Fairly Dividing Mixtures of Goods and Chores under Lexicographics Preferences

Hosseini, Sikdar, Vaish & Xia

Soham Samaddar Aditya Tanwar Akanksha Singh

April 2024

Acknowledgement

Before we begin, we would like to thank:

- The authors of the paper:
  Hadi Hosseini (PSU), Sujoy Sikdar (BU), Rohit Vaish (IITD), and Lirong Xia (RPI)
- Our course instructor Dr. Sunil Simon

Introduction

- Lexicographic Preferences
  - Have we seen them before?
- Fair Division – lack of *envy*

Real World Significance

Division of indivisible goods between players that have lexicographic preferences has several real world applications:

- Course allocation in universities
- Distribution of various medical supplies to hospitals
- Allocation of public housing units

Mathematical Abstraction

Allocation Problem

An allocation problem is a tuple $(N, M, G, C, \succ)$ defined as:

- N is a set of n agents, $\{1, 2, \cdots, n\}$
- M is a set of m items, $\{1, 2, \cdots, m\}$
- $G := (G_1, G_2, \cdots, G_n)$ is a set of subsets of M, G_i are goods for the i^{th} agent
- $C := (C_1, C_2, \cdots, C_n)$ is a set of subsets of M, C_i are chores for the i^{th} agent
- $\succ := (\succ_1, \succ_2, \cdots, \succ_n)$ is a set of lexicographic preference profiles, $\succ_i$ denotes the preference ordering over M for the i^{th} agent

An allocation $A = (A_1, A_2, \cdots, A_n)$ is a **partition** of M such that the i^{th} agent receives the items in A_i.

Note: $\forall i \in N, G_i \cap C_i = \Phi, G_i \cup C_i = M$

Mathematical Abstraction

Allocation Problem

An allocation problem is a tuple $(N, M, G, C, \succ)$ defined as:

- N is a set of n agents, $\{1, 2, \cdots, n\}$
- M is a set of m items, $\{1, 2, \cdots, m\}$
- $G := (G_1, G_2, \cdots, G_n)$ is a set of subsets of M, G_i are goods for the i^{th} agent
- $C := (C_1, C_2, \cdots, C_n)$ is a set of subsets of M, C_i are chores for the i^{th} agent
- $\succ := (\succ_1, \succ_2, \cdots, \succ_n)$ is a set of lexicographic preference profiles, $\succ_i$ denotes the preference ordering over M for the i^{th} agent

An allocation $A = (A_1, A_2, \cdots, A_n)$ is a **partition** of M such that the i^{th} agent receives the items in A_i.

Note: $\forall i \in N, G_i \cap C_i = \Phi, G_i \cup C_i = M$

Mathematical Abstraction

Allocation Problem

An allocation problem is a tuple $(N, M, G, C, \succ)$ defined as:

- N is a set of n agents, $\{1, 2, \cdots, n\}$
- M is a set of m items, $\{1, 2, \cdots, m\}$
- $G := (G_1, G_2, \cdots, G_n)$ is a set of subsets of M, G_i are goods for the i^{th} agent
- $C := (C_1, C_2, \cdots, C_n)$ is a set of subsets of M, C_i are chores for the i^{th} agent
- $\succ := (\succ_1, \succ_2, \cdots, \succ_n)$ is a set of lexicographic preference profiles, $\succ_i$ denotes the preference ordering over M for the i^{th} agent

An allocation $A = (A_1, A_2, \cdots, A_n)$ is a **partition** of M such that the i^{th} agent receives the items in A_i.

Note: $\forall i \in N, G_i \cap C_i = \Phi, G_i \cup C_i = M$

Mathematical Abstraction

Allocation Problem

An allocation problem is a tuple $(N, M, G, C, \succ)$ defined as:

- N is a set of n agents, $\{1, 2, \cdots, n\}$
- M is a set of m items, $\{1, 2, \cdots, m\}$
- $G := (G_1, G_2, \cdots, G_n)$ is a set of subsets of M, G_i are goods for the i^{th} agent
- $C := (C_1, C_2, \cdots, C_n)$ is a set of subsets of M, C_i are chores for the i^{th} agent
- $\succ := (\succ_1, \succ_2, \cdots, \succ_n)$ is a set of lexicographic preference profiles, $\succ_i$ denotes the preference ordering over M for the i^{th} agent

An allocation $A = (A_1, A_2, \cdots, A_n)$ is a **partition** of M such that the i^{th} agent receives the items in A_i.

Note: $\forall i \in N, G_i \cap C_i = \Phi, G_i \cup C_i = M$

Mathematical Abstraction

Allocation Problem

An allocation problem is a tuple $(N, M, G, C, \succ)$ defined as:

- N is a set of n agents, $\{1, 2, \cdots, n\}$
- M is a set of m items, $\{1, 2, \cdots, m\}$
- $G := (G_1, G_2, \cdots, G_n)$ is a set of subsets of M, G_i are goods for the i^{th} agent
- $C := (C_1, C_2, \cdots, C_n)$ is a set of subsets of M, C_i are chores for the i^{th} agent
- $\succ := (\succ_1, \succ_2, \cdots, \succ_n)$ is a set of lexicographic preference profiles, $\succ_i$ denotes the preference ordering over M for the i^{th} agent

An allocation $A = (A_1, A_2, \cdots, A_n)$ is a **partition** of M such that the i^{th} agent receives the items in A_i.

Note: $\forall i \in N, G_i \cap C_i = \Phi, G_i \cup C_i = M$

Mathematical Abstraction

Allocation Problem

An allocation problem is a tuple $(N, M, G, C, \succ)$ defined as:

- N is a set of n agents, $\{1, 2, \cdots, n\}$
- M is a set of m items, $\{1, 2, \cdots, m\}$
- $G := (G_1, G_2, \cdots, G_n)$ is a set of subsets of M, G_i are goods for the i^{th} agent
- $C := (C_1, C_2, \cdots, C_n)$ is a set of subsets of M, C_i are chores for the i^{th} agent
- $\succ := (\succ_1, \succ_2, \cdots, \succ_n)$ is a set of lexicographic preference profiles, $\succ_i$ denotes the preference ordering over M for the i^{th} agent

An allocation $A = (A_1, A_2, \cdots, A_n)$ is a **partition** of M such that the i^{th} agent receives the items in A_i.

Note: $\forall i \in N, G_i \cap C_i = \Phi, G_i \cup C_i = M$

Mathematical Abstraction

Allocation Problem

An allocation problem is a tuple $(N, M, G, C, \succ)$ defined as:

- N is a set of n agents, $\{1, 2, \cdots, n\}$
- M is a set of m items, $\{1, 2, \cdots, m\}$
- $G := (G_1, G_2, \cdots, G_n)$ is a set of subsets of M, G_i are goods for the i^{th} agent
- $C := (C_1, C_2, \cdots, C_n)$ is a set of subsets of M, C_i are chores for the i^{th} agent
- $\succ := (\succ_1, \succ_2, \cdots, \succ_n)$ is a set of lexicographic preference profiles, $\succ_i$ denotes the preference ordering over M for the i^{th} agent

An allocation $A = (A_1, A_2, \cdots, A_n)$ is a **partition** of M such that the i^{th} agent receives the items in A_i.

