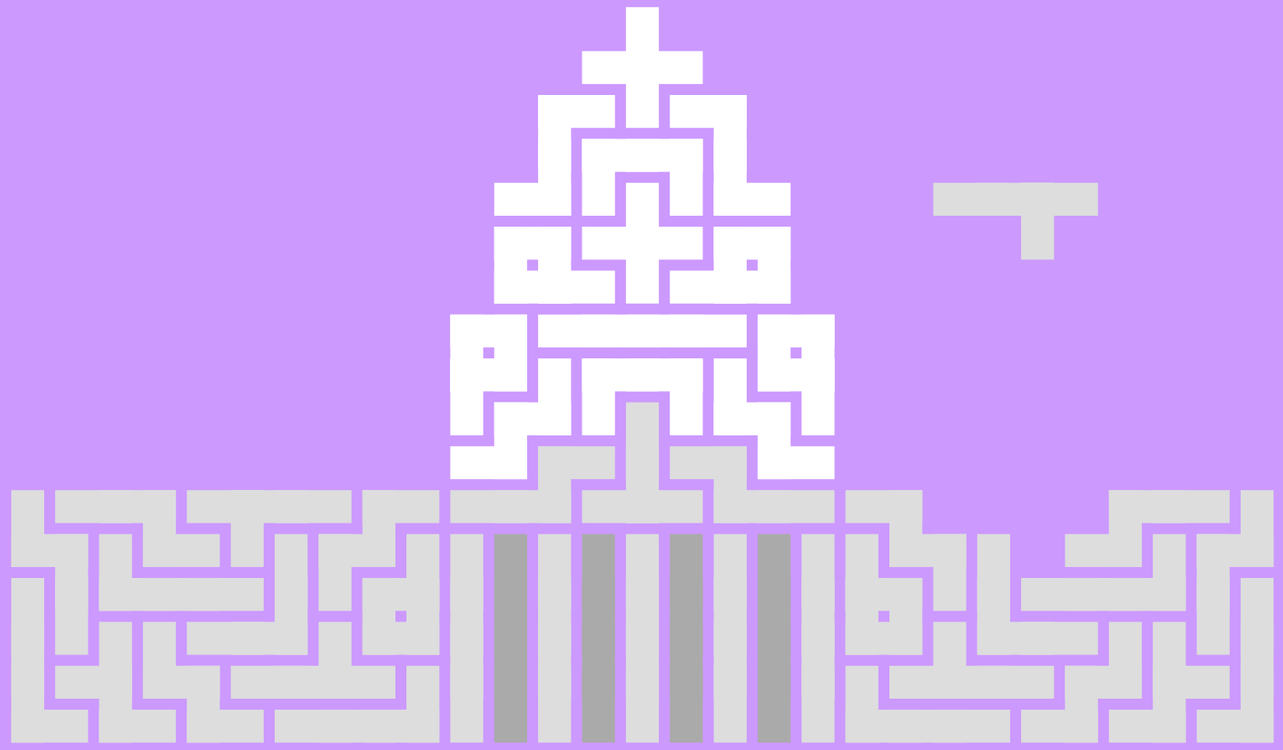


Algorithms For Democratic Decision-Making

Jamie Tucker-Foltz • Yale University • Spring 2026

Lecture 13: **Fair Division 1: Indivisible Goods**

Introduction

Fair Division: How do you fairly allocate *items* among *agents*?

Introduction

Fair Division: How do you fairly allocate *items* among *agents*?

http://spliddit.org/

Most widely-studied setting: indivisible goods, additive valuations

There are n agents (numbered 1, 2, ..., n) and m indivisible goods g_1, g_2, ..., g_m,.

Most widely-studied setting: indivisible goods, additive valuations

There are n agents (numbered 1, 2, ..., n) and m indivisible goods g_1, g_2, ..., g_m,.

Agent i has value $v_i(g_k)$ for good g_k. Define $v_i(S)$ to be the sum of $v_i(g)$ over $g \in S$.

Most widely-studied setting: indivisible goods, additive valuations

There are n agents (numbered 1, 2, ..., n) and m indivisible goods g_1, g_2, ..., g_m,.

Agent i has value $v_i(g_k)$ for good g_k. Define $v_i(S)$ to be the sum of $v_i(g)$ over $g \in S$.

Objective:
Find a fair allocation $A = (A_1, A_2, ..., A_n)$, where each A_i is a disjoint subset of goods.

Most widely-studied setting: indivisible goods, additive valuations

There are n agents (numbered 1, 2, ..., n) and m indivisible goods g_1, g_2, ..., g_m,.

Agent i has value $v_i(g_k)$ for good g_k. Define $v_i(S)$ to be the sum of $v_i(g)$ over $g \in S$.

Objective:
Find a fair allocation $A = (A_1, A_2, ..., A_n)$, where each A_i is a disjoint subset of goods.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

Most widely-studied setting: indivisible goods, additive valuations

There are n agents (numbered 1, 2, ..., n) and m indivisible goods $g_1, g_2, \ldots, g_m$,.

Agent i has value $v_i(g_k)$ for good g_k. Define $v_i(S)$ to be the sum of $v_i(g)$ over $g \in S$.

Objective:
Find a fair allocation $A = (A_1, A_2, \ldots, A_n)$, where each A_i is a disjoint subset of goods.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

An allocation A is *envy-free* *(EF)* if $\forall\, i, j \in [n]$,
$v_i(A_i) \geq v_i(A_j$)

Most widely-studied setting: indivisible goods, additive valuations

There are n agents (numbered 1, 2, ..., n) and m indivisible goods g_1, g_2, ..., g_m,.

Agent i has value $v_i(g_k)$ for good g_k. Define $v_i(S)$ to be the sum of $v_i(g)$ over $g \in S$.

Objective:
Find a fair allocation $A = (A_1, A_2, ..., A_n)$, where each A_i is a disjoint subset of goods.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

An allocation A is *envy-free up to one good (EF1)* if $\forall\ i, j \in [n]$, $\exists\ g \in A_j$,
$v_i(A_i) \geq v_i(A_j \setminus \{g\})$

Most widely-studied setting: indivisible goods, additive valuations

There are n agents (numbered 1, 2, ..., n) and m indivisible goods g_1, g_2, ..., g_m,.

Agent i has value $v_i(g_k)$ for good g_k. Define $v_i(S)$ to be the sum of $v_i(g)$ over $g \in S$.

Objective:
Find a fair allocation $A = (A_1, A_2, ..., A_n)$, where each A_i is a disjoint subset of goods.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

An allocation A is *envy-free up to one good (EF1)* if $\forall\ i, j \in [n]$, $\exists\ g \in A_j$,
$v_i(A_i) \geq v_i(A_j \setminus \{g\})$

Most widely-studied setting: indivisible goods, additive valuations

There are n agents (numbered 1, 2, ..., n) and m indivisible goods g_1, g_2, ..., g_m,.

Agent i has value $v_i(g_k)$ for good g_k. Define $v_i(S)$ to be the sum of $v_i(g)$ over $g \in S$.

Objective:
Find a fair allocation $A = (A_1, A_2, ..., A_n)$, where each A_i is a disjoint subset of goods.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

✘ Not EF1

An allocation A is *envy-free up to one good (EF1)* if $\forall\, i, j \in [n]$, $\exists\, g \in A_j$,
$v_i(A_i) \geq v_i(A_j \setminus \{g\})$

Most widely-studied setting: indivisible goods, additive valuations

There are n agents (numbered 1, 2, ..., n) and m indivisible goods $g_1, g_2, \ldots, g_m$,.

Agent i has value $v_i(g_k)$ for good g_k. Define $v_i(S)$ to be the sum of $v_i(g)$ over $g \in S$.

