items can include both **goods** (agents would like these items to be allocated to them) and **chores** (agents would not like these items to be allocated to them).

This paper discusses some results related to dividing **mixed goods**, i.e both goods and chores among agents. Additionally, these items can be **objective** (where every agent agrees that an item is a good or a chore) or **subjective** (where the same item can be seen as a good or a chore by different agents). The utility function for deciding how good an allocation is, is based on a **lexicographic** preference ordering of items by the agents. We define these terms more formally later on.

There are some results in this paper that we feel are incorrect. As such, we have come up with some corrections and results of our own. In this report, we first describe the results of the paper completely, following which we show our results in Section 4.

2 Mathematical Model

We first describe the mathematical model that we use to abstract the problem:

- **Model**
  For any $k \in \mathbb{N}$, we define $[k] \coloneqq \{1, \ldots, k\}$. An instance of the allocation problem with *mixed items* is a tuple $\langle N, M, G, C, \rhd \rangle$ where $N \coloneqq [n]$ is a set of n agents and M is a set of m items $\{o_1, \ldots, o_m\}$. Here, $G \coloneqq (G_1, \ldots, G_n)$ and $C \coloneqq (C_1, \ldots, C_n)$ are collections of subsets of M, where, for each $i \in [n], G_i \subseteq M$ is the set of goods and $C_i = M \backslash G_i$ is the set of chores for agent i, respectively. $\rhd = (\rhd_1, \ldots, \rhd_n)$ is a preference profile, specifying for each $i \in N$, there (lexicographic) preference over the set of items M.

- **Bundles**
  A bundle is any subset $X \subseteq M$ of the items. Given any bundle $X \subseteq M$, we will write $X^{i+} := X \cap G_i$ and $X^{i} := X \cap C_i$ to denote the sets of goods and chores in X, respectively, according to agent i.

- **Allocations**
  An allocation $A = (A_1, \dots, A_n)$ is an n-partition of M, where $A_i \subseteq M$ is the bundle assigned to agent i. An allocation is *complete* if $\bigcup_{i=1}^{n} A_i = M$, while it is *incomplete/partial* otherwise. We will write $\Pi(M)$ to denote the set of all n-partitions of M.
  Unless mentioned otherwise, an allocation refers to a complete allocation in our proceeding discussions.

- **Goods**
  Items that i prefers to have *present* in a bundle. When comparing two sets X and Y, such that a good $g \in X, g \notin Y$, then i prefers the bundle X over the bundle Y *ceteris paribus* (other items being the same in both the sets).

- **Chores**
  Items that i prefers to have *absent* in a bundle. When comparing two sets X and Y, such that a chore $g \in X, g \notin Y$, then i prefers the bundle Y over the bundle X *ceteris paribus* (other items being equal in both the sets).

- **Lexicographic Preferences**
  Each agent i has a lexicographic preference, $\rhd_i$ (a transitive total ordering), over the set of items M. It can be interpreted as which item's membership in a bundle is more important to the agent i. Consequently, $\rhd_i$ of agent i over M induces a preference $\succ_i$ over 2^M as well, i.e., the set of bundles.
  As an example, if $o_1^+ \rhd_i o_2^- \rhd_i o_3^+$, then while comparing two bundles, the agent i would first check the membership of o_1^+ in both the sets, and if it is present in only one of the sets, it would prefer that set since o_1 is a good according to i. If the membership is the same in both the sets (present in both the sets or absent from both the sets), then it would move on to the next item in the preference ordering and so on.
  There is some additional notation in the context of Lexicographic preeferences that we shall require later on:

  * $\rhd_i(k) :=$ The k^{th} most important/prioritized item in the preference order of agent i.
  * $\rhd_i([k]) :=$ The k most important/prioritized items in the preference order of agent i.
  * $\rhd_i(k, S) :=$ The k^{th} most important/prioritized item in the preference order of agent i in the bundle S.
  * $\rhd_i([k], S) :=$ The k most important/prioritized items in the bundle S according to the preference order of agent i.

- **Envy Freeness**
  This is one of the first notions of fairness we discuss, it is also one of the more commonly discussed ones. The basic idea is that an agent does not envy any other agent's bundle – it prefers its own bundle over anyone else's. It comes in different flavours, each varying in its strength (decreasing order):

  * *Envy-Free (EF)*
    An allocation A is EF if for every pair of agents $i, j \in N, A_i \succ_i A_j$.
  * *Envy-Free up to one item (EF1)*: An allocation A is EF1 if for every pair of agents $i, j \in N : A_i^{i-} \cup A_j^{i+} \neq \varnothing, \exists o \in A_i^{i-} \cup A_j^{i+}$ such that either $A_i \succ_i A_j \backslash \{o\}$ or $A_i \backslash \{o\} \succ_i A_j$.
    Informally, it can be understood as we are given a choice to remove one item from A_i or A_j, so that i continues to prefer their bundle over j's (since we want i to prefer A_i, it is optimal to consider removing a chore from A_i or a good from A_j).
  * *Envy-Free up to any item (EFX)*: An allocation A is EFX if for every pair of agents $i, j \in N : A_i^{i-} \cup A_j^{i+} \neq \varnothing, \exists o \in A_i^{i-} \cup A_j^{i+}$ such that if $o \in A_j^{i+}, A_i \succ_i A_j \backslash \{o\}$ and if $o \in A_i^{i-}, A_i \backslash \{o\} \succ_i A_j$.
    Informally, it can be understood as ensuring i prefers their own bundle over any other j's if at most one chore is removed from A_i or at most one good is removed from A_j.
  * Note that $A_i \backslash \{c\} \succ_i A_i$, and $A_j \succ_i A_j \backslash \{g\}$ for a chore c and a good g (according to i). Thus, an allocation being EF implies that it is EFX and EF1 as well. Similarly, since EFX has a *"for all"* quantifier while EF1 has a *"there exists"* quantifier, an allocation being EFX implies that it is EF1 as well.