Note: $\forall i \in N, G_i \cap C_i = \Phi, G_i \cup C_i = M$

Mathematical Abstraction

We define some additional notations:

- **Goods**: (g_i or o_i^+)
- **Chores**: (c_i or o_i^-)
- **Bundles**: Any subset $X \subseteq M$ of items. For any bundle X, let $X^{i+} := X \cap G_i$ and $X^{i-} := X \cap C_i$.
- $\succ_i (k)$: k^{th} ranked item in the preference ordering of i
- $\succ_i (k, S)$: k^{th} ranked item in the bundle S according to the preference order of i
  - $\succ_i (k) = \succ_i (k, M)$
- $> := (>_1, >_2, \cdots, >_n)$ is a set of preferences over bundles such that $>_i$ denotes the preference ordering over 2^M for the i^{th} agent, induced by $\succ_i$

Mathematical Abstraction

We define some additional notations:

- **Goods**: (g_i or o_i^+)
- **Chores**: (c_i or o_i^-)
- **Bundles**: Any subset $X \subseteq M$ of items. For any bundle X, let $X^{i+} := X \cap G_i$ and $X^{i-} := X \cap C_i$.
- $\succ_i (k)$: k^{th} ranked item in the preference ordering of i
- $\succ_i (k, S)$: k^{th} ranked item in the bundle S according to the preference order of i
  - $\succ_i (k) = \succ_i (k, M)$
- $> := (>_1, >_2, \cdots, >_n)$ is a set of preferences over bundles such that $>_i$ denotes the preference ordering over 2^M for the i^{th} agent, induced by $\succ_i$

Mathematical Abstraction

We define some additional notations:

- **Goods**: (g_i or o_i^+)
- **Chores**: (c_i or o_i^-)
- **Bundles**: Any subset $X \subseteq M$ of items. For any bundle X, let $X^{i+} := X \cap G_i$ and $X^{i-} := X \cap C_i$.
- $\succ_i (k)$: k^{th} ranked item in the preference ordering of i
- $\succ_i (k, S)$: k^{th} ranked item in the bundle S according to the preference order of i
  - $\succ_i (k) = \succ_i (k, M)$
- $> := (>_1, >_2, \cdots, >_n)$ is a set of preferences over bundles such that $>_i$ denotes the preference ordering over 2^M for the i^{th} agent, induced by $\succ_i$

Mathematical Abstraction

We define some additional notations:

- **Goods**: (g_i or o_i^+)
- **Chores**: (c_i or o_i^-)
- **Bundles**: Any subset $X \subseteq M$ of items. For any bundle X, let $X^{i+} := X \cap G_i$ and $X^{i-} := X \cap C_i$.
- $\succ_i (k)$: k^{th} ranked item in the preference ordering of i
- $\succ_i (k, S)$: k^{th} ranked item in the bundle S according to the preference order of i
  - $\succ_i (k) = \succ_i (k, M)$
- $>:= (>_1, >_2, \cdots, >_n)$ is a set of preferences over bundles such that $>_i$ denotes the preference ordering over 2^M for the i^{th} agent, induced by $\succ_i$

Mathematical Abstraction

We define some additional notations:

- **Goods**: (g_i or o_i^+)
- **Chores**: (c_i or o_i^-)
- **Bundles**: Any subset $X \subseteq M$ of items. For any bundle X, let $X^{i+} := X \cap G_i$ and $X^{i-} := X \cap C_i$.
- $\succ_i (k)$: k^{th} ranked item in the preference ordering of i
- $\succ_i (k, S)$: k^{th} ranked item in the bundle S according to the preference order of i
  - $\succ_i (k) = \succ_i (k, M)$
- $> := (>_1, >_2, \cdots, >_n)$ is a set of preferences over bundles such that $>_i$ denotes the preference ordering over 2^M for the i^{th} agent, induced by $\succ_i$

Mathematical Abstraction

We define some additional subclasses:

- Objectivity over items: $\forall i, j \in N, G_i = G_j, C_i = C_j$
- Subjectivity over items
- Goods only: $\forall i \in N, G_i = M$
- Chores only: $\forall i \in N, C_i = M$

Mathematical Abstraction

We define some additional subclasses:

- Objectivity over items: $\forall i, j \in N, G_i = G_j, C_i = C_j$
- Subjectivity over items
- Goods only: $\forall i \in N, G_i = M$
- Chores only: $\forall i \in N, C_i = M$

Mathematical Abstraction

We define some additional subclasses:

- Objectivity over items: $\forall i, j \in N, G_i = G_j, C_i = C_j$
- Subjectivity over items
- Goods only: $\forall i \in N, G_i = M$
- Chores only: $\forall i \in N, C_i = M$

Mathematical Abstraction

We define some additional subclasses:

- Objectivity over items: $\forall i, j \in N, G_i = G_j, C_i = C_j$
- Subjectivity over items
- Goods only: $\forall i \in N, G_i = M$
- Chores only: $\forall i \in N, C_i = M$

Mathematical Abstraction

We define some additional subclasses:

- Objectivity over items: $\forall i, j \in N, G_i = G_j, C_i = C_j$
- Subjectivity over items
- Goods only: $\forall i \in N, G_i = M$
- Chores only: $\forall i \in N, C_i = M$

Lexicographic Preferences

- Goods: Prefer to *have* them
- Chores: Prefer to *not have* them
- A preference, $\succ$, over M thus induces a preference, $>$, over 2^M.
- Example: $c_1^- \succ g_2^+ \succ c_3^- \succ g_4^+$
- Sample allocations:

$$\{g_2^+, g_4^+\} >_i \{g_2^+, c_3^-, g_4^+\} >_i \{g_2^+, c_3^-\} >_i \{c_1^-, g_2^+\}$$

