Fair Allocation of Indivisible Goods

EF, EF1, EFX, MMS, Pareto Optimality, Proportionality, & Price of Fairness

Joseph Chuang-Chieh Lin

Department of Computer Science & Engineering,
National Taiwan Ocean University

Fall 2026

Source and Scope

- Main source: Amanatidis et al., "Fair division of indivisible goods: Recent progress and open questions," *Artificial Intelligence*, 2023.
- Scope of these slides: indivisible **goods** only.
- We omit the parts on chores and mixed manna.

Learning goals

- The main fairness notions.
- Basic algorithms.
- Existence and computational results.
- Price of Fairness.

Outline

Outline

Why Indivisible Goods?

- Many resources **cannot be split**: courses, conference rooms, tasks with positive value, books, GPUs, dorm rooms.
- Agents have heterogeneous preferences.
- Exact fairness can be impossible even in tiny instances.
  - Consider one ball and two agents, John and Chris, both of whom want the ball.

Basic instance & (additive) valuations

Discrete fair allocation instance

$$I = (N, M, v), \qquad N = \{1, 2, \dots, n\}, \quad M = \{g_1, g_2, \dots, g_m\}.$$

- Each agent i has a valuation function

$$v_i : 2^M \to \mathbb{R}_{\geq 0}.$$

- Standard assumptions for goods:

$$v_i(\emptyset) = 0, \qquad S \subseteq T \Rightarrow v_i(S) \leq v_i(T).$$

- In this lecture we mostly focus on **additive** valuations:

$$v_i(S) = \sum_{g \in S} v_i(g).$$

Allocation

Allocation

An allocation is a tuple $A = (A_1, \ldots, A_n)$ such that

$$A_i \cap A_j = \emptyset \quad (i \neq j), \qquad \bigcup_{i \in N} A_i = M.$$

Agent i receives bundle A_i.

$A_1 = \{g_2, g_5\}$	$A_2 = \{g_3, g_4\}$	$A_3 = \{g_1\}$
a_1	a_2	a_3

Valuation Table Representation

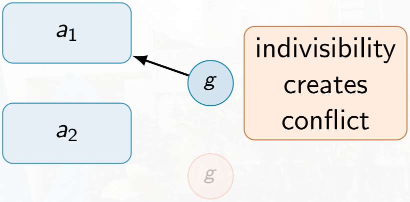
- For additive valuations, an instance can be represented by an $n \times m$ table.
- Entry (i, g) is the number $v_i(g)$.

	g_1	g_2	g_3	g_4	g_5
a_1	15	3	2	2	6
a_2	7	5	5	5	7
a_3	20	3	3	3	3

This instance will be reused to illustrate EF1, EFX, and MMS.

Why Exact Fairness Can Fail

- Two agents.
- One good g.
- Both agents value g positively.



Consequence

Whoever does not receive g obtains value 0 and envies the winner.

This is why relaxations such as EF1, EFX, and MMS are central.

Algorithmic Questions

Three recurring questions

1. **Existence:** Does a fair allocation always exist?
2. **Computation:** Can we find one in polynomial time?
3. **Compatibility:** Can we combine fairness with efficiency such as Pareto optimality?

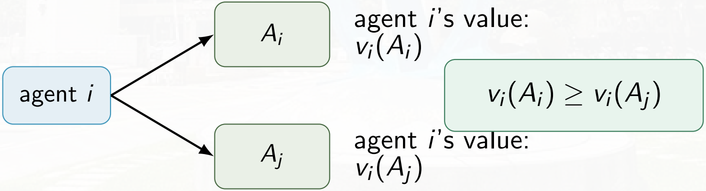
Outline

Envy-Freeness

Definition (Envy-freeness)

An allocation A is **envy-free** *(EF) if* $v_i(A_i) \geq v_i(A_j) \quad \forall i, j \in N.$

- Agent i compares her own bundle to agent j's bundle using i's valuation.
- EF is pairwise and very strong for indivisible goods.



Agent i compares both bundles using her own valuation v_i.

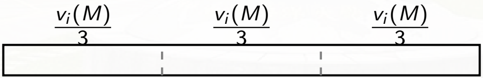
Proportionality

Definition (Proportionality)

An allocation A is **proportional** *(PROP) if*

$$v_i(A_i) \geq \frac{v_i(M)}{n} \qquad \forall i \in N.$$

- Agent i expects at least a $1/n$ fraction of her total value.
- PROP is an absolute guarantee, not a pairwise comparison.



value benchmark for agent i

EF Implies PROP

Proposition

For additive valuations over goods, every EF allocation is PROP.

Proof idea.

If A is EF, then for every agent i, $v_i(A_i) \geq v_i(A_j) \quad \forall j \in N$.

Summing over j gives

$$n \cdot v_i(A_i) \geq \sum_{j \in N} v_i(A_j) = v_i(M),$$

so $v_i(A_i) \geq v_i(M)/n$. $\square$

EF Implies PROP

Proposition

For additive valuations over goods, every EF allocation is PROP.

Proof idea.

If A is EF, then for every agent i, $v_i(A_i) \geq v_i(A_j) \quad \forall j \in N$.

Summing over j gives

$$n \cdot v_i(A_i) \geq \sum_{j \in N} v_i(A_j) = v_i(M),$$

so $v_i(A_i) \geq v_i(M)/n$. $\square$

The converse need not hold: PROP may allow envy.

Example: PROP Without EF

	g_1	g_2	g_3	g_4
a_1	10	6	6	8
a_2	10	5	5	10
a_3	10	0	10	10

Allocation B:

$$B_1 = \{g_1\}, \qquad B_2 = \{g_2, g_3\}, \qquad B_3 = \{g_4\}.$$

- Every agent gets value at least $v_i(M)/3 = 10$: hence PROP.
- But a_1 envies a_2 because $v_1(B_1) = 10 < 12 = v_1(B_2)$.