- **Maximin Share**
  An agent's maximin share is defined mathematically as MMS_i,

$$\mathit{MMS}_i := \max_P \in \Pi(M) \min_j \{P_j\}_{j \in N}$$

  where all the comparisons follow i's preference orderings. Intuitively, it is the best bundle i can *guarantee* itself when i is making the allocations and everyone else picks their bundles. Crudely, to find MMS_i, we pick the worst bundle according to i in each allocation, and maximise this worst bundle over all possible allocations.

- **Pareto Optimality**
  An allocation A is said to be Pareto optimal (PO) if there is no other allocation B such that $\forall i \in N, B_i \succeq_i A_i$ and $\exists j \in N : B_j \succ_j A_j$. Intuitively, an allocation is PO if there is no way to make someone better off without making someone else strictly worse off.

3 Results

The main three results of the paper are as follows:

1. EFX does not always exist – there exists an instance with lexicographic preferences that does not admit any EFX allocation.
2. EFX and Pareto Optimality – there is a subclass of lexicographic instances for which an EFX+PO allocation is guaranteed to exist.
3. Maximin Share (MMS) – An allocation that satisfies MMS always exists and can be found in PTIME.

We elaborate on each of the results in the consequent subsections.

3.1 Non-existence of EFX

We show an allocation instance which does not admit an EFX solution, showing that it is not always possible to have an EFX allocation over mixed items. Furthermore, the counterexample given contains *objective* mixed items only.

Claim 3.1. *Consider the allocation instance* $(N, M, G, C, \rhd)$ *with:*

- $N = \{1, 2, 3, 4\}$
- $M = \{o_1^+, o_2^-, o_3^-, o_4^-, o_5^-, o_6^-, o_7^-\}$ *- since the items are objective, they have been marked as a good or a chore in the item set* M *itself*
- $\rhd_1, \rhd_2 : o_2^- \rhd o_3^- \rhd o_4^- \rhd o_1^+ \rhd o_5^- \rhd o_6^- \rhd o_7^-$
- $\rhd_3, \rhd_4 : o_5^- \rhd o_6^- \rhd o_7^- \rhd o_1^+ \rhd o_2^- \rhd o_3^- \rhd o_4^-$

(Since the items are objective, we omit the description of the sets G *and* C*)*
The above instance has no EFX allocation possible.

Proof. Without loss of generality, let agent 1 get o_1^+. We now split into 3 cases:

- Case 1: $A_1 \cap \{o_2^-, o_3^-, o_4^-\} = \varnothing$
  Now, regardless of what agent 2 is assigned, they will continue to envy agent 1. So, A_2 must be empty, otherwise even after removing a chore from their set, they will continue to envy A_1. So o_2^-, o_3^-, o_4^- will be assigned to agents 3 and 4. One of them must receive at least two chores. That agent will continue to envy the empty bundle A_2, even after one chore is removed from their bundle, making an EFX allocation impossible.

- Case 2: $A_1 \cap \{o_5^-, o_6^-, o_7^-\} = \varnothing$
  Regardless of what agents 3 and 4 are assigned, they will envy A_1. Hence they A_3 and A_4 must be empty, otherwise we could have removed their chores and still maintained enviousness. So, agent 2 must get at least $\{o_5^-, o_6^-, o_7^-\}$. But they will continue to envy A_3 (which is empty) even after removing one chore, a contradiction to EFX again.

- Case 3: $A_1 \cap \{o_2^-, o_3^-, o_4^-\} \neq \varnothing$ and $A_1 \cap \{o_5^-, o_6^-, o_7^-\} \neq \varnothing$
  Let $x \in A_1 \cap \{o_2^-, o_3^-, o_4^-\}$ and $y \in A_1 \cap \{o_5^-, o_6^-, o_7^-\}$. By EFX, agent 1 should prefer their own bundle after y is removed over any other bundle A_2, A_3, A_4. So, each of those bundles must contain a chore ranked higher than x according to A_1. But there are at most 2 such chores available, a contradiction.

Hence EFX is not possible for this instance. □

3.2 EFX and Pareto Optimality

Owing to the negative result in the previous subsection, we try to find (sub-)classes of games that do admit an EFX allocation, and for which, an EFX allocation can be found efficiently as well. Specifically, the paper identifies two broad sub-classes of lexicographic preferences, for which an EFX+PO allocation is guaranteed and can be found in PTIME, these are:

1. $\exists i \in N : \rhd_i(1) \in G_i$
   The subclass where at least one player has a good as their most important item.

2. $\cap_{i \in N} C_i = \varnothing \Leftrightarrow \forall o \in M \exists i \in N : o \in G_i$
   The subclass where there are no common chores.

We now present the algorithm (verbatim) provided by the authors to find an EFX+PO allocation for the first subclass of games:

ALGORITHM 1: Finding an EFX+PO allocation when there is an agent whose top-ranked item is a good.