Objective:
Find a fair allocation $A = (A_1, A_2, \ldots, A_n)$, where each A_i is a disjoint subset of goods.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

✔ EF1

An allocation A is *envy-free up to one good (EF1)* if $\forall\, i, j \in [n]$, $\exists\, g \in A_j$,
$v_i(A_i) \geq v_i(A_j \setminus \{g\})$

Algorithm 1: Round Robin

Agents take turns picking their favorite item remaining.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

Algorithm 1: Round Robin

Agents take turns picking their favorite item remaining.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

Algorithm 1: Round Robin

Agents take turns picking their favorite item remaining.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

Algorithm 1: Round Robin

Agents take turns picking their favorite item remaining.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

Algorithm 1: Round Robin

Agents take turns picking their favorite item remaining.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

Algorithm 1: Round Robin

Agents take turns picking their favorite item remaining.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

Algorithm 1: Round Robin

Agents take turns picking their favorite item remaining.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

Algorithm 1: Round Robin

Agents take turns picking their favorite item remaining.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

Proposition

Round Robin computes an EF1 allocation.

Algorithm 1: Round Robin

Agents take turns picking their favorite item remaining.

Example

		g_1	g_2	g_3	g_4	g_5	g_6
$i \rightarrow$	Agent 1	3	3	3	10	2	1
$j \rightarrow$	Agent 2	1	5	6	9	2	1
	Agent 3	1	7	6	9	2	1

Proposition

Round Robin computes an EF1 allocation.

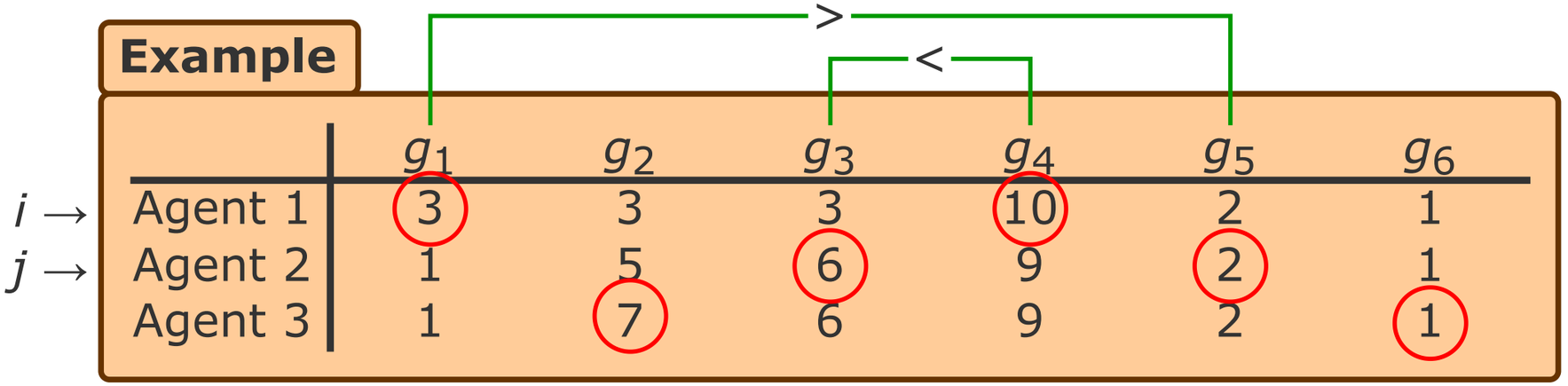
Proof. Consider an arbitrary pair of agents, $i < j$.

Observe that i does not envy j because

Algorithm 1: Round Robin

Agents take turns picking their favorite item remaining.

Example



Proposition

Round Robin computes an EF1 allocation.

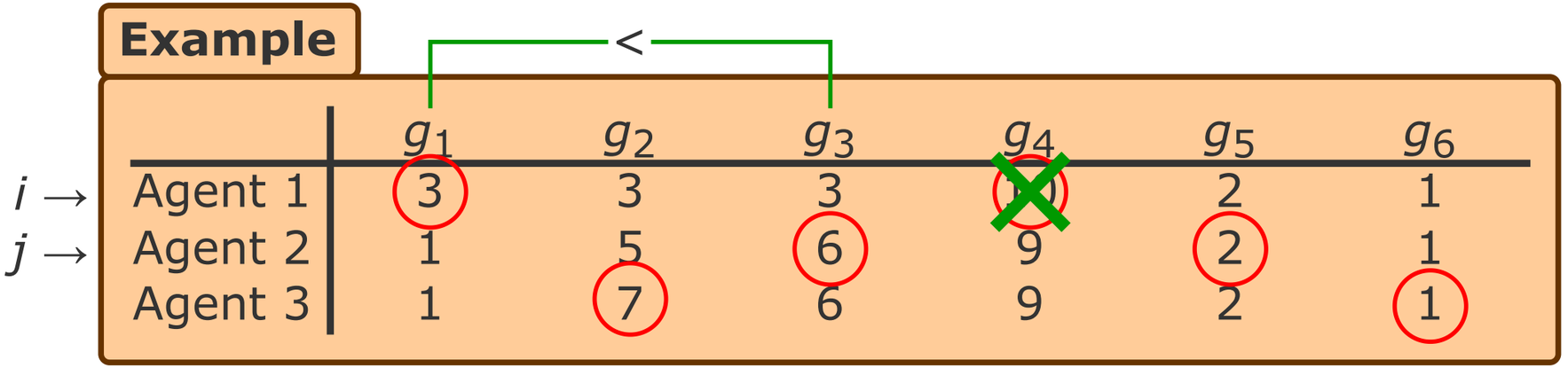
Proof. Consider an arbitrary pair of agents, $i < j$.

Observe that i does not envy j because i's k^{th} good is worth more to i than j's k^{th} good.

Algorithm 1: Round Robin

Agents take turns picking their favorite item remaining.

Example



Proposition

Round Robin computes an EF1 allocation.

Proof. Consider an arbitrary pair of agents, $i < j$.

Observe that i does not envy j because i's k^{th} good is worth more to i than j's k^{th} good.

And j does not envy i after removing the first good i picked, because j's k^{th} good is worth more to j than i's $(k + 1)^{\text{th}}$ good. ■

Algorithm 2: Envy Cycle Elimination

While there are some unallocated items:

Let G be the envy graph on the set of agents, where $i \to j$ whenever i envies j;

Algorithm 2: Envy Cycle Elimination

While there are some unallocated items:
    Let G be the envy graph on the set of agents, where $i \to j$ whenever i envies j;
    If G has a source vertex i:
        Give i the next (arbitrarily selected) item;
    Else:

Algorithm 2: Envy Cycle Elimination

While there are some unallocated items:
    Let G be the envy graph on the set of agents, where $i \to j$ whenever i envies j;
    If G has a source vertex i:
        Give i the next (arbitrarily selected) item;
    Else:
        Then G must have a cycle, so shift bundles around some cycle;

Algorithm 2: Envy Cycle Elimination

While there are some unallocated items:
    Let G be the envy graph on the set of agents, where $i \to j$ whenever i envies j;
    If G has a source vertex i:
        Give i the next (arbitrarily selected) item;
    Else:
        Then G must have a cycle, so shift bundles around some cycle;

This terminates after a polynomial number of iterations since each time we either:
(A) Allocate an item, or
(B) Eliminate eges from G.

Algorithm 2: Envy Cycle Elimination

While there are some unallocated items:
    Let G be the envy graph on the set of agents, where $i \to j$ whenever i envies j;
    If G has a source vertex i:
        Give i the next (arbitrarily selected) item;
    Else:
        Then G must have a cycle, so shift bundles around some cycle;

This terminates after a polynomial number of iterations since each time we either:
(A) Allocate an item, or
(B) Eliminate eges from G.