EF1: Envy-Free Up to One Good

Definition (EF1)

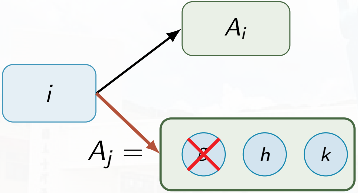
An allocation A is **envy-free up to one good** *if, for every $i, j \in N$,*

$$v_i(A_i) \geq v_i(A_j \setminus \{g\})$$

for **some** *good $g \in A_j$.*

- If i envies j, removing one suitable good from j's bundle eliminates the envy.
- EF1 is achievable by simple algorithms.

EF1 as a Diagram



$$v_i(A_i) \geq v_i(A_j \setminus \{g\})$$

possible envy toward A_j

remove one suitable good from A_j

Key point

The removed good may be the most valuable good in A_j from i's perspective.

EFX: Envy-Free Up to Any Good

Definition (EFX)

An allocation A is **envy-free up to any good** *if, for every $i, j \in N$,*

$$v_i(A_i) \geq v_i(A_j \setminus \{g\})$$

for **every** *good $g \in A_j$.*

- EFX is a stronger relaxation of EF than EF1.
- Intuition: even removing the least important good in A_j should eliminate i's envy.

EFX: Envy-Free Up to Any Good

Definition (EFX)

An allocation A is **envy-free up to any good** *if, for every $i, j \in N$,*

$$v_i(A_i) \geq v_i(A_j \setminus \{g\})$$

for **every** *good $g \in A_j$.*

- EFX is a stronger relaxation of EF than EF1.
- Intuition: even removing the least important good in A_j should eliminate i's envy.

$$\text{EF} \;\Rightarrow\; \text{EFX} \;\Rightarrow\; \text{EF1}.$$

EF1 vs. EFX

EF1

$$\exists g \in A_j : \quad v_i(A_i) \geq v_i(A_j \setminus \{g\}).$$

$$A_j = \{g_1, g_2, g_3\}$$

g_1 g_2 g_3

remove a suitable good

EFX

$$\forall g \in A_j : \quad v_i(A_i) \geq v_i(A_j \setminus \{g\}).$$

$$A_j = \{g_1, g_2, g_3\}$$

g_1 g_2 g_3

any removed good works

Maximin Share Fairness

Definition (Maximin share)

For agent i, the maximin share (value) is

$$\mu_i^n(M) = \max_{B \in \mathcal{A}_n(M)} \min_{S \in B} v_i(S),$$

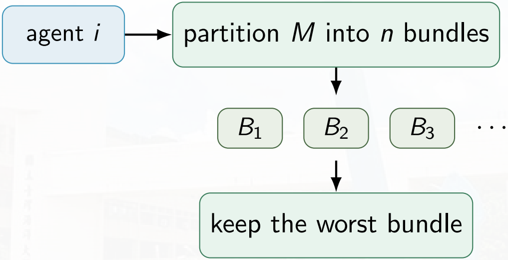
where $\mathcal{A}_n(M)$ is the set of all partitions of M into n bundles.

Definition (MMS allocation)

A is MMS if

$$v_i(A_i) \geq \mu_i^n(M) \qquad \forall i \in N.$$

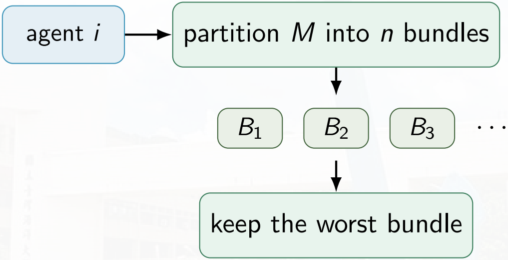
MMS: Cut-and-Choose Intuition



μ_i^n: best value agent i can guarantee
by partitioning into n bundles and receiving the worst one

- MMS is a relaxation of proportionality.
- Exact MMS allocations **need not exist** for more than two agents.

MMS: Cut-and-Choose Intuition



μ_i^n: best value agent i can guarantee
by partitioning into n bundles and receiving the worst one

- MMS is a relaxation of proportionality.
- Exact MMS allocations **need not exist** for more than two agents.
  - **Question:** How about the case of exactly two agents?

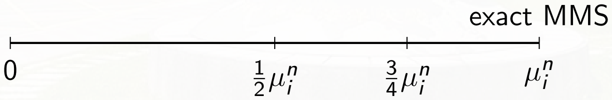
Approximate MMS

Definition (α-MMS)

For $\alpha \in (0,1]$, an allocation A is α-MMS if

$$v_i(A_i) \geq \alpha \cdot \mu_i^n(M) \qquad \forall i \in N.$$

- Because exact MMS may fail, approximation is central.
- Several algorithms guaranteeing constant-factor MMS approximations exist!



Relationships: A First Map

$$\begin{tikzcd}
\mathrm{EF} \arrow[r] \arrow[d] & \mathrm{EFX} \arrow[r] & \mathrm{EF1} \arrow[d, dashed] \\
\mathrm{PROP} \arrow[r] & \mathrm{MMS} \arrow[r, dashed] & \mathrm{PROP1}
\end{tikzcd}$$

Several implications require additivity or additional assumptions.

Existence and Computation Snapshot

Notion	Exists?	Poly-time?	Comment
EF	No	NP-hard to decide	too strong
PROP	No	NP-hard to decide	too strong
EF1	Yes	Yes	Round-Robin; ECE
EFX	Open	Open	special cases known
MMS	No	–	use approximation
α-MMS	Yes	Yes	best known $> 3/4$
PO alone	Yes	Yes	maximize welfare
EF1+PO	Yes	open	via MNW existence

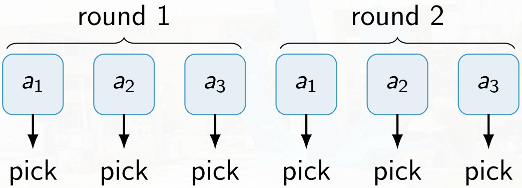
This table focuses on additive valuations over goods.

Outline

Round-Robin Algorithm

Algorithm idea

Fix an order of agents. Repeatedly let each agent pick her favorite remaining good.



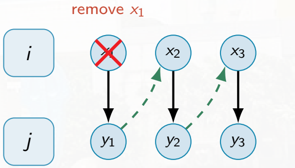
Theorem

Round-Robin returns an EF1 allocation for additive valuations.