Input: A lexicographic mixed instance $\langle N, M, G, C, \rhd \rangle$

Output: An EFX+PO allocation A

1 Select an arbitrary agent $i \in N$ such that $\rhd_i(1) \in G_i$

2 Let $C' := \{o \in M : \forall j \in N \setminus \{i\}, o \in C_j\}$ `// The set of all common chores for the remaining agents.`

3 $A_i \leftarrow \rhd_i(1) \cup C'$

4 $N \leftarrow N \setminus \{i\}$

5 $M \leftarrow M \setminus A_i$

$\rhd$ `The remaining instance has no common chore.`

6 **while** *there exists an unallocated item* **do**

7   **if** $|N| = 1$ **then**

8   Assign all items to the remaining agent

9   **else**

10   Find the smallest $k \in \{1, 2, \ldots, |M|\}$ such that the set $S^k := \{i \in N : \rhd_i(k) \in G_i\}$ is non-empty `// set of agents whose` k^{th}`-ranked item is a good.`

11   Select any agent $j \in S^k$

12         $C' := \{o \in M : \forall i \in N \setminus \{j\}, o \in C_i\}$

13         $A_j \leftarrow \{\rhd_j(k)\} \cup C'$

14         $N \leftarrow N \setminus \{j\}$

15         $M \leftarrow M \setminus A_j$

16 **return** A

Note: An algorithm for the second subclass can be made using this algorithm. More precisely, if the above algorithm is run from line 6 through line 16, it returns an EFX+PO allocation in all instances of the second subclass. An explanation on why this works is provided later on.

Intuitively, the algorithm tries to get rid of *common chores* whilst still maintaining satisfaction of the players.
To achieve the same, it picks an agent who has the highest-ranked good remaining, and allots the good to this agent. It then proceeds to allot all the items which are considered *common chores* by all the *remaining* players to that same player. In doing so, the algorithm holds the invariant that a player's most important item in their bundle is still a good (in fact, for players other than the first one, it holds a stronger invariant – all the items in a bundle are goods for the corresponding agent).

We shall discuss the steps of the algorithm, as well as their interpretation, in more detail in the subsequent discussions.

3.2.1 Proof of EFX

In this subsection, we show that the allocation returned by the algorithm is indeed EFX for the first subclass of games.
For the same, we split the proof into two cases, one for the first person to be allotted items, and one

for the rest, since the algorithm treats the first agent differently than the rest as well. Also, before we begin with the proof, for the sake of convenience, we rename the agents so that the i^{th} agent refers to the agent that was called in the i^{th} round of the algorithm. We also refer to the allocation returned by the algorithm as $A = (A_1, \ldots, A_n)$

- Case 1: Agent 1 is EFX under A. (Lines 1-5)

  Observe that, by the way the algorithm defines *agent 1*, we have that $\rhd_1(1) \in G_1$. Also, the algorithm allots $\rhd_1(1)$ to A_1 as well. Since the most important item (a good) according to 1 is in A_1, we obtain that 1 is EF of others, thus EFX naturally follows. Mathematically, for $j \neq 1$,

  $$A_1 \succ_1 A_j \qquad (\rhd_1(1) \in A_1, \rhd_1(1) \notin A_j)$$
  $$A_1 \backslash \{c\} \succ_1 A_1 \succ_1 A_j \succ_1 A_j \backslash \{g\} \qquad (\forall c \in C_1, g \in G_1)$$

  Thus, the claim follows.

- Claim: There are no common chores (w.r.t. remaining agents) left to be allotted at the start of a round $r (\geq 2)$. (Lines 6-15)

  This can be shown by induction on r.
  For $r = 2$, the claim is immediately apparent since all the common chores were allotted to agent 1 in Lines 1-5.
  For $r > 2$, there cannot be any common chores left. This is because in round $(r-1)$, all the common chores are collected and assigned in A_{r-1}.
  Thus, the claim follows from induction.

- Corollary: The item $\rhd_j(k)$ is in G_j. (Line 13)
  If $\rhd_j(k) \notin G_j$, then every item is considered a chore by every agent remaining. This leads to a contradiction, thanks to the previously shown claim.

- Claim: $A_j \subseteq G_j \quad \forall j \geq 2$. (Line 13)
  Clearly, $\rhd_j(k) \in G_j$, by the corollary shown above. We now need to show that $C' \subseteq G_j$.
  Suppose, towards contradiction, $\exists o \in C' \backslash G_j$. Observe that $C' \backslash G_j \equiv C' \cap C_j$. Then, $o \in C' \cap C_j \Rightarrow$ there is a common chore remaining at the beginning of round j, a contradiction.
  Lastly, if agent j has been allotted items due to Lines 7-8, observe that there were no common chores at the start of round j – all the remaining items are goods according to agent $j \Rightarrow A_j \subseteq G_j$. Thus, the claim follows.

  Also, since we have $A_j \subseteq G_j$, to test EFX, we only need to consider removing goods from the other bundle, since A_j does not have any chores to be removed. We shall use this fact in the second case.

- Case 2: Agent $i(\geq 2)$ is EFX under A. (Lines 10-15)

  For $j < i$, observe that the only possible item that can be a good according to i in A_j, was the element that j was given specially, i.e., $\rhd_j(k)$. All of the other items in A_j are necessarily chores according to i (since they were common chores during round j, for all the remaining players other than j). Thus, we have,

  $$G_i \supseteq A_i \succ_i A_j \backslash \{\rhd_j(k)\} \subseteq C_i$$

  Further, for $j > i$, notice that agent i was already given the most important good remaining (according to i) in A_i. Using this, and the fact that $A_i \subseteq G_i$, we can safely conclude that the agent i is EF of all agents $J(> i)$, and hence EFX follows as well.

We claim that the algorithm corresponding to the second subclass also returns an EFX allocation. The proof is almost the same as the one provided above, except that the first case is omitted. Therefore, we omit repeating the proof for the second subclass, and instead provide an argument for why the proof works in the second subclass as well.
We make the keen observation, that from the perspective of the agents in $N \backslash \{1\}$, Lines 1-5 can be seen as *removal of common chores and at most one good*. More importantly, the reduced instance, that remains, has the property that *there are no common chores*, and the loop (Line 6) utilizes this property in its invariant. Observe also that we have only used this property to prove our claims above.
Thus, a truncation of the algorithm to Lines 6-16 does provide an EFX allocation in the second subclass, and in fact, the proof for the same also follows from the truncation of the proof provided above (excepting "Case 1", all the remaining claims and proofs follow).