Lexicographic Preferences

- Goods: Prefer to *have* them
- Chores: Prefer to *not have* them
- A preference, $\succ$, over M thus induces a preference, $>$, over 2^M.
- Example: $c_1^- \succ g_2^+ \succ c_3^- \succ g_4^+$
- Sample allocations:

$$\{g_2^+, g_4^+\} >_i \{g_2^+, c_3^-, g_4^+\} >_i \{g_2^+, c_3^-\} >_i \{c_1^-, g_2^+\}$$

$$\{c_1^-, g_2^+\} \ ? \ \{c_3^-, g_4^+\}$$

Lexicographic Preferences

- Goods: Prefer to *have* them
- Chores: Prefer to *not have* them
- A preference, $\succ$, over M thus induces a preference, $>$, over 2^M.
- Example: $c_1^- \succ g_2^+ \succ c_3^- \succ g_4^+$
- Sample allocations:

$$\{g_2^+, g_4^+\} >_i \{g_2^+, c_3^-, g_4^+\} >_i \{g_2^+, c_3^-\} >_i \{c_1^-, g_2^+\}$$

$$\{c_1^-, g_2^+\} \prec \{c_3^-, g_4^+\}$$

Notions of fairness

- Envy-free (EF): $\forall i, j \in N, A_i \succ_i A_j$
- Envy-free upto one item (EF1): $\forall i, j \in N$ such that $A_i^{i-} \cup A_j^{i+} \neq \Phi, \exists o \in A_i^{i-} \cup A_j^{i+}$ such that either $A_i \succ_i A_j \backslash \{o\}$ or $A_i \backslash \{o\} \succ_i A_j$
    - Essentially, every agent becomes envy free by removing **some chore** from their bundle or **some good** from the other agent's bundle
- Envy-free upto any item (EFX): $\forall i, j \in N$ such that $A_i^{i-} \cup A_j^{i+} \neq \Phi, \forall o \in A_i^{i-} \cup A_j^{i+}$ it must be that:
    - if $o \in A_j^{i+}$, $A_i \succ_i A_j \backslash \{o\}$
    - if $o \in A_i^{i-}$, $A_i \backslash \{o\} \succ_i A_j$
    - Essentially, every agent becomes envy free by removing **any one chore** from their bundle or **any one good** from the other agent's bundle
- Increasing strength: EF < EFX < EF1

Notions of fairness

- Envy-free (EF): $\forall i, j \in N, A_i \succ_i A_j$
- Envy-free upto one item (EF1): $\forall i, j \in N$ such that $A_i^{i-} \cup A_j^{i+} \neq \Phi, \exists o \in A_i^{i-} \cup A_j^{i+}$ such that either $A_i \succ_i A_j \backslash \{o\}$ or $A_i \backslash \{o\} \succ_i A_j$
  - Essentially, every agent becomes envy free by removing **some chore** from their bundle or **some good** from the other agent's bundle
- Envy-free upto any item (EFX): $\forall i, j \in N$ such that $A_i^{i-} \cup A_j^{i+} \neq \Phi, \forall o \in A_i^{i-} \cup A_j^{i+}$ it must be that:
  - if $o \in A_j^{i+}$, $A_i \succ_i A_j \backslash \{o\}$
  - if $o \in A_i^{i-}$, $A_i \backslash \{o\} \succ_i A_j$
  - Essentially, every agent becomes envy free by removing **any one chore** from their bundle or **any one good** from the other agent's bundle
- Increasing strength: EF < EFX < EF1

Notions of fairness

- Envy-free (EF): $\forall i, j \in N, A_i \succ_i A_j$
- Envy-free upto one item (EF1): $\forall i, j \in N$ such that $A_i^{i-} \cup A_j^{i+} \neq \Phi, \exists o \in A_i^{i-} \cup A_j^{i+}$ such that either $A_i \succ_i A_j \backslash \{o\}$ or $A_i \backslash \{o\} \succ_i A_j$
  - Essentially, every agent becomes envy free by removing **some chore** from their bundle or **some good** from the other agent's bundle
- Envy-free upto any item (EFX): $\forall i, j \in N$ such that $A_i^{i-} \cup A_j^{i+} \neq \Phi, \forall o \in A_i^{i-} \cup A_j^{i+}$ it must be that:
  - if $o \in A_j^{i+}$, $A_i \succ_i A_j \backslash \{o\}$
  - if $o \in A_i^{i-}$, $A_i \backslash \{o\} \succ_i A_j$
  - Essentially, every agent becomes envy free by removing **any one chore** from their bundle or **any one good** from the other agent's bundle
- Increasing strength: EF < EFX < EF1

Notions of fairness

- Envy-free (EF): $\forall i, j \in N, A_i \succ_i A_j$
- Envy-free upto one item (EF1): $\forall i, j \in N$ such that $A_i^{i-} \cup A_j^{i+} \neq \Phi, \exists o \in A_i^{i-} \cup A_j^{i+}$ such that either $A_i \succ_i A_j \backslash \{o\}$ or $A_i \backslash \{o\} \succ_i A_j$
  - Essentially, every agent becomes envy free by removing **some chore** from their bundle or **some good** from the other agent's bundle
- Envy-free upto any item (EFX): $\forall i, j \in N$ such that $A_i^{i-} \cup A_j^{i+} \neq \Phi, \forall o \in A_i^{i-} \cup A_j^{i+}$ it must be that:
  - if $o \in A_j^{i+}$, $A_i \succ_i A_j \backslash \{o\}$
  - if $o \in A_i^{i-}$, $A_i \backslash \{o\} \succ_i A_j$
  - Essentially, every agent becomes envy free by removing **any one chore** from their bundle or **any one good** from the other agent's bundle
- Increasing strength: EF $<$ EFX $<$ EF1

Notions of fairness

- Maximin Share (MMS):
  $MMS_i := \max_P \min_i\{P_1, P_2, \ldots, P_n\}$ – comparison w.r.t. $>_i$
  Intuition: Cake cutting problem/Security
  A satisfies MMS if $\forall i \in N, A_i \geq_i MMS_i$
- Pareto Optimality (PO): A is PO if $\nexists B$ s.t. $\forall i \in N, B_i \geq_i A_i$ and $\exists j \in N, B_j >_j A_j$

Notions of fairness

- Maximin Share (MMS):
  $MMS_i := \max_P \min_i\{P_1, P_2, \ldots, P_n\}$ – comparison w.r.t. $>_i$
  Intuition: Cake cutting problem/Security
  A satisfies MMS if $\forall i \in N, A_i \geq_i MMS_i$
- Pareto Optimality (PO): A is PO if $\nexists B$ s.t. $\forall i \in N, B_i \geq_i A_i$ and $\exists j \in N, B_j >_j A_j$

Notions of fairness

- Maximin Share (MMS):
  $MMS_i := \max_P \min_i\{P_1, P_2, \ldots, P_n\}$ – comparison w.r.t. $>_i$
  Intuition: Cake cutting problem/Security
  A satisfies MMS if $\forall i \in N, A_i \geq_i MMS_i$
- Pareto Optimality (PO): A is PO if $\nexists B$ s.t. $\forall i \in N, B_i \geq_i A_i$ and $\exists j \in N, B_j >_j A_j$

Notions of fairness

- Maximin Share (MMS):
  $MMS_i := \max_P \min_i \{P_1, P_2, \ldots, P_n\}$ – comparison w.r.t. $>_i$
  Intuition: Cake cutting problem/Security
  A satisfies MMS if $\forall i \in N, A_i \geq_i MMS_i$
- Pareto Optimality (PO): A is PO if $\nexists B$ s.t. $\forall i \in N, B_i \geq_i A_i$ and $\exists j \in N, B_j >_j A_j$