Why Round-Robin Is EF1

Fix two agents i and j.

- If i picks before j in every round, then each pick of i is at least as good for i as the corresponding later pick of j.
- Therefore i does not envy j.
- If j envies i, remove i's first selected good; from then on, j picks before i in the shifted comparison.



Since i picks before j,
$v_i(x_t) \geq v_i(y_t)$ for each round t.

j's bundle is compared with $A_i \setminus \{x_1\}$

Example: EF1 Allocation

	g_1	g_2	g_3	g_4	g_5
a_1	15	3	2	2	6
a_2	7	5	5	5	7
a_3	20	3	3	3	3

$$A_1 = \{g_3, g_4\}, \qquad A_2 = \{g_2, g_5\}, \qquad A_3 = \{g_1\}.$$

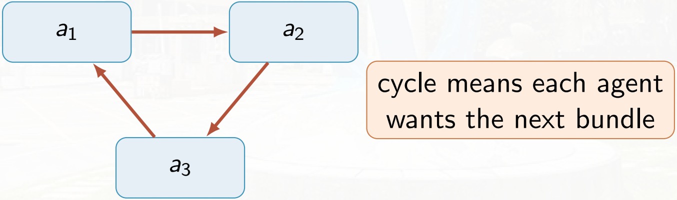
- a_2 and a_3 are not envious.
- a_1's envy toward a_2 is removed by deleting g_5 from A_2.
- a_1's envy toward a_3 is removed by deleting g_1 from A_3.

Envy Graph

Definition (Envy graph)

Given allocation A, the envy graph has one vertex per agent and an edge

$$i \to j \quad \textit{if } v_i(A_i) < v_i(A_j).$$

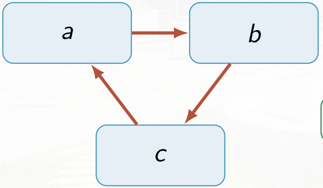


Envy-Cycle Elimination

Algorithm idea

Maintain an envy graph.

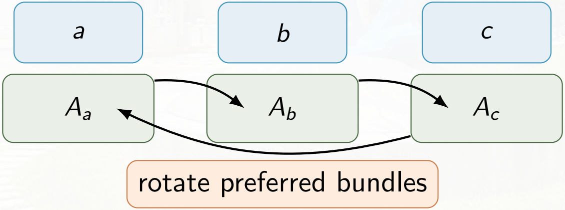
1. If there is an unenvied agent, give her an unallocated good.
2. If every agent is envied, find a directed envy cycle.
3. Rotate bundles along the cycle to eliminate it.



$$A_a \leftarrow A_b, \quad A_b \leftarrow A_c, \quad A_c \leftarrow A_a$$

Why Cycle Elimination Helps

- Along a directed cycle, each agent weakly prefers the next agent's bundle to her own.
- Rotating bundles along the cycle does not decrease the value of any cycle participant.
- At least one envy edge in the cycle disappears.
- Repeating this yields an envy graph with a source: an unenvied agent.



The EF1 Guarantee of Envy-Cycle Elimination

Theorem

Envy-Cycle Elimination outputs an EF1 allocation for monotone valuations.

Proof idea.

- A good is allocated only to an agent who is currently unenvied.
- Consider the last good g received by agent j.
- Just before g is added, no agent i envies j.
- Therefore, after the final allocation,

$$v_i(A_i) \geq v_i(A_j \setminus \{g\}),$$

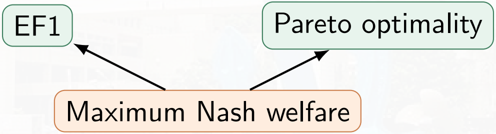
up to bundle rotations, which only improve the rotating agents' values.

$\square$

EF1 Is Stronger with Efficiency

Question

Can we obtain EF1 and Pareto optimality simultaneously?



Theorem (Caragiannis et al.)

Every maximum Nash welfare allocation is EF1 and PO.

Maximum Nash Welfare

Definition (Nash welfare)

For an allocation A, the Nash welfare is

$$\left(\prod_{i \in N} v_i(A_i)\right)^{1/n},$$

or equivalently, for exact maximization, the product $\prod_i v_i(A_i)$.

- MNW first maximizes the number of agents with positive value.
- Then it maximizes the product of positive values.
- It balances welfare and equality.

Why MNW Implies PO

Proof idea.

Suppose A is MNW but not PO. Then there is an allocation B such that

$$v_i(B_i) \geq v_i(A_i) \quad \forall i,$$

and strict inequality for at least one agent.

- Then the product of values under B is strictly larger.
- This contradicts maximality of Nash welfare.

$\square$

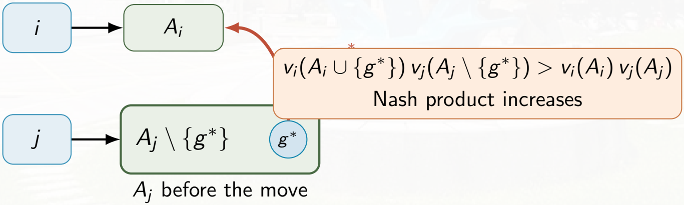
$$\boxed{A} \xrightarrow{\text{Pareto improves}} \boxed{B}$$

product increases

Why MNW Implies EF1 (rough idea)

Assume that A is an MNW allocation which is not EF1.

- Suppose i strongly envies j even after removing any good from A_j.
- Move a carefully chosen good $g^* \in \operatorname*{arg\,min}_{\substack{g \in A_j \\ v_i(g)>0}} \frac{v_j(g)}{v_i(g)}$ from j to i.
- The increase in i's value outweighs the decrease in j's value.
- Hence the Nash product increases, contradicting MNW.