3.2.2 Proof of PO

We now show that the allocation A returned by the algorithm is Pareto Optimal (PO) as well. For the same, assume, towards contradiction, that there is another allocation $B \neq A$ that Pareto dominates the allocation A, i.e.,

$$\forall i \in N, B_i \succeq_i A_i, \quad \exists j \in N, B_j \succ_i A_j$$

Based on the fact that B Pareto dominates A, we now show some interesting properties between B_i from B, and correspondingly, A_i from A; finally, we shall arrive at $B = A$, which would lead to a contradiction. The proof is as follows:

- $\rhd_i(1, A_i) \in B_i \quad \forall i \in N$
  Interpretation: The most important item for i in A_i has been retained in B_i as well.

  We show the same by inducting on i (increasing order). Before that, we make a quick observation that for all $i \in N$, if $A_i \neq \varnothing$, then $\rhd_i(1, A_i) \in G_i$ – evident from prior discussions. Since this most important item is a good for all the agents, we use g_i to denote $\rhd_i(1, A_i)$. Now, as for the induction:

* For $i = 1$, this object is also the most important item, $\rhd_1(1)$, according to agent 1. If it is not, then, we have,

$$\rhd_1(1) \in G_1, \quad \rhd_1(1) \in A_1, \quad \rhd_1(1) \in B_1 \Rightarrow A_1 \succ_1 B_1$$

  Thus, contradicting the fact that $B_i \succeq_i A_i \quad \forall i \in N$.
  Hence, $\rhd_1(1, A_i) \in B_i$.

* For $i > 1$, this object is the most important good from amongst the *remaining* items. Suppose it is not retained in B_i, then since $B_i \succeq_i A_i$, we must have that either there is a more important chore that is present in A_i and not in B_i, or we must have a more important chore that is present in B_i and not in A_i, by lexicographic preferences.
  Since $A_i \subseteq G_i$, the first case is not possible. The only possibility is the second one – a more important good is present in B_i which is absent from A_i; call this good g' with $g' \rhd_i g_i$. As A is a complete allocation, g' must be present in some $A_j, j \neq i$.
  Now, if $j < i$, then since g' is a good for agent i, g' has to be $\rhd_j(1, A_j)$, as all the other items in A_j would have been chores according to i. As we are inducting on i in increasing order, $g' = \rhd_j(1, A_j) \in B_j$. The only remaining possibility is that $g' \in A_j$ for some $j > i$. But note that, during round i, agent i picked the most important good from the remaining items. If $g' \rhd_i g_i$, then i would not have picked g_i in the algorithm. Thus, if $g_i \notin B_i$, then $B_i \succeq_i A_i$ is not possible. Hence g_i is retained in B_i for $i \neq 1$ as well.

This completes the proof by induction.

- $A_i \subseteq B_i \quad \forall i \in N$
  Interpretation: All the items in A_i have been retained in B_i as well.

  We show the same by again inducting on i, but this time in reverse order:

  * For $i > 1$, suppose that an item, $g' \in A_i$ is not retained in B_i. Then, since $A_i \subseteq G_i$, g' has to be a good. Also, since B pareto dominates A, we have $B_i \succeq_i A_i$. Since there are no chores in A_i, this necessarily means that there is a more important good than g' in B_i. Call this more important good, g_b, such that $g_b \in B_i, g_b \notin A_i$. Since A is a complete allocation, g_b must have been allocated to some $A_j, j \neq i$.
    If $j < i$, then $g_b \in G_i, g_b \in A_j \Rightarrow g_b = \rhd_j(1, A_j)$ (the most important good according to j from the remaining items at the beginning of round j). By the previous proof, $g_b = \rhd_j(1, A_j) \in B_j$.
    Then, the only possibility that remains is $j > i$. But, since we are inducting on decreasing i, $g_b \in A_j \Rightarrow g_b \in B_j$ because $A_j \subseteq B_j$.
    Again, it follows that if $A_i \not\subseteq B_i$, then B cannot Pareto dominate A.
  * For $i = 1$, if $\rhd_1(1)$ is retained in B_1 by the previous property of B. For all the other items in A_1, if they are not retained in B_1, then they will go to some $B_j, j > 1$. Then, B_j will strictly be worse than A_j, due to the inclusion of a chore, thus violating B's Pareto dominance over A.

Thus, the claim follows from induction on i

- $B = A$
  Since $A_i \subseteq B_i \, \forall i \in N$, and both A and B are complete allocations, it must be the case that $B_i = A_i \, \forall i \in N \Rightarrow B = A$.

Thus, we have shown that if B Pareto dominates A, then the only possibility is that $B = A$, which leads to a contradiction, completing the proof that A is PO for the first subclass.

As with the proof for showing A is EFX for the first subclass and second subclass, the proof for showing A is PO for the second subclass is also more or less the same as the proof for the first subclass, except that the discussions for agent 1 can be omitted in the second subclass – all the other claims and properties, as well as their proofs naturally translate to the second subclass as well.

3.3 Maximin Share (MMS)

For the sake of convenience, we mention the mathematical definition of an MMS bundle here, though it has been discussed briefly in the previous section,

$$\mathit{MMS}_i := \max_P \in \Pi(M) \min_j \{P_j\}_{j \in N}$$

In this section, we first characterize what the MMS bundle looks like for an agent i based on their lexicographic preferences, followed by the proof provided for EFX$\Longrightarrow$MMS, and finally, using it to show that MMS always exists and can be computed efficiently.
Note: We feel that there are some fallacies in the arguments provided for EFX$\Longrightarrow$MMS, and we have consequently constructed a counter-example for the same. However, our fixes (in the proof)/counter-example/proposed solutions are included in a later section named our contributions. We have only highlighted the arguments we disagree with, and discuss the disagreement in more detail in the same later section.