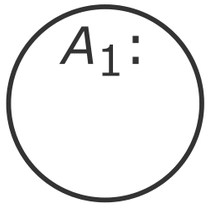
Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

Algorithm 2: Envy Cycle Elimination

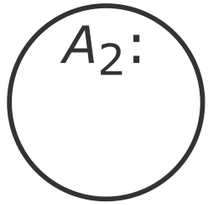
While there are some unallocated items:
    Let G be the envy graph on the set of agents, where $i \to j$ whenever i envies j;
    If G has a source vertex i:
        Give i the next (arbitrarily selected) item;
    Else:
        Then G must have a cycle, so shift bundles around some cycle;

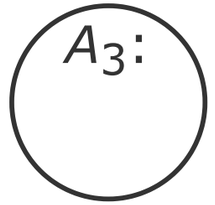
This terminates after a polynomial number of iterations since each time we either:
(A) Allocate an item, or
(B) Eliminate eges from G.



Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1





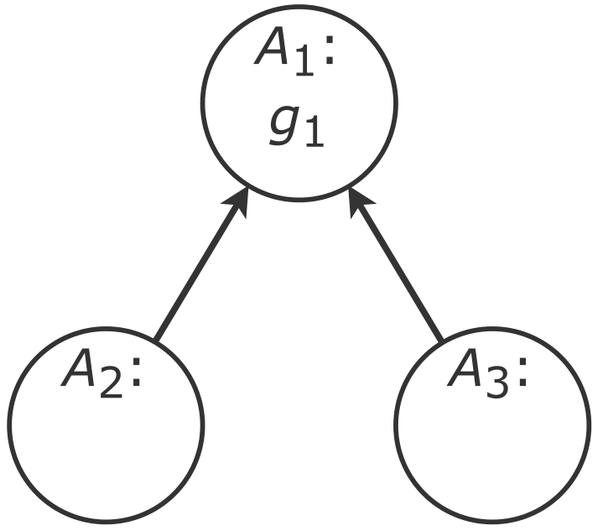
Algorithm 2: Envy Cycle Elimination

While there are some unallocated items:
    Let G be the envy graph on the set of agents, where $i \to j$ whenever i envies j;
    If G has a source vertex i:
        Give i the next (arbitrarily selected) item;
    Else:
        Then G must have a cycle, so shift bundles around some cycle;

This terminates after a polynomial number of iterations since each time we either:
(A) Allocate an item, or
(B) Eliminate eges from G.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1



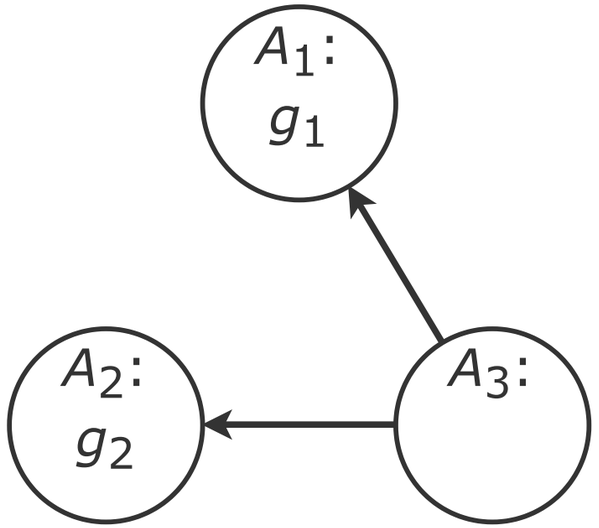
Algorithm 2: Envy Cycle Elimination

While there are some unallocated items:
    Let G be the envy graph on the set of agents, where $i \to j$ whenever i envies j;
    If G has a source vertex i:
        Give i the next (arbitrarily selected) item;
    Else:
        Then G must have a cycle, so shift bundles around some cycle;

This terminates after a polynomial number of iterations since each time we either:
(A) Allocate an item, or
(B) Eliminate eges from G.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1



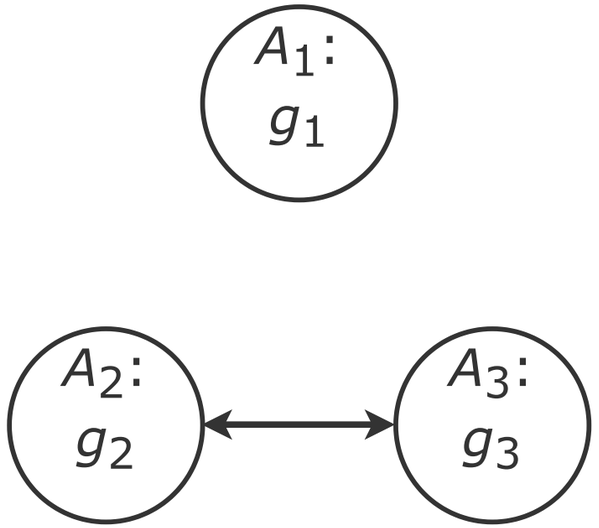
Algorithm 2: Envy Cycle Elimination

While there are some unallocated items:
    Let G be the envy graph on the set of agents, where $i \to j$ whenever i envies j;
    If G has a source vertex i:
        Give i the next (arbitrarily selected) item;
    Else:
        Then G must have a cycle, so shift bundles around some cycle;

This terminates after a polynomial number of iterations since each time we either:
(A) Allocate an item, or
(B) Eliminate eges from G.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1



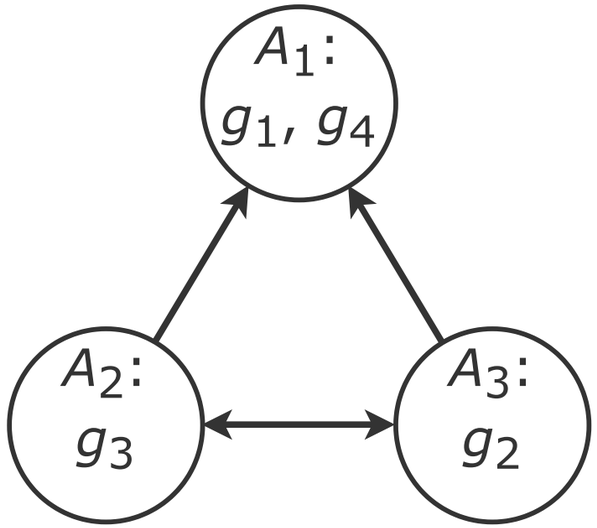
Algorithm 2: Envy Cycle Elimination

While there are some unallocated items:
    Let G be the envy graph on the set of agents, where $i \to j$ whenever i envies j;
    If G has a source vertex i:
        Give i the next (arbitrarily selected) item;
    Else:
        Then G must have a cycle, so shift bundles around some cycle;

This terminates after a polynomial number of iterations since each time we either:
(A) Allocate an item, or
(B) Eliminate eges from G.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1



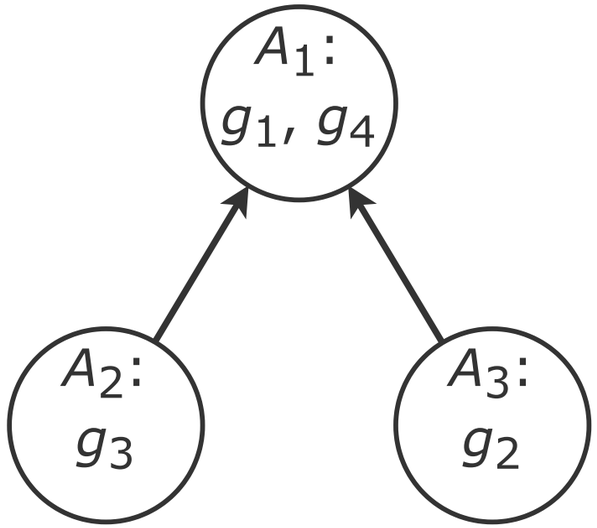
Algorithm 2: Envy Cycle Elimination

While there are some unallocated items:
    Let G be the envy graph on the set of agents, where $i \to j$ whenever i envies j;
    If G has a source vertex i:
        Give i the next (arbitrarily selected) item;
    Else:
        Then G must have a cycle, so shift bundles around some cycle;