Why MNW Implies EF1: Choosing the Good

- Suppose that an MNW allocation $A = (A_1, A_2, \ldots, A_n)$ is not EF1. Then there exist agents i and j such that

$$v_i(A_i) < v_i(A_j \setminus \{g\}) \quad \text{for every } g \in A_j. \tag{1}$$

- Choose

$$g^* \in \mathop{\arg\min}_{\substack{g \in A_j \\ v_i(g) > 0}} \frac{v_j(g)}{v_i(g)}.$$

- Thus, g^* has the smallest loss to j per unit of gain to i.
- Let $x = v_i(A_i)$, $z = v_i(A_j)$, $y = v_j(A_j)$, and $a = v_i(g^*)$, $b = v_j(g^*)$. By the choice of g^*,

$$\frac{v_j(g^*)}{v_i(g^*)} \leq \frac{v_j(A_j)}{v_i(A_j)} \qquad \text{(Please check!)}$$

Why MNW Implies EF1: Choosing the Good

- Suppose that an MNW allocation $A = (A_1, A_2, \dots, A_n)$ is not EF1. Then there exist agents i and j such that

$$v_i(A_i) < v_i(A_j \setminus \{g\}) \quad \text{for every } g \in A_j. \tag{1}$$

- Choose

$$g^* \in \arg\min_{\substack{g \in A_j \\ v_i(g) > 0}} \frac{v_j(g)}{v_i(g)}.$$

- Thus, g^* has the smallest loss to j per unit of gain to i.
- Let $x = v_i(A_i)$, $z = v_i(A_j)$, $y = v_j(A_j)$, and $a = v_i(g^*)$, $b = v_j(g^*)$. By the choice of g^*,

$$\frac{b}{a} = \frac{v_j(g^*)}{v_i(g^*)} \le \frac{v_j(A_j)}{v_i(A_j)} = \frac{y}{z}. \quad \text{(Please check!)}$$

Why MNW Implies EF1: The Nash Product Increases

- Recall: $x = v_i(A_i),\ z = v_i(A_j),\ y = v_j(A_j)$, and $a = v_i(g^*),\ \ b = v_j(g^*)$
- From the failure of EF1, $x < v_i(A_j \setminus \{g^*\}) = z - a$

Why MNW Implies EF1: The Nash Product Increases

- Recall: $x = v_i(A_i)$, $z = v_i(A_j)$, $y = v_j(A_j)$, and $a = v_i(g^*)$, $b = v_j(g^*)$
- From the failure of EF1, $x < v_i(A_j \setminus \{g^*\}) = z - a \Rightarrow x + a < z$.

Why MNW Implies EF1: The Nash Product Increases

- Recall: $x = v_i(A_i)$, $z = v_i(A_j)$, $y = v_j(A_j)$, and $a = v_i(g^*)$, $b = v_j(g^*)$
- From the failure of EF1, $x < v_i(A_j \setminus \{g^*\}) = z - a \Rightarrow x + a < z$.
- Then we have,

$$\frac{b}{y} \leq \frac{a}{z} < \frac{a}{x+a}. \quad (*)$$

Why MNW Implies EF1: The Nash Product Increases

- Recall: $x = v_i(A_i)$, $z = v_i(A_j)$, $y = v_j(A_j)$, and $a = v_i(g^*)$, $b = v_j(g^*)$
- From the failure of EF1, $x < v_i(A_j \setminus \{g^*\}) = z - a \;\Rightarrow\; x + a < z$.
- Then we have,

$$\frac{b}{y} \leq \frac{a}{z} < \frac{a}{x+a}. \quad (*)$$

- After transferring g^* from j to i, their contribution to the Nash product changes from xy to $(x+a)(y-b)$.

Why MNW Implies EF1: The Nash Product Increases

- Recall: $x = v_i(A_i)$, $z = v_i(A_j)$, $y = v_j(A_j)$, and $a = v_i(g^*)$, $b = v_j(g^*)$
- From the failure of EF1, $x < v_i(A_j \setminus \{g^*\}) = z - a \;\Rightarrow\; x + a < z$.
- Then we have,

$$\frac{b}{y} \le \frac{a}{z} < \frac{a}{x+a}. \quad (*)$$

- After transferring g^* from j to i, their contribution to the Nash product changes from xy to $(x+a)(y-b)$.
- Using $(*)$,

$$\frac{(x+a)(y-b)}{xy} = \left(1 + \frac{a}{x}\right)\left(1 - \frac{b}{y}\right) > \left(1 + \frac{a}{x}\right)\left(1 - \frac{a}{x+a}\right) = 1.$$

Therefore, $(x+a)(y-b) > xy$.

Why MNW Implies EF1: The Nash Product Increases

- Recall: $x = v_i(A_i)$, $z = v_i(A_j)$, $y = v_j(A_j)$, and $a = v_i(g^*)$, $b = v_j(g^*)$
- From the failure of EF1, $x < v_i(A_j \setminus \{g^*\}) = z - a \ \Rightarrow \ x + a < z$.
- Then we have,

$$\frac{b}{y} \leq \frac{a}{z} < \frac{a}{x+a}. \quad (*)$$

- After transferring g^* from j to i, their contribution to the Nash product changes from xy to $(x+a)(y-b)$.
- Using $(*)$,

$$\frac{(x+a)(y-b)}{xy} = \left(1 + \frac{a}{x}\right)\left(1 - \frac{b}{y}\right) > \left(1 + \frac{a}{x}\right)\left(1 - \frac{a}{x+a}\right) = 1.$$

  Therefore, $(x+a)(y-b) > xy$.
- All other agents' values remain unchanged, so the Nash welfare strictly increases. $(\Rightarrow\Leftarrow)$.

Generalization of the round-robin algorithm

A *picking sequence* specifies the **order** in which agents choose goods.

Example

For three agents, the sequence

$$1, 2, 3, 1, 2, 3, \ldots$$

is the usual round-robin sequence.

Sincere picking

When an agent is called, she chooses a remaining good that maximizes her own value.

That is, if agent i is called and the set of remaining goods is R, she chooses some

$$g \in \arg\max_{h \in R} v_i(h).$$

Definitions: Recursively Balanced Sequences

Recursively balanced picking sequence

A picking sequence is *recursively balanced* if it can be partitioned into rounds such that each complete round contains every agent **exactly once**. The order of agents may change from round to round.

Example

For three agents, the sequence

$$(1, 2, 3) \mid (3, 1, 2) \mid (2, 3, 1)$$

is recursively balanced.

Definitions: Recursively Balanced Sequences

Recursively balanced picking sequence

A picking sequence is *recursively balanced* if it can be partitioned into rounds such that each complete round contains every agent **exactly once**. The order of agents may change from round to round.