3.3.1 Characterizing MMS

We give a complete characterization of the MMS allocation of an agent.

Claim 3.2.

$$\mathit{MMS}_i = \begin{cases} G_i \backslash \{\rhd([n-1], G_i)\}, & if \rhd_i (1) \in G_i \wedge |G_i| \geq n \\ \varnothing, & if \rhd_i (1) \in G_i \wedge |G_i| < n \\ \rhd_i(1, C_i) \cup G_i, & if \rhd_1 (1) \in C_i \end{cases}$$

Proof. We explicitly construct the allocations case by case.

- Case 1: $\triangleright_i(1) \in G_i \land |G_i| \geq n$
  Consider the partition:

  $$A = (\{\triangleright_i(1, G_i) \cup C_i\}, \{\triangleright_i(2, G_i)\}, \{\triangleright_i(3, G_i)\}, \cdots, \{\triangleright_i(n-1, G_i)\}, \{G_i \backslash \{\triangleright([n-1], G_i)\}\})$$

  Clearly, the worst bundle for i in the above partition is $A_n = \{G_i \backslash \{\triangleright([n-1], G_i)\}\}$, since all other bundles contain a good which is lexicographically preferred over every good in A_n. Additionally, all chores are pushed under A_1 and since the top item of i is a good, A_1 continues to be preferred by i over every other bundle. We omit a formal proof but intuitively, it can be seen that i cannot do a better partition. Hence $A_n = \mathit{MMS}_i$.

- Case 2: $\triangleright_i(1) \in G_i \land |G_i| < n$
  Consider the partition:

  $$A = (\{\triangleright_i(1, G_i) \cup C_i\}, \{\triangleright_i(2, G_i)\}, \{\triangleright_i(3, G_i)\}, \cdots, \{\triangleright_i(|G_i|, G_i)\}, \varnothing, \cdots, \varnothing)$$

  There are $n - |G_i|$ empty bundles in the above allocation. The argument is exactly the same as in case 1. Essentially, we have run out of goods to give i and hence $\mathit{MMS}_i = \varnothing$.

- Case 3: $\triangleright_1(1) \in C_i$
  Consider the partition:

  $$A = (\{\triangleright_i(1) \cup G_i\}, \cdots)$$

  Since the top item of agent i is a chore, he will always prefer any bundle which does not contain $\triangleright_i(1)$ to one which does. Hence in any allocation, the bundle containing $\triangleright_i(1)$ will become the worst in the partition for i. As such, he improves his bundle as much as he can by adding all his goods to it.

$\square$

3.3.2 EFX $\implies$ MMS

Claim 3.3. *For mixed items under lexicographic preferences, an EFX allocation (whenever it exists) satisfies MMS, but the converse is not always true.*

Note: This claim is, in fact, incorrect. We construct a counter-example (and provide some fixes) for the same in our contribution. However, we have provided the authors' reasoning in the following proof.

Proof. We prove the same via contradiction – assume there is an allocation A that is EFX, if A does not satisfy MMS, then it must be the case that A is not EFX either, a contradiction.
Indeed, let A be an allocation that is EFX, and let it not satisfy MMS for an agent i. We split the proof into some cases, inspired by the characterisation of MMS in the previous section:

1. $\rhd_i(1) \in C_i \Rightarrow \mathit{MMS}_i = \{\rhd_i(1)\} \cup G_i$
   Interpretation: When the top-ranked item in i's preference ordering is a chore.

   As A violates MMS for the agent i, the authors claim that A_i must contain $\rhd_i(1)$, and a strict subset of its goods G_i. Assuming this to be true, the proof can be completed by saying that a good $g \in G_i$ is received by some h in their bundle A_h, and it then follows that $A_h \backslash \{g\} \succ_i A_i$, violating A being EFX.
   Note: This does not have to be always true, since A_i can contain all the items in MMS_i, and an additional chore, resulting in $\mathit{MMS}_i \succ_i A_i$. However, this part of the proof can be recitified, and one possible rectification is provided later in our contributions.

2. $\rhd_i(1) \in G_i, |G_i| < n \Rightarrow \mathit{MMS}_i = \varnothing$
   Interpretation: When the top-ranked item in i's preference is a good, and the number of items considered as a good by i is less than n.

   Observe that since *MMS* is violated for i, we must have $\mathit{MMS}_i = \varnothing \succ_i A_i$. Consequently, this must mean that the top-ranked item in A_i is a chore according to i (if it were a good, it would prefer A_i over $\varnothing$). It also means that $\rhd_i(1)$ is not in A_i and is instead received by some h in A_h. Thus, it follows that $A_h \succ_i A_i \backslash \{\rhd_i(1, A_i)\}$, violating A being EFX, a contradiction.

3. $\rhd_i(1) \in G_i, |G_i| \geq n \Rightarrow \mathit{MMS}_i = G_i \backslash \{\rhd_i([n-1], G_i)\}$
   Interpretation: When the top-ranked item in i's preference is a good, and the number of items considered as a good by i is at least n.

   Note: Since $\mathit{MMS}_i \succ_i A_i$ (A violates MMS for i), the authors claim that agent i does not receive any of its favorite $(n-1)$ goods under A. Again, this does not have to be the case. This is elaborated further in our contributions.
   We now split this case into two further subcases, based on whether i is allocated any chores in A_i or not. Before starting the cases, we observe that $\rhd_i(1) \notin A_i$, since it is a good and $\mathit{MMS}_i \succ_i A_i$.