This terminates after a polynomial number of iterations since each time we either:
(A) Allocate an item, or
(B) Eliminate eges from G.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1



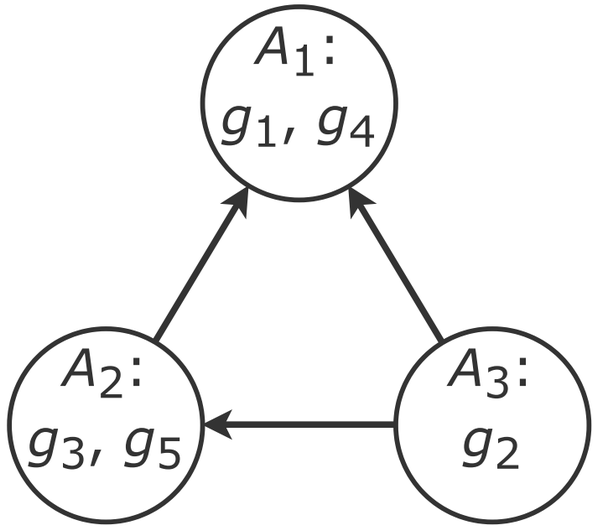
Algorithm 2: Envy Cycle Elimination

While there are some unallocated items:
    Let G be the envy graph on the set of agents, where $i \to j$ whenever i envies j;
    If G has a source vertex i:
        Give i the next (arbitrarily selected) item;
    Else:
        Then G must have a cycle, so shift bundles around some cycle;

This terminates after a polynomial number of iterations since each time we either:
(A) Allocate an item, or
(B) Eliminate eges from G.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1



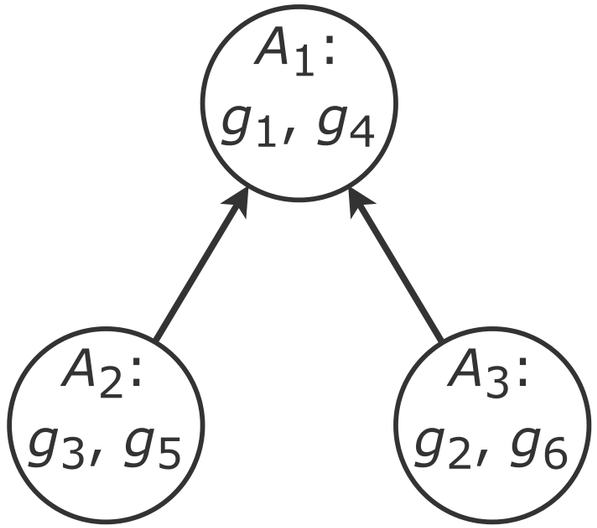
Algorithm 2: Envy Cycle Elimination

While there are some unallocated items:
    Let G be the envy graph on the set of agents, where $i \to j$ whenever i envies j;
    If G has a source vertex i:
        Give i the next (arbitrarily selected) item;
    Else:
        Then G must have a cycle, so shift bundles around some cycle;

This terminates after a polynomial number of iterations since each time we either:
(A) Allocate an item, or
(B) Eliminate eges from G.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1



Algorithm 2: Envy Cycle Elimination

While there are some unallocated items:
    Let G be the envy graph on the set of agents, where $i \to j$ whenever i envies j;
    If G has a source vertex i:
        Give i the next (arbitrarily selected) item;
    Else:
        Then G must have a cycle, so shift bundles around some cycle;

Theorem (Lipton, Markakis, Mossel, Saberi, 2004)

The Envy Cycle Elimination algorithm computes an EF1 allocation

Algorithm 2: Envy Cycle Elimination

While there are some unallocated items:
    Let G be the envy graph on the set of agents, where $i \to j$ whenever i envies j;
    If G has a source vertex i:
        Give i the next (arbitrarily selected) item;
    Else:
        Then G must have a cycle, so shift bundles around some cycle;

Theorem (Lipton, Markakis, Mossel, Saberi, 2004)

The Envy Cycle Elimination algorithm computes an EF1 allocation

Proof. The following claim holds inductively:
At every iteration, all edges in the envy graph are only up to one good.

Algorithm 2: Envy Cycle Elimination

While there are some unallocated items:
    Let G be the envy graph on the set of agents, where $i \to j$ whenever i envies j;
    If G has a source vertex i:
        Give i the next (arbitrarily selected) item;
    Else:
        Then G must have a cycle, so shift bundles around some cycle;

Theorem (Lipton, Markakis, Mossel, Saberi, 2004)

The Envy Cycle Elimination algorithm computes an EF1 allocation

Proof. The following claim holds inductively:
At every iteration, all edges in the envy graph are only up to one good.

- If case: The only new edges are towards i, and can be eliminated by removing the newly-allocated good.

Algorithm 2: Envy Cycle Elimination

While there are some unallocated items:
    Let G be the envy graph on the set of agents, where $i \to j$ whenever i envies j;
    If G has a source vertex i:
        Give i the next (arbitrarily selected) item;
    Else:
        Then G must have a cycle, so shift bundles around some cycle;

Theorem (Lipton, Markakis, Mossel, Saberi, 2004)

The Envy Cycle Elimination algorithm computes an EF1 allocation

Proof. The following claim holds inductively:
At every iteration, all edges in the envy graph are only up to one good.

- If case: The only new edges are towards i, and can be eliminated by removing the newly-allocated good.
- Else case: The only new bundles any agent could possibly envy are the ones that agents in the cycle previously had, but they like their new bundles better. ■

Algorithm 2: Envy Cycle Elimination

While there are some unallocated items:
    Let G be the envy graph on the set of agents, where $i \to j$ whenever i envies j;
    If G has a source vertex i:
        Give i the next (arbitrarily selected) item;
    Else:
        Then G must have a cycle, so shift bundles around some cycle;

Theorem (Lipton, Markakis, Mossel, Saberi, 2004)

The Envy Cycle Elimination algorithm computes an EF1 allocation ***even for arbitrary monotone valuation functions.***

Proof. The following claim holds inductively:
At every iteration, all edges in the envy graph are only up to one good.

- If case: The only new edges are towards i, and can be eliminated by removing the newly-allocated good.
- Else case: The only new bundles any agent could possibly envy are the ones that agents in the cycle previously had, but they like their new bundles better. ■

Algorithm 3: Max Nash Welfare

Select the allocation A maximizing the social welfare of A:

$$\mathrm{SW}(A) := \sum_{i=1}^{n} v_i(A_i)$$

Algorithm 3: Max Nash Welfare

Select the allocation A maximizing the social welfare of A:

$$\mathrm{SW}(A) := \sum_{i=1}^{n} v_i(A_i)$$

✘ Not EF1

Algorithm 3: Max Nash Welfare

Select the allocation A maximizing the *Nash welfare* of A:

$$\mathrm{NW}(A) := \prod_{i=1}^{n} v_i(A_i)$$

Algorithm 3: Max Nash Welfare

Select the allocation A maximizing the *Nash welfare* of A:

$$\mathrm{NW}(A) := \prod_{i=1}^{n} v_i(A_i)$$

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

Algorithm 3: Max Nash Welfare

Select the allocation A maximizing the *Nash welfare* of A:

$$\mathrm{NW}(A) := \prod_{i=1}^{n} v_i(A_i)$$

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

$\mathrm{NW}(A)$ =
(3 + 10)(6 + 2)(7 + 1)
= 832

Algorithm 3: Max Nash Welfare

Select the allocation A maximizing the *Nash welfare* of A:

$$\mathrm{NW}(A) := \prod_{i=1}^{n} v_i(A_i)$$

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	(3)	3	3	(10)	2	1
Agent 2	1	5	(6)	9	(2)	1
Agent 3	1	(7)	6	9	2	(1)

$\mathrm{NW}(A) =$
$(3 + 10)(6 + 2)(7 + 1)$
$= 832$

Theorem (Caragiannis, Kurokawa, Moulin, Procaccia, Shah, Wang, 2004)

The allocation maximizing Nash welfare satisfies EF1.