Example

For three agents, the sequence

$$(1, 2, 3) \mid (3, 1, 2) \mid (2, 3, 1)$$

is recursively balanced. While the sequence

$$(1, 1, 2) \mid (2, 3, 3)$$

is not recursively balanced.

Theorem

Theorem

Suppose agents have nonnegative additive valuations and choose goods sincerely according to a recursively balanced picking sequence.
Then the resulting allocation is EF1.

Proof idea

- Fix two agents i and j.
- We show that agent i does not envy agent j after removing j's first selected good.

Proof: Pairwise Comparison

- Fix two agents i and j. Let the goods selected by i be

$$x_1, x_2, \ldots, x_p,$$

and let the goods selected by j be

$$y_1, y_2, \ldots, y_q,$$

in chronological order.

Proof: Pairwise Comparison

- Fix two agents i and j. Let the goods selected by i be

$$x_1, x_2, \ldots, x_p,$$

  and let the goods selected by j be

$$y_1, y_2, \ldots, y_q,$$

  in chronological order.
- Because the sequence is recursively balanced, before j makes her r-th pick, agent i has already made at least $r-1$ picks.

Proof: Pairwise Comparison

- Fix two agents i and j. Let the goods selected by i be

$$x_1, x_2, \ldots, x_p,$$

  and let the goods selected by j be

$$y_1, y_2, \ldots, y_q,$$

  in chronological order.
- Because the sequence is recursively balanced, before j makes her r-th pick, agent i has already made at least $r-1$ picks.
- Hence, for every $r = 2, 3, \ldots, q$, the good y_r is still available when agent i chooses x_{r-1}.

Proof: Pairwise Comparison

- Fix two agents i and j. Let the goods selected by i be

$$x_1, x_2, \ldots, x_p,$$

  and let the goods selected by j be

$$y_1, y_2, \ldots, y_q,$$

  in chronological order.
- Because the sequence is recursively balanced, before j makes her r-th pick, agent i has already made at least $r - 1$ picks.
- Hence, for every $r = 2, 3, \ldots, q$, the good y_r is still available when agent i chooses x_{r-1}.
- By sincere picking,

$$v_i(x_{r-1}) \geq v_i(y_r).$$

Proof: Induction

- We prove by induction that for every $k = 1, 2, \ldots, q$,

$$P(k): \quad \sum_{r=1}^{k-1} v_i(x_r) \geq \sum_{r=2}^{k} v_i(y_r).$$

- **Base case.** For $k = 1$, both sums are empty, so $P(1)$ holds.
- **Inductive step.** Assume $P(k)$ holds:

$$\sum_{r=1}^{k-1} v_i(x_r) \geq \sum_{r=2}^{k} v_i(y_r).$$

Proof: Induction

- We prove by induction that for every $k = 1, 2, \ldots, q$,

$$P(k): \quad \sum_{r=1}^{k-1} v_i(x_r) \geq \sum_{r=2}^{k} v_i(y_r).$$

- **Base case.** For $k = 1$, both sums are empty, so $P(1)$ holds.
- **Inductive step.** Assume $P(k)$ holds:

$$\sum_{r=1}^{k-1} v_i(x_r) \geq \sum_{r=2}^{k} v_i(y_r).$$

- By sincere picking, $v_i(x_k) \geq v_i(y_{k+1})$.

Proof: Induction

- We prove by induction that for every $k = 1, 2, \ldots, q$,

$$P(k): \quad \sum_{r=1}^{k-1} v_i(x_r) \geq \sum_{r=2}^{k} v_i(y_r).$$

- **Base case.** For $k = 1$, both sums are empty, so $P(1)$ holds.
- **Inductive step.** Assume $P(k)$ holds:

$$\sum_{r=1}^{k-1} v_i(x_r) \geq \sum_{r=2}^{k} v_i(y_r).$$

- By sincere picking, $v_i(x_k) \geq v_i(y_{k+1})$. Adding these two inequalities gives

$$\sum_{r=1}^{k} v_i(x_r) \geq \sum_{r=2}^{k+1} v_i(y_r). \quad \text{Thus } P(k+1) \text{ holds.}$$

Proof: Concluding EF1

- By induction,

$$\sum_{r=1}^{q-1} v_i(x_r) \geq \sum_{r=2}^{q} v_i(y_r).$$

- Also, recursively balancedness implies $p \geq q - 1$.
- Therefore, using nonnegativity and additivity,

$$v_i(A_i) = \sum_{r=1}^{p} v_i(x_r) \geq \sum_{r=1}^{q-1} v_i(x_r) \geq \sum_{r=2}^{q} v_i(y_r) = v_i(A_j \setminus \{y_1\}).$$

- Hence agent i does not envy agent j after removing y_1. Since i and j were arbitrary, the allocation is EF1.

Outline

Why EFX Is Difficult

- EF1 asks for one suitable good to remove.
- EFX asks that every good in the envied bundle works.
- The EFX existence problem for unrestricted additive valuations remains open for $n \geq 4$ agents.

Open Problem

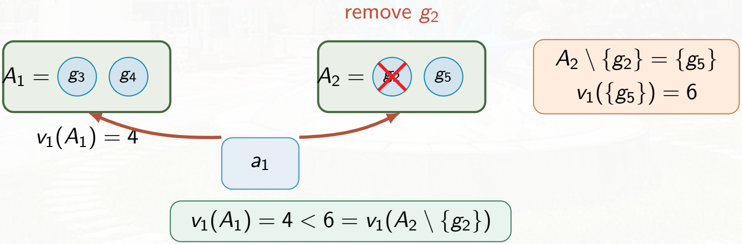
Do EFX allocations always exist for $n \geq 4$ agents with unrestricted additive valuations?

Example: EF1 but Not EFX

Using the earlier instance:

$$A_1 = \{g_3, g_4\}, \qquad A_2 = \{g_2, g_5\}, \qquad A_3 = \{g_1\}.$$

- This allocation is EF1.
- But it is not EFX: for a_1, removing g_2 from A_2 leaves $\{g_5\}$.
- $v_1(A_1) = 4 < 6 = v_1(\{g_5\})$.