Results on Non-existence of EFX

- **EFX may not exist** for objective mixed items with lexicographic preferences.
- **Proof by counterexample**: Consider an objective mixed items instance with four agents and 7 objects.This is an objective instance, each item is either a good for all agents or a chore for all agents. o_1 is a good for all agents and all the other objects are chores.
  - Agents 1 and 2 have the same preference order, as do agents 3 and 4:

$$1,2 : o_2^- \succ o_3^- \succ o_4^- \succ o_1^+ \succ o_5^- \succ o_6^- \succ o_7^-$$
$$3,4 : o_5^- \succ o_6^- \succ o_7^- \succ o_1^+ \succ o_2^- \succ o_3^- \succ o_4^-$$

  - Consider, wlog, that the good o_1^+ is allocated to agent 1, then we have 3 possible allocations.
  - **Case 1**: If $A_1 \cap \{o_2^-, o_3^-, o_4^-\} \neq \Phi$
    A_2 must be empty, otherwise, regardless of what agent 2 gets, it will prefer A_1 even after removing a chore from A_2. Thus, $\{o_5^-, o_6^-, o_7^-\}$ must be distributed among A_3, A_4. Whoever gets 2 or more chores out of these, prefers empty A_2 even after one chore is removed. Thus, EFX is not possible.

Results on Non-existence of EFX

- **EFX may not exist** for objective mixed items with lexicographic preferences.
- **Proof by counterexample**: Consider an objective mixed items instance with four agents and 7 objects.This is an objective instance, each item is either a good for all agents or a chore for all agents. o_1 is a good for all agents and all the other objects are chores.
  - Agents 1 and 2 have the same preference order, as do agents 3 and 4:
    $$1, 2 : o_2^- \succ o_3^- \succ o_4^- \succ o_1^+ \succ o_5^- \succ o_6^- \succ o_7^-$$
    $$3, 4 : o_5^- \succ o_6^- \succ o_7^- \succ o_1^+ \succ o_2^- \succ o_3^- \succ o_4^-$$
  - Consider, wlog, that the good o_1^+ is allocated to agent 1, then we have 3 possible allocations.
  - **Case 1**: If $A_1 \cap \{o_2^-, o_3^-, o_4^-\} \neq \Phi$
    A_2 must be empty, otherwise, regardless of what agent 2 gets, it will prefer A_1 even after removing a chore from A_2. Thus, $\{o_5^-, o_6^-, o_7^-\}$ must be distributed among A_3, A_4. Whoever gets 2 or more chores out of these, prefers empty A_2 even after one chore is removed. Thus, EFX is not possible.

Results on Non-existence of EFX

- **EFX may not exist** for objective mixed items with lexicographic preferences.
- **Proof by counterexample**: Consider an objective mixed items instance with four agents and 7 objects.This is an objective instance, each item is either a good for all agents or a chore for all agents. o_1 is a good for all agents and all the other objects are chores.
  - Agents 1 and 2 have the same preference order, as do agents 3 and 4:

$$1,2 : o_2^- \succ o_3^- \succ o_4^- \succ o_1^+ \succ o_5^- \succ o_6^- \succ o_7^-$$
$$3,4 : o_5^- \succ o_6^- \succ o_7^- \succ o_1^+ \succ o_2^- \succ o_3^- \succ o_4^-$$

  - Consider, wlog, that the good o_1^+ is allocated to agent 1, then we have 3 possible allocations.
  - **Case 1**: If $A_1 \cap \{o_2^-, o_3^-, o_4^-\} \neq \Phi$
    A_2 must be empty, otherwise, regardless of what agent 2 gets, it will prefer A_1 even after removing a chore from A_2. Thus, $\{o_5^-, o_6^-, o_7^-\}$ must be distributed among A_3, A_4. Whoever gets 2 or more chores out of these, prefers empty A_2 even after one chore is removed. Thus, EFX is not possible.

Results on Non-existence of EFX

- **EFX may not exist** for objective mixed items with lexicographic preferences.
- **Proof by counterexample**: Consider an objective mixed items instance with four agents and 7 objects.This is an objective instance, each item is either a good for all agents or a chore for all agents. o_1 is a good for all agents and all the other objects are chores.
  - Agents 1 and 2 have the same preference order, as do agents 3 and 4:
    $$1,2 : o_2^- \succ o_3^- \succ o_4^- \succ o_1^+ \succ o_5^- \succ o_6^- \succ o_7^-$$
    $$3,4 : o_5^- \succ o_6^- \succ o_7^- \succ o_1^+ \succ o_2^- \succ o_3^- \succ o_4^-$$
  - Consider, wlog, that the good o_1^+ is allocated to agent 1, then we have 3 possible allocations.
  - **Case 1**: If $A_1 \cap \{o_2^-, o_3^-, o_4^-\} \neq \Phi$
    A_2 must be empty, otherwise, regardless of what agent 2 gets, it will prefer A_1 even after removing a chore from A_2. Thus, $\{o_5^-, o_6^-, o_7^-\}$ must be distributed among A_3, A_4. Whoever gets 2 or more chores out of these, prefers empty A_2 even after one chore is removed. Thus, EFX is not possible.

Results on Non-existence of EFX

- **EFX may not exist** for objective mixed items with lexicographic preferences.
- **Proof by counterexample**: Consider an objective mixed items instance with four agents and 7 objects.This is an objective instance, each item is either a good for all agents or a chore for all agents. o_1 is a good for all agents and all the other objects are chores.
  - Agents 1 and 2 have the same preference order, as do agents 3 and 4:
    $$1, 2 : o_2^- \succ o_3^- \succ o_4^- \succ o_1^+ \succ o_5^- \succ o_6^- \succ o_7^-$$
    $$3, 4 : o_5^- \succ o_6^- \succ o_7^- \succ o_1^+ \succ o_2^- \succ o_3^- \succ o_4^-$$
  - Consider, wlog, that the good o_1^+ is allocated to agent 1, then we have 3 possible allocations.
  - **Case 1**: If $A_1 \cap \{o_2^-, o_3^-, o_4^-\} \neq \Phi$
    A_2 must be empty, otherwise, regardless of what agent 2 gets, it will prefer A_1 even after removing a chore from A_2. Thus, $\{o_5^-, o_6^-, o_7^-\}$ must be distributed among A_3, A_4. Whoever gets 2 or more chores out of these, prefers empty A_2 even after one chore is removed. Thus, EFX is not possible.