Proof that MNW satisfies EF1

Proof. Suppose i envies j by more than one good. We'll show that we can increase the Nash welfare by transferring some good g^* from A_j to A_i.

Proof that MNW satisfies EF1

Proof. Suppose i envies j by more than one good. We'll show that we can increase the Nash welfare by transferring some good g^* from A_j to A_i.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

$$\begin{aligned}\mathrm{NW}(A) &= 13 \cdot 11 \cdot 3 \\ &= 429\end{aligned}$$

Proof that MNW satisfies EF1

Proof. Suppose i envies j by more than one good. We'll show that we can increase the Nash welfare by transferring some good g^* from A_j to A_i.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
$j \rightarrow$ Agent 2	1	5	6	9	2	1
$i \rightarrow$ Agent 3	1	7	6	9	2	1

$$\begin{aligned} \mathrm{NW}(A) &= 13 \cdot 11 \cdot 3 \\ &= 429 \end{aligned}$$

Proof that MNW satisfies EF1

Proof. Suppose i envies j by more than one good. We'll show that we can increase the Nash welfare by transferring some good g^* from A_j to A_i.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
$j \rightarrow$ Agent 2	1	5	6	9	2	1
$i \rightarrow$ Agent 3	1	7	6	9	2	1

$$\begin{aligned}\mathrm{NW}(A) &= 13 \cdot 11 \cdot 3 \\ &= 429\end{aligned}$$

Let $g^* \in A_j$ be the good maximizing the *critical ratio* $v_i(g^*)/v_j(g^*)$.

Proof that MNW satisfies EF1

Proof. Suppose i envies j by more than one good. We'll show that we can increase the Nash welfare by transferring some good g^* from A_j to A_i.

Example

		g_1	$\boldsymbol{g_2}$	g_3	g_4	g_5	g_6
	Agent 1	3	3	3	10	2	1
$j \to$	Agent 2	1	5	6	9	2	1
$i \to$	Agent 3	1	7	6	9	2	1
	CR:		*7/5*	*6/6*			

$$\begin{aligned}\mathrm{NW}(A) &= 13 \cdot 11 \cdot 3 \\ &= 429\end{aligned}$$

Let $g^* \in A_j$ be the good maximizing the *critical ratio* $v_i(g^*)/v_j(g^*)$.

Proof that MNW satisfies EF1

Proof. Suppose i envies j by more than one good. We'll show that we can increase the Nash welfare by transferring some good g^* from A_j to A_i.

Example

	g_1	g_2	g_3	g_4	g_5	g_6
		CR: 7/5	*CR: 6/6*			
Agent 1	3	3	3	10	2	1
$j \rightarrow$ Agent 2	1	5	6	9	2	1
$i \rightarrow$ Agent 3	1	7	6	9	2	1

$$\begin{aligned}\mathrm{NW}(A) &= 13 \cdot 11 \cdot 3 \\ &= 429\end{aligned}$$

Let $g^* \in A_j$ be the good maximizing the *critical ratio* $v_i(g^*)/v_j(g^*)$.

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

$$\begin{aligned}\mathrm{NW}(A) &= 13 \cdot 6 \cdot 10 \\ &= 780\end{aligned}$$

Proof that MNW satisfies EF1, continued

g^* maximizes $\dfrac{v_i(g^*)}{v_j(g^*)}$ over A_j

Proof that MNW satisfies EF1, continued

g^* maximizes $\dfrac{v_i(g^*)}{v_j(g^*)}$ over $A_j \implies \dfrac{v_i(g^*)}{v_j(g^*)} \geq \dfrac{v_i(A_j)}{v_j(A_j)}$

Proof that MNW satisfies EF1, continued

$$g^* \text{ maximizes } \frac{v_i(g^*)}{v_j(g^*)} \text{ over } A_j \implies \frac{v_i(g^*)}{v_j(g^*)} \geq \frac{v_i(A_j)}{v_j(A_j)} \implies v_i(g^*)v_j(A_j) \geq v_i(A_j)v_j(g^*)$$

Proof that MNW satisfies EF1, continued

g^* maximizes $\dfrac{v_i(g^*)}{v_j(g^*)}$ over $A_j \implies \dfrac{v_i(g^*)}{v_j(g^*)} \geq \dfrac{v_i(A_j)}{v_j(A_j)} \implies v_i(g^*)v_j(A_j) \geq v_i(A_j)v_j(g^*)$

Let A' be the new allocation where we transfer g^* from A_j to A_i. Then:

$\mathrm{NW}(A') =$

Proof that MNW satisfies EF1, continued

$$g^* \text{ maximizes } \frac{v_i(g^*)}{v_j(g^*)} \text{ over } A_j \implies \frac{v_i(g^*)}{v_j(g^*)} \geq \frac{v_i(A_j)}{v_j(A_j)} \implies v_i(g^*)v_j(A_j) \geq v_i(A_j)v_j(g^*)$$

Let A' be the new allocation where we transfer g^* from A_j to A_i. Then:

$$\mathrm{NW}(A') = \left(\prod_{k \notin \{i,j\}} v_k(A_k) \right) \cdot (v_i(A_i) + v_i(g^*)) \cdot (v_j(A_j) - v_j(g^*))$$

Proof that MNW satisfies EF1, continued

$$g^* \text{ maximizes } \frac{v_i(g^*)}{v_j(g^*)} \text{ over } A_j \implies \frac{v_i(g^*)}{v_j(g^*)} \geq \frac{v_i(A_j)}{v_j(A_j)} \implies v_i(g^*)v_j(A_j) \geq v_i(A_j)v_j(g^*)$$

Let A' be the new allocation where we transfer g^* from A_j to A_i. Then:

$$\begin{aligned}
\mathrm{NW}(A') &= \left(\prod_{k \notin \{i,j\}} v_k(A_k)\right) \cdot (v_i(A_i) + v_i(g^*)) \cdot (v_j(A_j) - v_j(g^*)) \\
&= \left(\prod_{k \notin \{i,j\}} v_k(A_k)\right) \left(v_i(A_i)v_j(A_j) + v_i(g^*)v_j(A_j) - v_j(g^*)(v_i(A_i) + v_i(g^*))\right)
\end{aligned}$$

Proof that MNW satisfies EF1, continued

$$g^* \text{ maximizes } \frac{v_i(g^*)}{v_j(g^*)} \text{ over } A_j \implies \frac{v_i(g^*)}{v_j(g^*)} \geq \frac{v_i(A_j)}{v_j(A_j)} \implies v_i(g^*)v_j(A_j) \geq v_i(A_j)v_j(g^*)$$

Let A' be the new allocation where we transfer g^* from A_j to A_i. Then:

$$\begin{aligned}
\mathrm{NW}(A') &= \Big(\prod_{k \notin \{i,j\}} v_k(A_k)\Big) \cdot (v_i(A_i) + v_i(g^*)) \cdot (v_j(A_j) - v_j(g^*)) \\
&= \Big(\prod_{k \notin \{i,j\}} v_k(A_k)\Big) \Big(\underbrace{v_i(A_i)v_j(A_j)} + v_i(g^*)v_j(A_j) - v_j(g^*)(v_i(A_i) + v_i(g^*))\Big)
\end{aligned}$$

Proof that MNW satisfies EF1, continued

$$g^* \text{ maximizes } \frac{v_i(g^*)}{v_j(g^*)} \text{ over } A_j \implies \frac{v_i(g^*)}{v_j(g^*)} \geq \frac{v_i(A_j)}{v_j(A_j)} \implies v_i(g^*)v_j(A_j) \geq v_i(A_j)v_j(g^*)$$