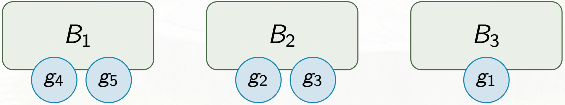


Example: EFX Allocation

For the same instance, consider

$$B_1 = \{g_4, g_5\}, \qquad B_2 = \{g_2, g_3\}, \qquad B_3 = \{g_1\}.$$

- a_1's possible envy toward a_3 is removed by deleting g_1.
- a_2's possible envy toward a_1 is removed even after deleting any good in B_1.
- The allocation is EFX.



Known EFX Existence Results

Setting	Result
Identical valuations	EFX exists via leximin++
Ordered instances	Envy-Cycle Elimination is EFX
Two agents	EFX exists via divide-and-choose
Three agents	EFX exists; pseudo-polynomial algorithm
Bi-valued valuations	EFX exists and is computable
General additive, $n \geq 4$	open

These are positive steps, but not a full answer for additive valuations.

Ordered Instances

Definition (Ordered instance)

An instance is ordered if the goods can be ordered as

$$g_1, g_2, \ldots, g_m$$

such that for every agent i,

$$v_i(g_1) \geq v_i(g_2) \geq \cdots \geq v_i(g_m).$$

- Agents may have different values.
- But they agree on the ranking of goods.

$$g_1 \quad g_2 \quad g_3 \quad g_4 \quad g_5$$

$$\longrightarrow$$

decreasing value for every agent

Why Envy-Cycle Elimination Gives EFX for Ordered Instances

- Goods are allocated from high value to low value for every agent.
- Let g be the last good allocated to agent j.
- Since j was unenvied just before receiving g,

$$v_i(A_i) \geq v_i(A_j \setminus \{g\}).$$

- Every earlier good $g' \in A_j$ is at least as valuable as g, so removing g' leaves an even smaller bundle:

$$v_i(A_j \setminus \{g'\}) \leq v_i(A_j \setminus \{g\}).$$

Therefore the EF1 argument upgrades to EFX.

α-EFX

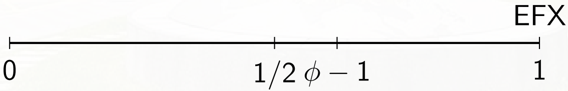
Definition (α-EFX)

For $\alpha \in (0, 1]$, allocation A is α-EFX if

$$v_i(A_i) \geq \alpha \cdot v_i(A_j \setminus \{g\})$$

for every $i, j \in N$ and every $g \in A_j$.

- $\alpha = 1$ recovers EFX.
- Approximate EFX asks for a multiplicative relaxation.



Approximate EFX Results

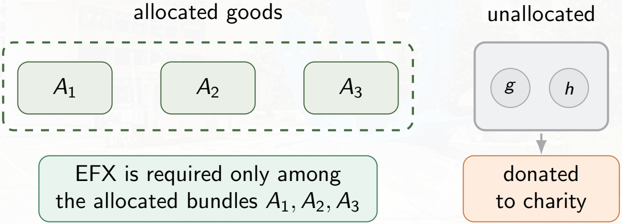
- 1/2-EFX allocations always exist, even for subadditive valuations.
- For additive valuations, Envy-Cycle Elimination computes 1/2-EFX with a suitable tie-breaking rule.
- The additive case has been improved to $\phi - 1 \approx 0.618$ by combining Round-Robin, envy-cycle ideas, and preprocessing.

Open Problem

What is the best α such that α-EFX allocations always exist?

Partial EFX: Donating Goods

- Another relaxation: allocate only a subset of goods.
- Leaving all goods unallocated is trivial, so we require few unallocated goods or high welfare.
- Known results allow EFX with at most $n-1$ or $n-2$ unallocated goods in different settings.



Open Problem

Can exact EFX be achieved by donating a sublinear number of goods?

Outline

MMS Values Are Monotone

Lemma (MMS monotonicity)

For any agent i and good g,

$$\mu_i^{n-1}(M \setminus \{g\}) \geq \mu_i^n(M).$$

Proof idea.

Take an n-partition that gives every part value at least $\mu_i^n(M)$ to agent i. Remove the bundle containing g, and distribute its remaining goods among the other $n-1$ bundles. Each remaining bundle still has value at least $\mu_i^n(M)$. □

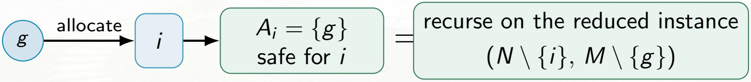
Using Monotonicity Algorithmically

Reduction step

If some good g already gives agent i enough value, allocate g to i and recurse.

$$v_i(g) \geq \alpha\mu_i^n(M) \quad \Longrightarrow \quad A_i = \{g\} \text{ is safe for } i.$$

- By monotonicity, the remaining agents' MMS benchmarks do not decrease.
- This is a common preprocessing step in MMS algorithms.



Round-Robin Gives 1/2-MMS

Assume after preprocessing,

$$v_i(g) \le \frac{1}{2}\mu_i^n(M) \quad \forall i, g.$$

Theorem

Round-Robin returns a 1/2-MMS allocation.

Proof sketch.

For agent i, compare A_i to every other bundle after deleting that bundle's first-round good. Summing the comparisons yields

$$n \cdot v_i(A_i) \ge v_i(M) - \sum_{j \in N} v_i(g_j) \ge n\mu_i^n - \frac{n}{2}\mu_i^n.$$

Hence $v_i(A_i) \ge \frac{1}{2}\mu_i^n$. $\square$

Envy-Cycle Elimination Gives 1/2-MMS

Theorem

Envy-Cycle Elimination computes a 1/2-MMS allocation.

- The same algorithm that gives EF1 also gives a constant MMS approximation.
- Proof idea: for any fixed agent i, most other bundles have value at most $2v_i(A_i)$ to agent i.
- Use the MMS benchmark on the remaining goods and agents.