Results on non-existence of EFX

- **Case 2**: If $A_1 \cap \{o_5^-, o_6^-, o_7^-\} \neq \Phi$
  If items are assigned to agent 3 or 4, they will envy A_1 even after one chore is removed, since the good o_1^+ is the most important item in A_1 for them. Thus, A_3 and A_4 must be empty. Thus, $\{o_5^-, o_6^-, o_7^-\}$ must all be in A_2. Even after removing one chore from A_2, agent 2 will envy the empty bundle A_3. Thus, EFX is not possible.
- **Case 3**: If $A_1 \cap \{o_2^-, o_3^-, o_4^-\} \neq \Phi$ and $A_1 \cap \{o_5^-, o_6^-, o_7^-\} \neq \Phi$.
  Choose any $x \in A_1 \cap \{o_2^-, o_3^-, o_4^-\}$ and $y \in A_1 \cap \{o_5^-, o_6^-, o_7^-\}$. For EFX, agent 1 should not be envious of any other bundle after y is removed. Which means all 3 bundles must have at least one chore that is ranked higher than x in the preference order of agent 1. However, there exist at most 2 such chores. Thus, EFX is not possible.

Results on non-existence of EFX

- **Case 2**: If $A_1 \cap \{o_5^-, o_6^-, o_7^-\} \neq \Phi$
  If items are assigned to agent 3 or 4, they will envy A_1 even after one chore is removed, since the good o_1^+ is the most important item in A_1 for them. Thus, A_3 and A_4 must be empty. Thus, $\{o_5^-, o_6^-, o_7^-\}$ must all be in A_2. Even after removing one chore from A_2, agent 2 will envy the empty bundle A_3. Thus, EFX is not possible.
- **Case 3**: If $A_1 \cap \{o_2^-, o_3^-, o_4^-\} \neq \Phi$ and $A_1 \cap \{o_5^-, o_6^-, o_7^-\} \neq \Phi$.
  Choose any $x \in A_1 \cap \{o_2^-, o_3^-, o_4^-\}$ and $y \in A_1 \cap \{o_5^-, o_6^-, o_7^-\}$. For EFX, agent 1 should not be envious of any other bundle after y is removed. Which means all 3 bundles must have at least one chore that is ranked higher than x in the preference order of agent 1. However, there exist at most 2 such chores. Thus, EFX is not possible.

Results on EFX+PO

- Existence of EFX not guaranteed in general
- Can we categorise classes of games that admit an EFX?
  1. $\exists i \in N : \succ_i (1) \in G_i$
  2. $\cap_i C_i = \Phi$

EFX+PO: Algorithm

Class: $\exists i \in N : \succ_i (1) \in G_i$

Idea: Satisfy the current agent with the highest-ranking good. Push the remaining common chores under the current agent's umbrella to get rid of them.

Sketch:

1. (WLOG) Assume for $1 \in N : \succ_1 (1) \in G_1$
2. Allocate $\succ_1 (1)$ to 1
3. Allocate "*common chores*" to 1
4. Identify j with highest-ranking good
5. Allocate the good to j as well as the "*common chores*"

Observation: $A_j \subseteq G_j$

EFX+PO: Algorithm

Class: $\exists i \in N : \succ_i (1) \in G_i$
Idea: Satisfy the current agent with the highest-ranking good. Push the remaining common chores under the current agent's umbrella to get rid of them.
Sketch:

1. (WLOG) Assume for $1 \in N : \succ_1 (1) \in G_1$
2. Allocate $\succ_1 (1)$ to 1
3. Allocate "*common chores*" to 1
4. Identify j with highest-ranking good
5. Allocate the good to j as well as the "*common chores*"

Observation: $A_j \subseteq G_j$

EFX+PO: Algorithm

Class: $\exists i \in N : \succ_i (1) \in G_i$

Idea: Satisfy the current agent with the highest-ranking good. Push the remaining common chores under the current agent's umbrella to get rid of them.

Sketch:

1. (WLOG) Assume for $1 \in N : \succ_1 (1) \in G_1$
2. Allocate $\succ_1 (1)$ to 1
3. Allocate "*common chores*" to 1
4. Identify j with highest-ranking good
5. Allocate the good to j as well as the "*common chores*"

Observation: $A_j \subseteq G_j$

EFX+PO: Algorithm's Proof

Class: $\exists i \in N : \succ_i (1) \in G_i$

EFX – Inductive

- 1 is not envious of anyone else
- Reduced instance(s) has no common chores
- Let $i (\neq 1)$ be the agent in the current round, then
  - $A_i \subseteq G_i$
  - $\Rightarrow A_i \cap C_i = \Phi$
  - Need only check $A_j \cap G_i$ for EFX
- i is not envious of anyone that came before – $|A_{j<i} \cap G_i| \leq 1$
- i is not envious of anyone that comes after –
  - i received highest-ranking good (amongst the remaining G_i)
  - $A_i \cap C_i = \Phi$ and need only check goods for EFX

EFX+PO: Algorithm's Proof

Class: $\exists i \in N : \succ_i (1) \in G_i$

EFX – Inductive

- 1 is not envious of anyone else
- Reduced instance(s) has no common chores
- Let $i(\neq 1)$ be the agent in the current round, then
  - $A_i \subseteq G_i$
  - $\Rightarrow A_i \cap C_i = \Phi$
  - Need only check $A_j \cap G_i$ for EFX
- i is not envious of anyone that came before – $|A_{j<i} \cap G_i| \leq 1$
- i is not envious of anyone that comes after –
  - i received highest-ranking good (amongst the remaining G_i)
  - $A_i \cap C_i = \Phi$ and need only check goods for EFX

EFX+PO: Algorithm's Proof

Class: $\exists i \in N :\succ_i (1) \in G_i$

EFX – Inductive

- 1 is not envious of anyone else
- Reduced instance(s) has no common chores
- Let $i (\neq 1)$ be the agent in the current round, then
  - $A_i \subseteq G_i$
  - $\Rightarrow A_i \cap C_i = \Phi$
  - Need only check $A_j \cap G_i$ for EFX
- i is not envious of anyone that came before – $|A_{j<i} \cap G_i| \leq 1$
- i is not envious of anyone that comes after –
  - i received highest-ranking good (amongst the remaining G_i)
  - $A_i \cap C_i = \Phi$ and need only check goods for EFX

EFX+PO: Algorithm's Proof

Class: $\exists i \in N : \succ_i (1) \in G_i$
PO – Contradiction
Let B pareto dominate A (returned by the algorithm)

- $\succ_i (1, A_i) \in B_i$
  - Need to compensate i otherwise
  - True for $i = 1$ since $\succ_i (1, A_i) = \succ_i (1) \in G_1$
  - i picks highest-ranking good remaining
  - Inductively, $\forall j < i : \succ_j (1, A_j) \in B_j$. The other items in A_j are chores for i
  - $\forall j > i, j$ receives less valuable items or chores
  - i does not have chores – cannot be compensated

EFX+PO: Algorithm's Proof

Class: $\exists i \in N : \succ_i (1) \in G_i$

PO – Contradiction

Let B pareto dominate A (returned by the algorithm)