Let A' be the new allocation where we transfer g^* from A_j to A_i. Then:

$$\begin{aligned}
\mathrm{NW}(A') &= \Big(\prod_{k \notin \{i,j\}} v_k(A_k)\Big) \cdot (v_i(A_i) + v_i(g^*)) \cdot (v_j(A_j) - v_j(g^*)) \\
&= \underbrace{\Big(\prod_{k \notin \{i,j\}} v_k(A_k)\Big)\Big(v_i(A_i)v_j(A_j)}_{\mathrm{NW}(A)} + v_i(g^*)v_j(A_j) - v_j(g^*)(v_i(A_i) + v_i(g^*))\Big)
\end{aligned}$$

Proof that MNW satisfies EF1, continued

g^* maximizes $\dfrac{v_i(g^*)}{v_j(g^*)}$ over $A_j \implies \dfrac{v_i(g^*)}{v_j(g^*)} \geq \dfrac{v_i(A_j)}{v_j(A_j)} \implies v_i(g^*)v_j(A_j) \geq v_i(A_j)v_j(g^*)$

Let A' be the new allocation where we transfer g^* from A_j to A_i. Then:

$$\begin{aligned}
\mathrm{NW}(A') &= \Big(\prod_{k \notin \{i,j\}} v_k(A_k)\Big) \cdot (v_i(A_i) + v_i(g^*)) \cdot (v_j(A_j) - v_j(g^*)) \\
&= \Big(\prod_{k \notin \{i,j\}} v_k(A_k)\Big) \Big(\underbrace{v_i(A_i)v_j(A_j)}_{\mathrm{NW}(A)} + \underbrace{v_i(g^*)v_j(A_j) - v_j(g^*)(v_i(A_i) + v_i(g^*))}_{\text{WTS} > 0}\Big)
\end{aligned}$$

Proof that MNW satisfies EF1, continued

$$g^* \text{ maximizes } \frac{v_i(g^*)}{v_j(g^*)} \text{ over } A_j \implies \frac{v_i(g^*)}{v_j(g^*)} \geq \frac{v_i(A_j)}{v_j(A_j)} \implies v_i(g^*)v_j(A_j) \geq v_i(A_j)v_j(g^*)$$

Let A' be the new allocation where we transfer g^* from A_j to A_i. Then:

$$\begin{aligned}
\mathrm{NW}(A') &= \Big(\prod_{k \notin \{i,j\}} v_k(A_k)\Big) \cdot (v_i(A_i) + v_i(g^*)) \cdot (v_j(A_j) - v_j(g^*)) \\
&= \underbrace{\Big(\prod_{k \notin \{i,j\}} v_k(A_k)\Big)\Big(v_i(A_i)v_j(A_j)}_{\mathrm{NW}(A)} + \underbrace{v_i(g^*)v_j(A_j) - v_j(g^*)(v_i(A_i) + v_i(g^*))\Big)}_{\text{WTS} > 0}
\end{aligned}$$

Proof that MNW satisfies EF1, continued

$$g^* \text{ maximizes } \frac{v_i(g^*)}{v_j(g^*)} \text{ over } A_j \implies \frac{v_i(g^*)}{v_j(g^*)} \geq \frac{v_i(A_j)}{v_j(A_j)} \implies v_i(g^*)v_j(A_j) \geq v_i(A_j)v_j(g^*)$$

Let A' be the new allocation where we transfer g^* from A_j to A_i. Then:

$$\begin{aligned}
\mathrm{NW}(A') &= \Big(\prod_{k \notin \{i,j\}} v_k(A_k)\Big) \cdot (v_i(A_i) + v_i(g^*)) \cdot (v_j(A_j) - v_j(g^*)) \qquad \overbrace{\qquad\qquad}^{< v_i(A_j)} \\
&= \underbrace{\Big(\prod_{k \notin \{i,j\}} v_k(A_k)\Big)\Big(v_i(A_i)v_j(A_j)}_{\mathrm{NW}(A)} + \underbrace{v_i(g^*)v_j(A_j) - v_j(g^*)(v_i(A_i) + v_i(g^*))\Big)}_{\text{WTS} > 0}
\end{aligned}$$

Proof that MNW satisfies EF1, continued

$$g^* \text{ maximizes } \frac{v_i(g^*)}{v_j(g^*)} \text{ over } A_j \implies \frac{v_i(g^*)}{v_j(g^*)} \geq \frac{v_i(A_j)}{v_j(A_j)} \implies v_i(g^*)v_j(A_j) \geq v_i(A_j)v_j(g^*)$$

Let A' be the new allocation where we transfer g^* from A_j to A_i. Then:

$$\begin{aligned}
\mathrm{NW}(A') &= \Big(\prod_{k \notin \{i,j\}} v_k(A_k)\Big) \cdot (v_i(A_i) + v_i(g^*)) \cdot (v_j(A_j) - v_j(g^*)) \qquad \overbrace{}^{< v_i(A_j)} \\
&= \underbrace{\Big(\prod_{k \notin \{i,j\}} v_k(A_k)\Big)\big(v_i(A_i)v_j(A_j)}_{\mathrm{NW}(A)} + \underbrace{v_i(g^*)v_j(A_j) - v_j(g^*)(v_i(A_i) + v_i(g^*))\Big)}_{\text{WTS} > 0} \\
&> \Big(\prod_{k \notin \{i,j\}} v_k(A_k)\Big)\Big(v_i(A_i)v_j(A_j) + v_i(g^*)v_j(A_j) - v_i(A_j)v_j(g^*)\Big)
\end{aligned}$$

Proof that MNW satisfies EF1, continued

g^* maximizes $\dfrac{v_i(g^*)}{v_j(g^*)}$ over $A_j \implies \dfrac{v_i(g^*)}{v_j(g^*)} \geq \dfrac{v_i(A_j)}{v_j(A_j)} \implies \boxed{v_i(g^*)v_j(A_j) \geq v_i(A_j)v_j(g^*)}$

Let A' be the new allocation where we transfer g^* from A_j to A_i. Then:

$$\begin{aligned}
\mathrm{NW}(A') &= \Big(\prod_{k \notin \{i,j\}} v_k(A_k)\Big) \cdot (v_i(A_i) + v_i(g^*)) \cdot (v_j(A_j) - v_j(g^*)) \qquad \overbrace{}^{< v_i(A_j)} \\
&= \Big(\prod_{k \notin \{i,j\}} v_k(A_k)\Big) \Big(\underbrace{v_i(A_i)v_j(A_j)}_{\mathrm{NW}(A)} + \underbrace{v_i(g^*)v_j(A_j) - v_j(g^*)(v_i(A_i) + v_i(g^*))}_{\text{WTS} > 0}\Big) \\
&> \Big(\prod_{k \notin \{i,j\}} v_k(A_k)\Big) \Big(v_i(A_i)v_j(A_j) + \boxed{v_i(g^*)v_j(A_j) - v_i(A_j)v_j(g^*)}\Big)
\end{aligned}$$

Proof that MNW satisfies EF1, continued

$$g^* \text{ maximizes } \frac{v_i(g^*)}{v_j(g^*)} \text{ over } A_j \implies \frac{v_i(g^*)}{v_j(g^*)} \geq \frac{v_i(A_j)}{v_j(A_j)} \implies \boxed{v_i(g^*)v_j(A_j) \geq v_i(A_j)v_j(g^*)}$$

Let A' be the new allocation where we transfer g^* from A_j to A_i. Then:

$$\begin{aligned}
\mathrm{NW}(A') &= \Big(\prod_{k \notin \{i,j\}} v_k(A_k)\Big) \cdot (v_i(A_i) + v_i(g^*)) \cdot (v_j(A_j) - v_j(g^*)) \qquad \overbrace{}^{< v_i(A_j)} \\
&= \Big(\prod_{k \notin \{i,j\}} v_k(A_k)\Big) \Big(\underbrace{v_i(A_i)v_j(A_j)}_{\mathrm{NW}(A)} + \underbrace{v_i(g^*)v_j(A_j) - v_j(g^*)(v_i(A_i) + v_i(g^*))}_{\text{WTS} > 0}\Big) \\
&> \Big(\prod_{k \notin \{i,j\}} v_k(A_k)\Big) \Big(v_i(A_i)v_j(A_j) + \boxed{v_i(g^*)v_j(A_j) - v_i(A_j)v_j(g^*)}\Big) \geq \mathrm{NW}(A)
\end{aligned}$$

■

Pareto efficiency

Recall that an allocation is *Pareto efficient* if it is not possible to make all agents better off (weakly for all agents and strictly for at least one).