   (a) $A_i^{i-} \neq \varnothing$
   Interpretation: A_i contains some chores (according to i).

   Let $\rhd_i(1)$ be given to h in A_h. Further, let $c \in A_i^{i-}$. Then, we have, $A_h \succ_i A_i \backslash \{c\}$, contradicting that A is an EFX allocation.

   (b) $A_i^{i-} = \varnothing$
   Interpretation: A_i does not contain any chores (according to i).

   It is in this case, that the above claim holds – A_i does not receive any of its $(n-1)$ most important goods (if it did, then since A_i does not have any chores, i would strictly prefer A_i over *MMS*$_i$). Further, there is some good in *MMS*$_i$ that has to be missing from A_i.

Since there are n goods that are distributed among $(n-1)$ agents, there has to be some agent that is given 2 goods. Let this agent be h. Further, let the good that is preferred less (by i) among these be g. The authors then claim that $A_h \backslash \{g\} \succ_i A_i$ since $A_h \backslash \{g\}$ contains a good from $\rhd_i([n-1])$, while $A_i \subseteq \mathit{MMS}_i$, thus contradicting A being EFX.

<u>Note</u>: Just because A_h has at least two goods, one of them from the most important $(n-1)$ ones, does not mean that i has to prefer $A_h \backslash \{g\}$ over A_i. This is because A_h could have a more important chore which dissuades i from preferring it. For example, it might be the case that $\rhd_i(1, C_i) \in A_h$, and this item is more important than anything in A_h^{i+}.

We construct a counter-example inspired specifically by this observation in our contributions.

□

3.3.3 Computing MMS

We have provided the algorithm (verbatim) devised by the authors below, for computing an MMS in a mixed (possibly subjective) instance. The algorithm is the same as Algorithm 1, when there exists a player with a good as their most important item. In the second case, a new algorithm has been described.

ALGORITHM 2: Algorithm for finding an MMS allocation for mixed items.

Input: A lexicographic mixed instance $\langle N, M, G, C, \rhd \rangle$
Output: An MMS allocation A

Let $C' := \{o \in M : \forall i \in N,\ o \in C_i\}$
▷ `Step 1: Assigning chores according to top-ranked items`
if $\exists i \in N$ *such that* $\rhd_i(1) \in G_i$ **then**
    Run Algorithm 1
else `// Else if` $\forall i \in N, \rhd_i(1) \in C_i$
    Fix a priority ordering σ over n agents
    if $|C'| \geq n$ **then**
        Run a serial dictatorship where σ_1 picks his best $|C'| - n + 1$ chores
        All remaining agents pick one chore
    else
        Agents pick one chore according to σ, and none if no chore is remaining
    If exists an agent who picked its worst chore (highest priority), give that agent its remaining goods
▷ `Step 2: Serial dictatorship for assigning remaining items`
Run a serial dictatorship according to priority ordering τ; agents pick any number of goods among remaining items or nothing (if no item is a good for them).
return A

The basic idea of the second algorithm is that the common chores are distributed (in a serial dictatorship), and at most one player gets their worst chore during this distribution. If there is indeed a player that gets their worst chore, we make sure to provide them their remaining goods, so that their bundle is exactly equal to their MMS bundle. All the remaining players only get an item they consider a good, if they get any item at all.

Claim 3.4. *MMS can be computed in* PTIME.

Proof. We split into two cases:

- Case 1: $\exists\, i \in N$, such that $\triangleright_i(1) \in G_i$
  We know that in this subclass of allocation instance, an EFX allocation always exists. We compute the EFX allocation in PTIME, using Algorithm 1. Using the claim, EFX⇒MMS, we get that this same allocation is MMS. *Note that this claim is incorrect however, and hence the proof breaks down here.*

- Case 2: $\forall\, i \in N, \triangleright_i(1) \in C_i$
  All agents' MMS bundles contain $\triangleright_i(1)$, which is a chore. So, if an agent is allocated a bundle which does not contain their most hated chore, that bundle is preferred over their MMS bundle. The above algorithm creates an allocation where at most one agent gets their most hated chore. In such a case, that agent also receives all his goods, making his bundle equal to the MMS bundle. All other agents receive a bundle strictly better than their MMS bundle.

  Since we allow agents to pick chores from C' as per a serial dictatorship σ, as long as $|C'| \geq 2$, an agent can ensure that they do not pick their most hated chore. Only when $|C'| = 1$ do we run the risk of allocating the most hated chore to an agent. In such a scenario, suppose agent i receives $\triangleright_i(1)$. Since $G_i \cap C' = \varnothing$, all items in G_i remain unallocated, and hence we can give G_i to i. Finally, all the items that remain are considered good by at least one agent. Hence, we allocate each item to an agent who considers them good. This only improves their allocation, which was already strictly better than their MMS bundle. Hence proved.

□

4 Our Contribution

In this section, we provide some negative, as well as, positive results. We start off by providing a counter-example to the claim EFX $\implies$ MMS, along with a discussion on the extent to which the provided proof was correct. We then show

4.1 EFX $\not\Longrightarrow$ MMS

4.1.1 Partial fixes to the proof

We noticed three arguments in the proof for EFX $\implies$ MMS that we feel might be wrong. In two of them, we were able to redeem the proof. We mention these here:

1. $\rhd_i(1) \in C_i \Rightarrow \mathit{MMS}_i = \{\rhd_i(1)\} \cup G_i$
   "A_i must contain $\rhd_i(1)$, and a strict subset of its goods G_i."