EF1 $\longrightarrow$ 1/2-EFX $\longrightarrow$ 1/2-MMS

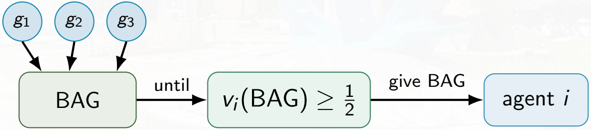
one simple algorithm has multiple guarantees

Bag-Filling Algorithm

Normalize $\mu_i^n(M) = 1$ for all agents.

Algorithm idea

1. Start with an empty bag.
2. Add goods until some remaining agent values the bag at least 1/2.
3. Give the bag to such an agent and remove her.
4. Repeat.

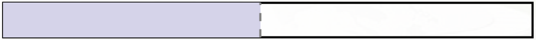


Once some remaining agent values the bag at least 1/2, allocate the bag to that agent and repeat.

Why Bag-Filling Gives 1/2-MMS

- Every allocated bag is worth at least $1/2$ to the agent receiving it.
- Before the last good enters the bag, every remaining agent values the bag below $1/2$.
- Each single good is worth at most $1/2$ after preprocessing.
- Hence every remaining agent loses less than her full MMS benchmark when a bag is removed.

bag before last good $< \frac{1}{2}$ last good $\leq \frac{1}{2}$



lost value < 1 for remaining agents

Beyond 1/2-MMS

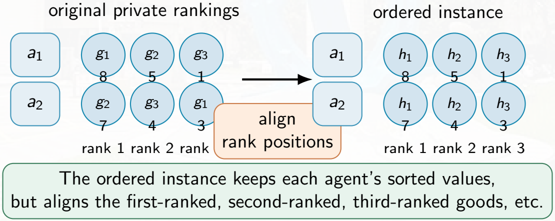
Guarantee	Main idea / status
1/2-MMS	simple algorithms: Round-Robin, ECE, bag-filling
2/3-MMS	polynomial-time algorithms
$3/4-\varepsilon$-MMS	elaborate approximation algorithms
$3/4+1/(12n)$-MMS	improved current-style guarantee
$>39/40$	impossible in general additive case

Open Problem

Can we improve the best known additive MMS approximation? Can the inapproximability gap be tightened?

Ordered Instances as Worst Cases

- For MMS approximation, one can often reduce to ordered instances.
- Sort the goods by each agent's private ranking, then align the ranks.
- If the ordered instance has a good allocation, one can map it back to the original instance without decreasing each agent's value.



Special Cases for Exact MMS

- Two agents: cut-and-choose style protocol gives MMS.
- Identical valuations: MMS exists by definition.
- Binary, ternary, and bi-valued valuations: exact MMS existence/computation results are known.
- Matroid-rank valuations: exact MMS allocations exist.

Open Problem

Identify broader structured valuation classes for which exact MMS allocations always exist.

MMS vs. EFX

- MMS relaxes proportionality.
- EFX relaxes envy-freeness.
- They are conceptually different: one is based on a personal partition guarantee; the other is based on **pairwise** envy comparisons.

MMS EFX

no simple dominance relation in general

Outline

Pareto Optimality

Definition (Pareto optimality)

An allocation A is **Pareto optimal** *(PO) if there is no allocation B such that*

$$v_i(B_i) \geq v_i(A_i) \quad \forall i \in N,$$

and

$$v_j(B_j) > v_j(A_j) \quad \textit{for some } j \in N.$$

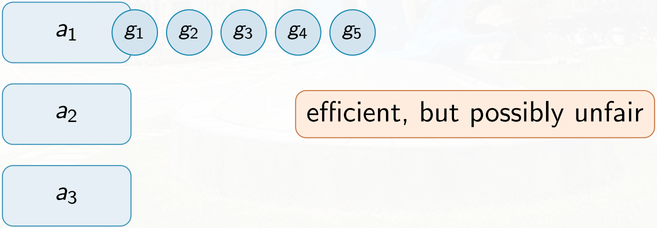
- PO rules out wasteful allocations.
- It does not, by itself, imply fairness.

PO Alone Can Be Unfair

Example 6.1

Allocate each good to the agent who values it most.

- This maximizes utilitarian social welfare.
- Therefore it is Pareto optimal.
- But one agent may receive almost everything.



Fairness Plus PO

Fairness target	Compatibility with PO
EF	may not exist
PROP	may not exist
EF1	exists via MNW
PROP1	exists and polynomial-time computable
EFX	compatible in some restricted settings
MMS	exact MMS may not exist

Note

Existence and efficient computation are different questions.

Fisher Market Proxy

- A Fisher market has agents, goods, budgets, and prices.
- With divisible goods, a competitive equilibrium from equal incomes is envy-free and fractionally Pareto optimal.
- Algorithms for indivisible goods often perturb budgets or valuations to obtain integral equilibria.

budgets → prices → market equilibrium → allocation

Market Equilibrium Conditions

A market outcome (X, p) consists of a fractional allocation X and prices p.

1. **Market clears:** every positive-price item is fully allocated.
2. **Budgets are exhausted:** each agent spends her budget.
3. **Bang-per-buck optimality:** agents spend only on goods maximizing

$$\frac{v_i(g)}{p_g}.$$

This connects discrete fair division to convex optimization and market algorithms.

Nash Welfare and Computation

- MNW allocations imply EF1 and PO.
- But computing or even approximating MNW is hard in general.
- Pseudo-polynomial algorithms exist for EF1+PO via market-based methods.
- A strongly polynomial-time algorithm is known for PROP1+PO.

Open Problem

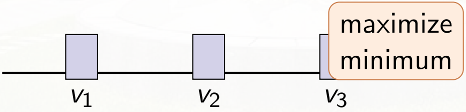
Can an EF1 and PO allocation be computed in polynomial time for additive valuations?

Egalitarian Welfare

Definition (Egalitarian welfare)

$$\min_{i \in N} v_i(A_i).$$

- Maximizing egalitarian welfare is itself a fairness objective.
- It is related to the Santa Claus problem.
- However, an egalitarian-optimal allocation may fail to approximate EF1 or MMS.



Outline

PROP1

Definition (PROP1)

An allocation A is **proportional up to one good** *if, for every agent i, there exists a good $g \in M \setminus A_i$ such that*

$$v_i(A_i \cup \{g\}) \geq \frac{v_i(M)}{n}.$$

- Agent i may be short of proportionality.
- But adding one suitable outside good would meet the proportional benchmark.