Pareto efficiency

Recall that an allocation is *Pareto efficient* if it is not possible to make all agents better off (weakly for all agents and strictly for at least one).

For the special case of $n = 2$, which of the following rules always yields a Pareto efficient allocation?

- **Round Robin**
- **Envy Cycle Elimination**
- **Max Nash Welfare**

Respond at:

pollev.com/jtuckerfoltz255 or

bit.ly/jtfpoll or

text jtuckerfoltz255 to 37607

Pareto efficiency

Recall that an allocation is *Pareto efficient* if it is not possible to make all agents better off (weakly for all agents and strictly for at least one).

For the special case of $n = 2$, which of the following rules always yields a Pareto efficient allocation?

- ✗ **Round Robin**
- ✗ **Envy Cycle Elimination**
- ✓ **Max Nash Welfare**

Pareto efficiency

Recall that an allocation is *Pareto efficient* if it is not possible to make all agents better off (weakly for all agents and strictly for at least one).

For the special case of $n = 2$, which of the following rules always yields a Pareto efficient allocation?

- ✗ **Round Robin**
- ✗ **Envy Cycle Elimination**
- ✓ **Max Nash Welfare**

Proof for MNW.
A Pareto improvement would strictly increase one term in the Nash welfare product, and weakly increase all others, giving an overall strictly higher Nash welfare. ■

Pareto efficiency

Recall that an allocation is *Pareto efficient* if it is not possible to make all agents better off (weakly for all agents and strictly for at least one).

For the special case of $n = 2$, which of the following rules always yields a Pareto efficient allocation?

- ✗ **Round Robin**
- ✗ **Envy Cycle Elimination**
- ✓ **Max Nash Welfare**

RR Counterexample

	g_1	g_2	g_3	g_4
Agent 1	6	5	1	1
Agent 2	9	2	1	1

Proof for MNW.
A Pareto improvement would strictly increase one term in the Nash welfare product, and weakly increase all others, giving an overall strictly higher Nash welfare. ■

Pareto efficiency

Recall that an allocation is *Pareto efficient* if it is not possible to make all agents better off (weakly for all agents and strictly for at least one).

For the special case of $n = 2$, which of the following rules always yields a Pareto efficient allocation?

- ✗ **Round Robin**
- ✗ **Envy Cycle Elimination**
- ✓ **Max Nash Welfare**

RR Counterexample

	g_1	g_2	g_3	g_4
Agent 1	6	5	1	1
Agent 2	9	2	1	1

Proof for MNW.
A Pareto improvement would strictly increase one term in the Nash welfare product, and weakly increase all others, giving an overall strictly higher Nash welfare. ■

Pareto efficiency

Recall that an allocation is *Pareto efficient* if it is not possible to make all agents better off (weakly for all agents and strictly for at least one).

For the special case of $n = 2$, which of the following rules always yields a Pareto efficient allocation?

- ✗ **Round Robin**
- ✗ **Envy Cycle Elimination**
- ✓ **Max Nash Welfare**

Proof for MNW.
A Pareto improvement would strictly increase one term in the Nash welfare product, and weakly increase all others, giving an overall strictly higher Nash welfare. ■

RR Counterexample

	g_1	g_2	g_3	g_4
Agent 1	6	5	1	1
Agent 2	9	2	1	1

(7, 3)

Pareto efficiency

Recall that an allocation is *Pareto efficient* if it is not possible to make all agents better off (weakly for all agents and strictly for at least one).

For the special case of $n = 2$, which of the following rules always yields a Pareto efficient allocation?

- ✗ **Round Robin**
- ✗ **Envy Cycle Elimination**
- ✓ **Max Nash Welfare**

Proof for MNW.
A Pareto improvement would strictly increase one term in the Nash welfare product, and weakly increase all others, giving an overall strictly higher Nash welfare. ■

RR Counterexample

	g_1	g_2	g_3	g_4
Agent 1	6	5	1	1
Agent 2	9	2	1	1

○ (7, 3)

◇ (7, 9)

Pareto efficiency

Recall that an allocation is *Pareto efficient* if it is not possible to make all agents better off (weakly for all agents and strictly for at least one).

▸ **For the special case of $n = 2$, which of the following rules always yields a Pareto efficient allocation?**

- ✗ **Round Robin**
- ✗ **Envy Cycle Elimination**
- ✓ **Max Nash Welfare**

Proof for MNW.
A Pareto improvement would strictly increase one term in the Nash welfare product, and weakly increase all others, giving an overall strictly higher Nash welfare. ■

RR Counterexample

	g_1	g_2	g_3	g_4
Agent 1	6	5	1	1
Agent 2	9	2	1	1

○ (7, 3)

◇ (7, 9)

ECE Counterexample

	g_1	g_2	g_3	g_4	g_5
Agent 1	4	3	1	2	2
Agent 2	4	1	3	2	2

Pareto efficiency

Recall that an allocation is *Pareto efficient* if it is not possible to make all agents better off (weakly for all agents and strictly for at least one).

▸ **For the special case of $n = 2$, which of the following rules always yields a Pareto efficient allocation?**

- ✗ **Round Robin**
- ✗ **Envy Cycle Elimination**
- ✓ **Max Nash Welfare**

Proof for MNW.
A Pareto improvement would strictly increase one term in the Nash welfare product, and weakly increase all others, giving an overall strictly higher Nash welfare. ■

RR Counterexample

	g_1	g_2	g_3	g_4
Agent 1	6	5	1	1
Agent 2	9	2	1	1

○ (7, 3)

◇ (7, 9)

ECE Counterexample

	g_1	g_2	g_3	g_4	g_5
Agent 1	4	3	1	2	2
Agent 2	4	1	3	2	2

Pareto efficiency

Recall that an allocation is *Pareto efficient* if it is not possible to make all agents better off (weakly for all agents and strictly for at least one).

▸ **For the special case of $n = 2$, which of the following rules always yields a Pareto efficient allocation?**

- ✗ **Round Robin**
- ✗ **Envy Cycle Elimination**
- ✓ **Max Nash Welfare**

Proof for MNW.
A Pareto improvement would strictly increase one term in the Nash welfare product, and weakly increase all others, giving an overall strictly higher Nash welfare. ■

RR Counterexample

	g_1	g_2	g_3	g_4
Agent 1	6	5	1	1
Agent 2	9	2	1	1

○ (7, 3)

◇ (7, 9)

ECE Counterexample

	g_1	g_2	g_3	g_4	g_5
Agent 1	4	3	1	2	2
Agent 2	4	1	3	2	2

Pareto efficiency

Recall that an allocation is *Pareto efficient* if it is not possible to make all agents better off (weakly for all agents and strictly for at least one).

▸ **For the special case of $n = 2$, which of the following rules always yields a Pareto efficient allocation?**

- ✗ **Round Robin**
- ✗ **Envy Cycle Elimination**
- ✓ **Max Nash Welfare**

Proof for MNW.
A Pareto improvement would strictly increase one term in the Nash welfare product, and weakly increase all others, giving an overall strictly higher Nash welfare. ■

RR Counterexample

	g_1	g_2	g_3	g_4
Agent 1	6	5	1	1
Agent 2	9	2	1	1

○ (7, 3)

◇ (7, 9)

ECE Counterexample

	g_1	g_2	g_3	g_4	g_5
Agent 1	4	3	1	2	2
Agent 2	4	1	3	2	2

Pareto efficiency

Recall that an allocation is *Pareto efficient* if it is not possible to make all agents better off (weakly for all agents and strictly for at least one).