   As mentioned before, $\mathit{MMS}_i \succ_i A_i$ does not necessarily mean that $A_i^{i+} \subset G_i$. Note that $A_i^{i+} \subset G_i$, coupled with $\rhd_i(1) \in A_i$, does necessarily imply that $\mathit{MMS}_i \succ_i A_i$, since i would prefer a bundle containing the good over a bundle that does not.
   However, another reason that i could prefer MMS_i over A_i is if $\mathit{MMS}_i \subset A_i$ and A_i has some additional chore. In such a case, since i would prefer a bundle *not* containing a chore over a bundle that does contain a chore, we would have $\mathit{MMS}_i \succ_i A_i$.

   Fortunately, if A_i retains each item in MMS_i, and it is still the case that $\mathit{MMS}_i \succ_i A_i$, then it must be the case that A_i has some other chore c as well. This would then mean, that for any $j\ (\neq i)$, we have $A_j \succ_i A_i \backslash \{c\}$, thus violating EFX of A.

2. $\rhd_i(1) \in G_i, |G_i| \geq n \Rightarrow \mathit{MMS}_i = G_i \backslash \{\rhd_i([n-1], G_i)\}$
   "Agent i does not receive any of its favorite $(n-1)$ goods under A"

   Again, this does not have to be the case necessarily, as A_i can have some of these goods along with a more important chore, making i prefer MMS_i over A_i.
   Correction: This claim is always correct when $A_i \cap C_i = A_i^{i-} = \varnothing$.

   Fortunately, this fact is used only in the second subcase, where $A_i^{i-} = \varnothing$, so that the statement is indeed true, and its consequences can be used.

In the third argument, however, we were able to construct a counter-example that shows that an EFX allocation does not always have to satisfy MMS as well.
However, fixing the proof for the other cases helps us characterize the possibilities when an EFX can possibly fail to be MMS. Specifically, it can fail to satisfy MMS only for those agents i for whom,

$$\rhd_i(1) \in G_i, \quad |G_i| \geq n, \quad A_i^{i-} = \varnothing$$

4.1.2 The Counter-example

Contrary to the paper claiming EFX $\implies$ MMS, we show a counterexample against the same.

Claim 4.1. *Consider the allocation instance* $(N, M, G, C, \rhd)$ *with:*

- $N = \{1, 2, 3\}$
- $M = \{o_1, o_2, o_3, o_4, o_5\}$
- $G_1 = G_2 = G_3 = \{o_1, o_4, o_5\}$ *and* $C_1 = C_2 = C_3 = \{o_2, o_3\}$
- $\rhd_1 : o_1^+ \rhd_1 o_2^- \rhd_1 o_3^- \rhd_1 o_4^+ \rhd_1 o_5^+$
- $\rhd_2 : o_1^+ \rhd_2 o_2^- \rhd_2 o_3^- \rhd_2 o_4^+ \rhd_2 o_5^+$
- $\rhd_3 : o_4^+ \rhd_3 o_5^+ \rhd_3 o_1^+ \rhd_3 o_2^- \rhd_3 o_3^-$

Consider the allocation $A = (\varnothing, \{o_1, o_2\}, \{o_3, o_4, o_5\})$. *Then* A *is EFX but* A *is not MMS.*

Proof. Using the MMS characterization theorems, we can see that *MMS*$_1 = \{o_5\}$. But, *MMS*$_1 \succ_1 A_1$. So A does *not* satisfy *MMS*. We now show that A *is EFX*, giving us a counterexample.

Note that agents 2 and 3 have their top item (which is a good) in their bundle. Hence, they prefer their own bundle over anyone else's bundle. So they satisfy the EFX criteria.
For agent 1, we only need to consider removing goods from the other agents' bundle, since his own bundle does not contain any chore. Indeed, we have:

$$A_1 \succ_1 A_2 \backslash \{o_1\} = \{o_2\} \qquad (o_2 \text{ is a chore for } 1)$$
$$A_1 \succ_1 A_3 \backslash \{o_4\} = \{o_3, o_5\} \qquad (o_3 \text{ is a chore with higher priority for } 1)$$
$$A_1 \succ_1 A_3 \backslash \{o_5\} = \{o_3, o_4\} \qquad (o_3 \text{ is a chore with higher priority for } 1)$$

Agent 1 satisfies the EFX criteria as well, and the conclusion follows. □

Note: The provided example helps exhibit that an EFX allocation might not satisfy MMS even in **objective** instances of lexicographic preferences.

4.1.3 Intuition behind the Counter-example

We now provide some intuition on how this counter-example was constructed. Note that the only case where the proof for $EFX \Rightarrow$ *MMS* fails, is when the MMS is violated for an i such that

$$\rhd_i(1) \in G_i, |G_i| \geq n, A_i^{i-} = \varnothing$$

So, we construct the (partial) preference of agent 1 to obey these conditions, then construct an EFX allocation that is a strict subset of *MMS*$_1 = \{o_5\}$. Naturally then, A_1 has to be $\varnothing$. We now want to

ensure EFX for agent 1 – we need to check goods present in other bundles.
We observe that some agent gets $\rhd_1(1)$ in their bundle, say agent 2. Then, A_2 cannot contain any other good g (if it did, then $A_2 \setminus \{g\} \succ_1 A_1$, violating EFX) – A_3 gets the remaining goods, namely, o_4 and o_5.

We realise that we have the liberty to allot some chores from C in A_2. We, thus allot o_2 in A_2. We also need to ensure that agent 1 is envy-free (upto EFX) of A_3 as well – allot a chore which is more important than o_4 and o_5. This also allows us to construct the remaining preference ordering of agent 1. The last thing we need to ensure is that agents 2 and 3 are envy-free (at least EFX) – we do so by assigning items from their bundles (A_2 and A_3 respectively) as their most important goods.
This completes the construction.