- $\succ_i (1, A_i) \in B_i$
  - Need to compensate i otherwise
  - True for $i = 1$ since $\succ_i (1, A_i) = \succ_i (1) \in G_1$
  - i picks highest-ranking good remaining
  - Inductively, $\forall j < i : \succ_j (1, A_j) \in B_j$. The other items in A_j are chores for i
  - $\forall j > i, j$ receives less valuable items or chores
  - i does not have chores – cannot be compensated

EFX+PO: Algorithm's Proof

Class: $\exists i \in N :\succ_i (1) \in G_i$

PO – Contradiction

Let B pareto dominate A (returned by the algorithm)

- $\succ_i (1, A_i) \in B_i$
  - Need to compensate i otherwise
  - True for $i = 1$ since $\succ_i (1, A_i) = \succ_i (1) \in G_1$
  - i picks highest-ranking good remaining
  - Inductively, $\forall j < i :\succ_j (1, A_j) \in B_j$. The other items in A_j are chores for i
  - $\forall j > i, j$ receives less valuable items or chores
  - i does not have chores – cannot be compensated

EFX+PO: Algorithm's Proof

Class: $\exists i \in N : \succ_i (1) \in G_i$

PO – Contradiction

Let B pareto dominate A (returned by the algorithm)

- $\succ_i (1, A_i) \in B_i$
  - Need to compensate i otherwise
  - True for $i = 1$ since $\succ_i (1, A_i) = \succ_i (1) \in G_1$
  - i picks highest-ranking good remaining
  - Inductively, $\forall j < i : \succ_j (1, A_j) \in B_j$. The other items in A_j are chores for i
  - $\forall j > i, j$ receives less valuable items or chores
  - i does not have chores – cannot be compensated

EFX+PO: Algorithm's Proof

Class: $\exists i \in N : \succ_i (1) \in G_i$

PO – Contradiction

Let B pareto dominate A (returned by the algorithm)

- $A_i \subseteq B_i$
    - No chores to lose for $i > 1$
    - Loss of a good for $i \geq 1$ cannot be compensated by a:
        1. Good from $j > i$ – (reverse) induction
        2. Good from $j < i$ – only possibility is $\succ_j (1, A_j)$, previous result
    - 1 cannot lose anything from A_1 – either $\succ_1 (1) \in G_1$ or the item is a chore for j receiving it $\Rightarrow B_j <_j A_j$
- $\Rightarrow \forall i \in N, A_i = B_i \Rightarrow A = B \Rightarrow\Leftarrow$

EFX+PO: Algorithm's Proof

Class: $\exists i \in N : \succ_i (1) \in G_i$

PO – Contradiction

Let B pareto dominate A (returned by the algorithm)

- $A_i \subseteq B_i$
  - No chores to lose for $i > 1$
  - Loss of a good for $i \geq 1$ cannot be compensated by a:
    1. Good from $j > i$ – (reverse) induction
    2. Good from $j < i$ – only possibility is $\succ_j (1, A_j)$, previous result
  - 1 cannot lose anything from A_1 – either $\succ_1 (1) \in G_1$ or the item is a chore for j receiving it $\Rightarrow B_j <_j A_j$
- $\Rightarrow \forall i \in N, A_i = B_i \Rightarrow A = B \Rightarrow\Leftarrow$

EFX+PO: Algorithm's Proof

Class: $\exists i \in N : \succ_i (1) \in G_i$

PO – Contradiction

Let B pareto dominate A (returned by the algorithm)

- $A_i \subseteq B_i$
  - No chores to lose for $i > 1$
  - Loss of a good for $i \geq 1$ cannot be compensated by a:
    1. Good from $j > i$ – (reverse) induction
    2. Good from $j < i$ – only possibility is $\succ_j (1, A_j)$, previous result
  - 1 cannot lose anything from A_1 – either $\succ_1 (1) \in G_1$ or the item is a chore for j receiving it $\Rightarrow B_j <_j A_j$
- $\Rightarrow \forall i \in N, A_i = B_i \Rightarrow A = B \Rightarrow\Leftarrow$

EFX+PO: Algorithm's Proof

Class: $\exists i \in N : \succ_i (1) \in G_i$

PO – Contradiction

Let B pareto dominate A (returned by the algorithm)

- $A_i \subseteq B_i$
    - No chores to lose for $i > 1$
    - Loss of a good for $i \geq 1$ cannot be compensated by a:
        1. Good from $j > i$ – (reverse) induction
        2. Good from $j < i$ – only possibility is $\succ_j (1, A_j)$, previous result
    - 1 cannot lose anything from A_1 – either $\succ_1 (1) \in G_1$ or the item is a chore for j receiving it $\Rightarrow B_j <_j A_j$
- $\Rightarrow \forall i \in N, A_i = B_i \Rightarrow A = B \Rightarrow\Leftarrow$

EFX+PO: Algorithm

Class: $\cap_i C_i = \Phi$
Idea: Same as the first algorithm*
Sketch:

1. Identify j with highest-ranking good
2. Allocate the good to j as well as the "*common chores*"
3. $A_j \subseteq G_j$

Results on MMS

- Maximin Share (MMS):
  $MMS_i := \max_P \min_i\{P_1, P_2, \ldots, P_n\}$ – comparison w.r.t. $>_i$
  Intuition: Cake cutting problem/Security
  A satisfies MMS if $\forall i \in N, A_i \geq_i MMS_i$
- Characterization of MMS_i
- EFX Allocation $\Rightarrow$ MMS
- MMS is in P

Results on MMS

- Maximin Share (MMS):
  $MMS_i := \max_P \min_i \{P_1, P_2, \dots, P_n\}$ – comparison w.r.t. $>_i$
  Intuition: Cake cutting problem/Security
  A satisfies MMS if $\forall i \in N, A_i \geq_i MMS_i$
- Characterization of MMS_i
- EFX Allocation $\Rightarrow$ MMS
- MMS is in P

Results on MMS

- Maximin Share (MMS):
  $MMS_i := \max_P \min_i \{P_1, P_2, \ldots, P_n\}$ – comparison w.r.t. $>_i$
  Intuition: Cake cutting problem/Security
  A satisfies MMS if $\forall i \in N, A_i \geq_i MMS_i$
- Characterization of MMS_i
- EFX Allocation $\Rightarrow$ MMS
- MMS is in P

Results on MMS

- Maximin Share (MMS):
  $MMS_i := \max_P \min_i \{P_1, P_2, \ldots, P_n\}$ – comparison w.r.t. $>_i$
  Intuition: Cake cutting problem/Security
  A satisfies MMS if $\forall i \in N, A_i \geq_i MMS_i$
- Characterization of MMS_i
- EFX Allocation $\Rightarrow$ MMS
- MMS is in P

Characterization of MMS_i

Intuition: Force i to make $|N|$ partititions of M; What is the worst bundle it can guarantee for itself?

Heuristics:

- H1: If it is guaranteed that a bundle **is not** the *worst* possible, make it as bad as possible – improve prospects of the *actual worst* bundle
- H2: If it is guaranteed that a bundle **is** the *worst* possible, make it as good as possible

Characterization of MMS_i

Intuition: Force i to make $|N|$ partititions of M; What is the worst bundle it can guarantee for itself?