PropX

Definition (PropX)

An allocation A is **proportional up to any good** *if, for every agent i and every good $g \in M \setminus A_i$,*

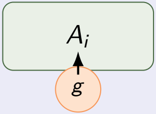
$$v_i(A_i \cup \{g\}) \geq \frac{v_i(M)}{n}.$$

- PropX is much stronger than PROP1.
- It requires even the least useful outside good to complete the proportional guarantee.
- PropX allocations need not exist for goods.

PROP1 vs. PropX

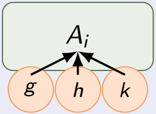
PROP1

$$\exists g \notin A_i : \; v_i(A_i \cup \{g\}) \geq v_i(M)/n.$$



PropX

$$\forall g \notin A_i : \; v_i(A_i \cup \{g\}) \geq v_i(M)/n.$$



PropM: A Middle Ground

- PropX may be too demanding.
- PropM uses the best among the least valuable goods in other agents' bundles.

$$d_i(A) = \max_{j \neq i} \min_{g \in A_j} v_i(g).$$

Definition (PropM)

An allocation A is PropM if

$$v_i(A_i) + d_i(A) \geq \frac{v_i(M)}{n} \qquad \forall i \in N.$$

PropM allocations always exist and can be computed in polynomial time.

Proportionality Relaxations: Summary

Notion	Guarantee style	Existence for goods
PROP	actual $1/n$ share	no
PROP1	add one suitable good	yes
PropM	add a maximin outside good	yes
PropX	add any outside good	no

PROP → PropX → PropM → PROP1

Outline

Social Welfare

Definition (Utilitarian social welfare)

$$SW(A) = \sum_{i \in N} v_i(A_i).$$

- The unconstrained welfare optimum assigns every good to an agent who values it most.
- Fairness constraints may force welfare loss.

maximize SW → impose fairness → welfare loss

Price of Fairness

Let F be a fairness criterion and $F(I)$ be the set of allocations satisfying F.

Definition (Price of fairness)

$$PoF(F) = \sup_{I} \min_{A \in F(I)} \frac{OPT(I)}{SW(A)},$$

where $OPT(I)$ is the maximum possible social welfare in instance I.

- Ratio 1 means no efficiency loss.
- Larger ratios mean fairness can be costly.

A Lower Bound Example for EF1

For two agents and three goods:

	g_1	g_2	g_3
a_1	$1/3 - 2\varepsilon$	$1/3 + \varepsilon$	$1/3 + \varepsilon$
a_2	0	$1/2$	$1/2$

- Welfare optimum: give g_1 to a_1 and g_2, g_3 to a_2.

$$\mathrm{OPT} = \frac{4}{3} - 2\varepsilon.$$

- But any EF1 allocation cannot give both g_2 and g_3 to a_2.

Computing the Ratio

In an EF1 allocation for the lower-bound instance, the welfare is at most

$$\left(\frac{1}{3} - 2\varepsilon\right) + \left(\frac{1}{3} + \varepsilon\right) + \frac{1}{2} = \frac{7}{6} - \varepsilon.$$

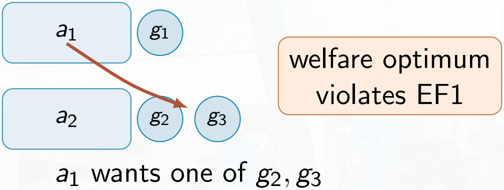
Thus

$$\frac{\mathrm{OPT}}{\mathrm{SW}(A)} \geq \frac{\frac{4}{3} - 2\varepsilon}{\frac{7}{6} - \varepsilon} \longrightarrow \frac{8}{7} \quad \text{as } \varepsilon \to 0.$$

Conclusion

The price of EF1 is at least 8/7 even for two agents.

Visualizing the EF1 Cost



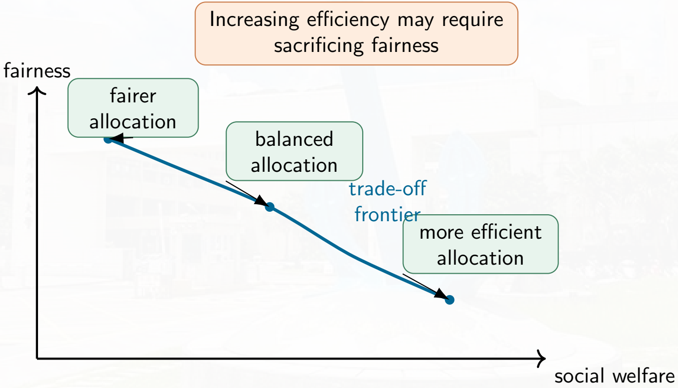
- Fairness may force reallocating a good away from the agent who values it most.

Price of Fairness: Known Bounds

Criterion	Price of fairness behavior
EF1	$\Theta(\sqrt{n})$
1/2-MMS	tight bounds known
PROP1	tight bounds known
Two-agent EF1	lower bound 8/7; matching-style upper analyses

- Price of fairness is a worst-case measure.
- It complements existence and algorithmic guarantees.

Efficiency-Fairness Trade-off



Outline

Main Algorithmic Techniques

sequential picking → envy graph → bag filling

Nash welfare → Fisher markets → ordered reductions

- Many results combine simple combinatorial algorithms with welfare or market structure.

Open Problems to Remember

1. Does EFX always exist for additive goods and $n \geq 4$ agents?
2. Can EF1+PO be computed in polynomial time?
3. What is the optimal approximation ratio for MMS?
4. What is the optimal approximation ratio for EFX?
5. Which structured valuation classes guarantee exact MMS or EFX?

Conceptual Summary

Notion	Origin idea	Main challenge
EF	no envy	often impossible
PROP	$1/n$ share	often impossible
EF1	remove one good	combine with PO efficiently
EFX	remove any good	existence open
MMS	partition guarantee	exact existence fails
PO	no Pareto improvement	may conflict with fairness
PoF	welfare loss of fairness	tight worst-case bounds

Discussions