4.2 Computing MMS

Because of the previous result, it is worth asking if an MMS allocation always exists. However, despite the fact that an allocation being EFX does not imply satisfaction of MMS, it can be shown that an MMS allocation does always exist and can be found efficiently.
We provide an algorithm to show the same, which returns an allocation which is always MMS:

ALGORITHM 3: Finding an MMS allocation

Input: A lexicographic mixed instance $\langle N, M, G, C, \rhd \rangle$
Output: An MMS allocation A
$C_N \leftarrow \{i \in N : \rhd_i(1) \in C_i\}$
$G_{<n} \leftarrow \{i \in N : \rhd_i(1) \in G_i, |G_i| < n\}$
$G_{\geq n} \leftarrow \{i \in N : \rhd_i(1) \in G_i, |G_i| \geq n\}$
if $G_{\geq n} \neq \varnothing$ **then**
    $A_i \leftarrow \varnothing \quad \forall i \in C_N \cup G_{<n}$
    Fix an ordering σ over $G_{\geq n}$
    loop Run the serial dictatorship σ:
        $j \leftarrow$ Current agent according to σ
        $g_j \leftarrow \rhd_j(1, M)$ // g_j is the most important good in the remaining items
        $A_j \leftarrow \{g_j\}$
        $M \leftarrow M \backslash \{g_j\}$
        if j is the last agent in σ **then**
            $A_j \leftarrow A_j \cup (G_j \cap M)$ // Give j all the remaining items from G_j
    endloop
    $A_{\sigma(1)} \leftarrow A_{\sigma(1)} \cup M$ // Give all the remaining items to $\sigma(1)$
else if $G_{<n} \neq \varnothing$ **then**
    $A_j \leftarrow \varnothing \quad \forall j \in N$
    Let $i \in G_{<n}$ // Arbitrary but fixed agent
    $A_i \leftarrow M$ // Give all items to agent i
else
    Run Algorithm 2
end if

Claim 4.2. *The allocation returned by Algorithm 3 satisfies MMS.*

Proof. The algorithm behaves differently based on the sizes of the sets $G_{\geq n}, G_{<n}, C_N$. We, thus, structure the proof around the same three cases. However, we make an important observation about the agents in these sets, specifically, we have:

$$\varnothing \succeq_i \mathit{MMS}_i \quad \forall i \in G_{<n} \cup C_N$$

This is because MMS_i has a chore as its most important item when $i \in C_N$, and $\mathit{MMS}_i = \varnothing$ when $i \in G_{<n}$.
We now begin with the proof:

- $G_{\geq n} \neq \varnothing$
  Following our previous observation, we conclude that the allocation satisfies MMS_i for each agent in $C_N \cup G_{<n}$. We only need to discuss MMS for $G_{\geq n}$.
  Now, $A_{\sigma(1)}$ satisfies MMS, since $\sigma(1)$ gets to pick their most important item, $\rhd_{\sigma(1)}(1)$ – a good, in its bundle. Therefore, even if more items are added to $A_{\sigma(1)}$, it continues to be more preferred over $MMS_{\sigma(1)}$ by $\sigma(1)$. Also note that everyone in $G_{\geq n}$, other than $\sigma(1)$ possibly, only has goods in their bundle.
  Let j be the last agent according to σ. Then, everyone in $G_{\geq n} \setminus \{\sigma(1), j\}$ gets to pick one from their most important $(n-1)$ goods (because their turn number is no more than $(n-1)$ in the ordering). Also note that everyone in $G_{\geq n} \setminus \{\sigma(1), j\}$ only has goods, therefore, this allocation satisfies MMS for them as well (MMS in $G_{\geq n}$ is simply every good except for the $(n-1)$ best ones).
  Finally, for j, observe that in the worst case, it gets to pick $\rhd_j(n)$. However, j is also handed all of the remaining goods from G_j as well. Since there have been n picks, these goods can be missing all of the n best goods in the worst case. Thus, even in the worst case, we have,

  $$A_j = \rhd_j(n) \cup (G_j \setminus \{\rhd_j([n], G_j)\}) = G_j \setminus \{\rhd_j([n-1], G_j)\} = MMS_j$$

  Thus, even in the worst case, MMS is satisfied for j as well.

- $G_{>n} \neq \varnothing, G_{\geq n} = \varnothing$
  Let $i \in G_{<n}$ be the person who is given all the items. The allocation satisfies MMS for everyone else, $j \in G_{<n} \cup C_N$, since $\varnothing \succeq_j MMS_j$.
  As for i, observe that i receives $\rhd_i(1)$, a good, in their bundle, thus, $A_i \succ_i MMS_i = \varnothing$.
  Evidently then, the allocation returned by the algorithm satisfies MMS in this case as well.

- $C_N = N, G_{>n} = G_{\geq n} = \varnothing$
  This is the same case as in the algorithm (Algorithm 2) discussed from the paper. We omit writing the proof here since it is the exact same in that algorithm's discussion.

□

Since the three cases are collectively exhaustive, this algorithm manages to show that an MMS allocation always exists. Since the algorithm is also PTIME, we have shown its existence and efficient calculation without relying on the relation between EFX and MMS.

5 Conclusion

We discussed, with proofs, the three main results of the paper, "Fairly Dividing Mixtures of Goods and Chores under Lexicographic Preferences", namely (a) EFX is not guaranteed to exist, even in objective instances, (b) there are subclasses for which an EFX+PO allocation is guaranteed to exist and can be found efficiently, and (c) MMS is always guaranteed to exist.
Additionally, we also highlighted some potential fallacies in some of the arguments provided in the paper, followed by our proposed solutions for the same.