Heuristics:

- H1: If it is guaranteed that a bundle **is not** the *worst* possible, make it as bad as possible – improve prospects of the *actual worst* bundle
- H2: If it is guaranteed that a bundle **is** the *worst* possible, make it as good as possible

Characterization of MMS_i

- $(\succ_i (1) \in G_i) \wedge (|G_i| \geq n) \Rightarrow MMS_i = G_i \setminus \succ_i ([n-1], G_i)$
- $(\succ_i (1) \in G_i) \wedge (|G_i| < n) \Rightarrow MMS_i = \Phi$
- $(\succ_i (1) \in C_i) \Rightarrow MMS_i = \{\succ_i (1)\} \cup G_i$
  - $MMS_i \stackrel{?}{=} \Phi$

Characterization of MMS_i

- $(\succ_i (1) \in G_i) \wedge (|G_i| \geq n) \Rightarrow MMS_i = G_i \setminus \succ_i ([n-1], G_i)$
- $(\succ_i (1) \in G_i) \wedge (|G_i| < n) \Rightarrow MMS_i = \Phi$
- $(\succ_i (1) \in C_i) \Rightarrow MMS_i = \{\succ_i (1)\} \cup G_i$
  - $MMS_i \stackrel{?}{=} \Phi$

Characterization of MMS_i

- $(\succ_i (1) \in G_i) \wedge (|G_i| \geq n) \Rightarrow MMS_i = G_i \setminus \succ_i ([n-1], G_i)$
- $(\succ_i (1) \in G_i) \wedge (|G_i| < n) \Rightarrow MMS_i = \Phi$
- $(\succ_i (1) \in C_i) \Rightarrow MMS_i = \{\succ_i (1)\} \cup G_i$
    - $MMS_i \stackrel{?}{=} \Phi$

EFX $\Rightarrow$ MMS

Let A be an EFX allocation that does not satisfy MMS for i. Then, we arrive at a contradiction (specifically, A is not EFX) in each of the following cases, inspired by the characterization of MMS:

1. $\succ_i (1) \in C_i$
2. $\succ_i (1) \in G_i, |G_i| < n$
3. $\succ_i (1) \in G_i, |G_i| \geq n, A_i \cap C_i \neq \Phi$
4. $\succ_i (1) \in G_i, |G_i| \geq n, A_i \cap C_i = \Phi$

Since A does not satisfy MMS for i, we have $MMS_i >_i A_i$

EFX $\Rightarrow$ MMS

Case 1: $\succ_i(1) \in C_i$
$\Rightarrow MMS_i = \{\succ_i(1)\} \cup G_i$

- Observation: $\succ_i(1) \in A_i$
- Observation: i prefers any bundle without $\succ_i(1)$
- If A_i misses out on some good g,
  - Some $j \neq i$ receives g in A_j
  - $A_j \setminus \{g\} >_i A_i$
- *If $G_i \subset A_i$
  - Since $A_i <_i MMS_i$, A_i must contain a chore other than $\succ_i(1)$, say c
  - $A_j >_i A_i \setminus \{c\}$

EFX $\Rightarrow$ MMS

Case 1: $\succ_i (1) \in C_i$
$\Rightarrow MMS_i = \{\succ_i (1)\} \cup G_i$

- Observation: $\succ_i (1) \in A_i$
- Observation: i prefers any bundle without $\succ_i (1)$
- If A_i misses out on some good g,
  - Some $j \neq i$ receives g in A_j
  - $A_j \backslash \{g\} >_i A_i$
- *If $G_i \subset A_i$
  - Since $A_i <_i MMS_i$, A_i must contain a chore other than $\succ_i (1)$, say c
  - $A_j >_i A_i \backslash \{c\}$

EFX $\Rightarrow$ MMS

Case 1: $\succ_i (1) \in C_i$
$\Rightarrow MMS_i = \{\succ_i (1)\} \cup G_i$

- Observation: $\succ_i (1) \in A_i$
- Observation: i prefers any bundle without $\succ_i (1)$
- If A_i misses out on some good g,
  - Some $j \neq i$ receives g in A_j
  - $A_j \backslash \{g\} >_i A_i$
- *If $G_i \subset A_i$
  - Since $A_i <_i MMS_i, A_i$ must contain a chore other than $\succ_i (1)$, say c
  - $A_j >_i A_i \backslash \{c\}$

EFX $\Rightarrow$ MMS

Case 1: $\succ_i (1) \in C_i$
$\Rightarrow MMS_i = \{\succ_i (1)\} \cup G_i$

- Observation: $\succ_i (1) \in A_i$
- Observation: i prefers any bundle without $\succ_i (1)$
- If A_i misses out on some good g,
  - Some $j \neq i$ receives g in A_j
  - $A_j \backslash \{g\} >_i A_i$
- *If $G_i \subset A_i$
  - Since $A_i <_i MMS_i, A_i$ must contain a chore other than $\succ_i (1)$, say c
  - $A_j >_i A_i \backslash \{c\}$

EFX $\Rightarrow$ MMS

Case 2: $\succ_i (1) \in G_i, |G_i| < n$
$\Rightarrow MMS_i = \Phi$

- Let $g := \succ_i (1)$ and $c := \succ_i (1, A_i)$
- $c \in C_i, g \notin A_i$ as $\Phi >_i A_i$
- Some $j \neq i$ receives g in A_j
- $A_j >_i A_i \backslash \{c\}$

EFX $\Rightarrow$ MMS

Case 2: $\succ_i (1) \in G_i, |G_i| < n$
$\Rightarrow MMS_i = \Phi$

- Let $g := \succ_i (1)$ and $c := \succ_i (1, A_i)$
- $c \in C_i, g \notin A_i$ as $\Phi >_i A_i$
- Some $j \neq i$ receives g in A_j
- $A_j >_i A_i \setminus \{c\}$

EFX $\Rightarrow$ MMS

Case 3: $\succ_i (1) \in G_i, |G_i| \geq n, A_i \cap C_i \neq \Phi$
$\Rightarrow MMS_i = G_i \setminus \succ_i ([n-1], G_i)$

- $\exists c \in A_i \cap C_i$
- Let $g := \succ_i (1)$
- $A_i <_i MMS_i \Rightarrow g \notin A_i$
- Some $j \neq i$ receives g in A_j
- $A_j >_i A_i \setminus \{c\}$

EFX $\Rightarrow$ MMS

Case 3: $\succ_i (1) \in G_i, |G_i| \geq n, A_i \cap C_i \neq \Phi$
$\Rightarrow MMS_i = G_i \setminus \succ_i ([n-1], G_i)$

- $\exists c \in A_i \cap C_i$
- Let $g := \succ_i (1)$
- $A_i <_i MMS_i \Rightarrow g \notin A_i$
- Some $j \neq i$ receives g in A_j
- $A_j >_i A_i \setminus \{c\}$

EFX $\Rightarrow$ MMS

Case 3: $\succ_i (1) \in G_i, |G_i| \geq n, A_i \cap C_i \neq \Phi$
$\Rightarrow MMS_i = G_i \setminus \succ_i ([n-1], G_i)$

- $\exists c \in A_i \cap C_i$
- Let $g := \succ_i (1)$
- $A_i <_i MMS_i \Rightarrow g \notin A_i$
- Some $j \neq i$ receives g in A_j
- $A_j >_i A_i \setminus \{c\}$

EFX $\Rightarrow$ MMS

*Case 4: $\succ_i (1) \in G_i, |G_i| \geq n, A_i \cap C_i = \Phi$
$\Rightarrow MMS_i = G_i \setminus \succ_i ([n-1], G_i)$

- "In particular, agent i does not receive any of its favorite $(n-1)$ goods under A"
- "Thus, there are n items, namely $o_1, \ldots, o_{n-1}$ and o', that are allocated among $(n-1)$ other agents"

EFX $\Rightarrow$ MMS

Counterexample

- $N = \{1, 2, 3\}$ and $M = \{o_1, o_2, o_3, o_4, o_5\}$
- $1 : o_1^+ \succ_1 o_2^- \succ_1 o_3^- \succ_1 o_4^+ \succ_1 o_5^+$
- $2 : \succ_2 (1) = o_2^+$, $3 : \succ_3 (1) = o_3^+$
- $MMS_1 = \{o_5^+\}$
- $A_1 = \phi, A_2 = \{o_1, o_2\}, A_3 = \{o_3, o_4, o_5\}$

- Allocation A is EFX
- Allocation A is **NOT** MMS; $MMS_1 >_1 A_1$

MMS always exists

The paper claims that *MMS* always exists (in mixed items), but we provide a counter-example:

Counterexample

- $N = \{1, 2, 3\}$ and $M = \{o_1^+, o_2^-, o_3^-, o_4^+, o_5^+\}$
- All agents have the same preference $\succ$, and the items are **objective**
- $o_1^+ \succ o_2^- \succ o_3^- \succ o_4^+ \succ o_5^+$
- $MMS_{1,2,3} = \{o_5^+\}$

- Under a complete allocation, some agent must get a chore o_3^- - that agent is MMS unsatisfied

MMS exists in a subclass

- However, the paper does manage to guarantee MMS for the following subclass:

SC_0

If an allocation instance $(N, M, G, C, \succ)$ has $\forall\, i \in N, \succ_i (1) \in C_i$, then an MMS allocation always exists.

- In essence, every agents' top-ranked item is a chore

MMS exists in a subclass

- However, the paper does manage to guarantee MMS for the following subclass:

SC_0

If an allocation instance $(N, M, G, C, \succ)$ has $\forall\, i \in N, \succ_i (1) \in C_i$, then an MMS allocation always exists.

- In essence, every agents' top-ranked item is a chore

MMS in SC_0

- In an SC_0 instance, each agents' MMS bundle contains their most hated chore i.e $\succ_i (1)$
- We create an allocation where at most one agent gets their most hated chore, and give the same agent all their goods - the MMS bundle
- C' is the set of all common chores - σ is an ordering of all the agents
- Allow agents to pick chores from C' according to σ - as long as $|C'| \geq 2$, an agent can ensure that they do not get their most hated chore
- $|C'| = 1$ is the only situation where an agent runs the risk of choosing their hated chore - give them all their goods in such an instance
- Allocate all other items to agents who considers that item as a good

MMS in SC_0

- In an SC_0 instance, each agents' MMS bundle contains their most hated chore i.e $\succ_i (1)$
- We create an allocation where at most one agent gets their most hated chore, and give the same agent all their goods - the MMS bundle
- C' is the set of all common chores - σ is an ordering of all the agents
- Allow agents to pick chores from C' according to σ - as long as $|C'| \geq 2$, an agent can ensure that they do not get their most hated chore
- $|C'| = 1$ is the only situation where an agent runs the risk of choosing their hated chore - give them all their goods in such an instance
- Allocate all other items to agents who considers that item as a good

MMS in SC_0

- In an SC_0 instance, each agents' MMS bundle contains their most hated chore i.e $\succ_i (1)$
- We create an allocation where at most one agent gets their most hated chore, and give the same agent all their goods - the MMS bundle
- C' is the set of all common chores - σ is an ordering of all the agents
- Allow agents to pick chores from C' according to σ - as long as $|C'| \geq 2$, an agent can ensure that they do not get their most hated chore
- $|C'| = 1$ is the only situation where an agent runs the risk of choosing their hated chore - give them all their goods in such an instance
- Allocate all other items to agents who considers that item as a good

MMS in SC_0

- In an SC_0 instance, each agents' MMS bundle contains their most hated chore i.e $\succ_i(1)$
- We create an allocation where at most one agent gets their most hated chore, and give the same agent all their goods - the MMS bundle
- C' is the set of all common chores - σ is an ordering of all the agents
- Allow agents to pick chores from C' according to σ - as long as $|C'| \geq 2$, an agent can ensure that they do not get their most hated chore
- $|C'| = 1$ is the only situation where an agent runs the risk of choosing their hated chore - give them all their goods in such an instance
- Allocate all other items to agents who considers that item as a good

MMS in SC_0

- In an SC_0 instance, each agents' MMS bundle contains their most hated chore i.e $\succ_i (1)$
- We create an allocation where at most one agent gets their most hated chore, and give the same agent all their goods - the MMS bundle
- C' is the set of all common chores - σ is an ordering of all the agents
- Allow agents to pick chores from C' according to σ - as long as $|C'| \geq 2$, an agent can ensure that they do not get their most hated chore
- $|C'| = 1$ is the only situation where an agent runs the risk of choosing their hated chore - give them all their goods in such an instance
- Allocate all other items to agents who considers that item as a good

MMS in SC_0

- In an SC_0 instance, each agents' MMS bundle contains their most hated chore i.e $\succ_i (1)$
- We create an allocation where at most one agent gets their most hated chore, and give the same agent all their goods - the MMS bundle
- C' is the set of all common chores - σ is an ordering of all the agents
- Allow agents to pick chores from C' according to σ - as long as $|C'| \geq 2$, an agent can ensure that they do not get their most hated chore
- $|C'| = 1$ is the only situation where an agent runs the risk of choosing their hated chore - give them all their goods in such an instance
- Allocate all other items to agents who considers that item as a good

Concluding Remarks

- EFX does not always exist
- EFX exists for some subclasses:
  - Some agent has a good at their highest priority
  - There are no common chores
- MMS always exists and is in P
  - MMS does not always exist
  - There exists a subclass for which an MMS allocation can be found in PTIME
- Future prospects: $\exists MMS \in P$?

But hey, that's just a theory . . .