▸ **For the special case of $n = 2$, which of the following rules always yields a Pareto efficient allocation?**

- ✗ **Round Robin**
- ✗ **Envy Cycle Elimination**
- ✓ **Max Nash Welfare**

Proof for MNW.
A Pareto improvement would strictly increase one term in the Nash welfare product, and weakly increase all others, giving an overall strictly higher Nash welfare. ■

RR Counterexample

	g_1	g_2	g_3	g_4
Agent 1	6	5	1	1
Agent 2	9	2	1	1

○ (7, 3)

◇ (7, 9)

ECE Counterexample

	g_1	g_2	g_3	g_4	g_5
Agent 1	4	3	1	2	2
Agent 2	4	1	3	2	2

Pareto efficiency

Recall that an allocation is *Pareto efficient* if it is not possible to make all agents better off (weakly for all agents and strictly for at least one).

For the special case of $n = 2$, which of the following rules always yields a Pareto efficient allocation?

- ✗ **Round Robin**
- ✗ **Envy Cycle Elimination**
- ✓ **Max Nash Welfare**

Proof for MNW.
A Pareto improvement would strictly increase one term in the Nash welfare product, and weakly increase all others, giving an overall strictly higher Nash welfare. ■

RR Counterexample

	g_1	g_2	g_3	g_4
Agent 1	6	5	1	1
Agent 2	9	2	1	1

○ (7, 3)
◇ (7, 9)

ECE Counterexample

	g_1	g_2	g_3	g_4	g_5
Agent 1	4	3	1	2	2
Agent 2	4	1	3	2	2

○ (6, 6)
◇ (7, 7)

Pareto efficiency

Open Question: Is there a polynomial-time algorithm to find a Pareto efficient and EF1 allocation?

For the special case of $n = 2$, which of the following rules always yields a Pareto efficient allocation?

- ✗ **Round Robin**
- ✗ **Envy Cycle Elimination**
- ✓ **Max Nash Welfare**

Proof for MNW.
A Pareto improvement would strictly increase one term in the Nash welfare product, and weakly increase all others, giving an overall strictly higher Nash welfare. ■

RR Counterexample

	g_1	g_2	g_3	g_4
Agent 1	6	5	1	1
Agent 2	9	2	1	1

○ (7, 3)
◇ (7, 9)

ECE Counterexample

	g_1	g_2	g_3	g_4	g_5
Agent 1	4	3	1	2	2
Agent 2	4	1	3	2	2

○ (6, 6)
◇ (7, 7)

EFX

An allocation A is *envy-free up to one good (EF1)* if $\forall\, i, j \in [n], \exists\, g \in A_j$, $v_i(A_i) \geq v_i(A_j \setminus \{g\})$

EFX

An allocation A is *envy-free up to any good (EFX)* if $\forall\ i, j \in [n]$, $\forall\ g \in A_j$, $v_i(A_i) \geq v_i(A_j \setminus \{g\})$

EFX

An allocation A is *envy-free up to any good (EFX)* if $\forall\ i, j \in [n]$, $\forall\ g \in A_j$, $v_i(A_i) \geq v_i(A_j \setminus \{g\})$

Open Question: Do EFX allocations always exist?

EFX

An allocation A is *envy-free up to any good (EFX)* if $\forall\ i, j \in [n]$, $\forall\ g \in A_j$,
$v_i(A_i) \geq v_i(A_j \setminus \{g\})$

Open Question: Do EFX allocations always exist?

*This is widely considered the most important open question in fair division, with ~10 new papers on EFX every year, largely trying to make progress on special cases or relaxations.

EFX

An allocation A is *envy-free up to any good (EFX)* if $\forall\, i, j \in [n]$, $\forall\, g \in A_j$,
$v_i(A_i) \geq v_i(A_j \setminus \{g\})$

Open Question: Do EFX allocations always exist?

*This is widely considered the most important open question in fair division, with ~10 new papers on EFX every year, largely trying to make progress on special cases or relaxations.

EC 2020 Exemplary Theory Paper & Best Student Paper Awards:
"EFX Exists for Three Agents" (Chaudhury, Garg, Mehlhorn)

$n = 4$ is still open!

EFX

An allocation A is *envy-free up to any good (EFX)* if $\forall\, i, j \in [n]$, $\forall\, g \in A_j$,
$v_i(A_i) \geq v_i(A_j \setminus \{g\})$

Open Question: Do EFX allocations always exist?

*This is widely considered the most important open question in fair division, with ~10 new papers on EFX every year, largely trying to make progress on special cases or relaxations.

EC 2020 Exemplary Theory Paper & Best Student Paper Awards:
"EFX Exists for Three Agents" (Chaudhury, Garg, Mehlhorn)

$n = 4$ is still open!

For $0 \leq \alpha \leq 1$, an allocation A is α-*EFX* if $\forall\, i, j \in [n]$, $\forall\, g \in A_j$,
$v_i(A_i) \geq \alpha \cdot v_i(A_j \setminus \{g\})$

EFX is tricky

Recall that RR, ECE, and MNW all produced the following allocation:

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

EFX is tricky

Recall that RR, ECE, and MNW all produced the following allocation:

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

✘ Not EFX!

Agent 3 envies Agent 1 even after removing g_1.

EFX is tricky

Recall that RR, ECE, and MNW all produced the following allocation:

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	10	2	1
Agent 2	1	5	6	9	2	1
Agent 3	1	7	6	9	2	1

✘ Not EFX!

Agent 3 envies Agent 1 even after removing g_1.

Proposition

RR, ECE, and MNW do not saitsfy α-EFX for any $\alpha > 0$.

EFX is tricky

Recall that RR, ECE, and MNW all produced the following allocation:

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	1000	2	1
Agent 2	0.01	5	6	100	2	1
Agent 3	0.01	7	6	100	2	1

✘ Not EFX!

Agent 3 envies Agent 1 even after removing g_1.

Proposition

RR, ECE, and MNW do not saitsfy α-EFX for any $\alpha > 0$.

EFX is tricky

Recall that RR, ECE, and MNW all produced the following allocation:

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	1000	2	1
Agent 2	0.01	5	6	100	2	1
Agent 3	0.01	7	6	100	2	1

✘ Not EFX!

Agent 3 envies Agent 1 even after removing g_1.

Proposition

RR, ECE, and MNW do not saitsfy α-EFX for any $\alpha > 0$.

Theorem (Amanatidis, Markakis, Ntokos, 2020)

There always exists a 0.618-EFX allocation, which can be computed in polynomial time.

EFX is tricky

Recall that RR, ECE, and MNW all produced the following allocation:

Example

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	1000	2	1
Agent 2	0.01	5	6	100	2	1
Agent 3	0.01	7	6	100	2	1

✘ Not EFX!

Agent 3 envies Agent 1 even after removing g_1.

Proposition

RR, ECE, and MNW do not saitsfy α-EFX for any $\alpha > 0$.

$\phi = 1.618\ldots$
is the *Golden Ratio*:
$\frac{1}{\phi} = \phi - 1 = 0.618\ldots$

Theorem (Amanatidis, Markakis, Ntokos, 2020)

There always exists a 0.618-EFX allocation, which can be computed in polynomial time.

Algorithm 4: RR and ECE hybrid

$L \leftarrow \emptyset$;
While there is some agent i without any goods:

Algorithm 4: RR and ECE hybrid

$L \leftarrow \emptyset$;
While there is some agent *i* without any goods:
    Agent *i* chooses to either:
        (a) Take an unallocated good;
        (b) Steal the good from another agent who's not in *L* - then *i* is added to *L*;

Algorithm 4: RR and ECE hybrid

$L \leftarrow \emptyset$;
While there is some agent *i* without any goods:
    Agent *i* chooses to either:
        (a) Take an unallocated good;
        (b) Steal the good from another agent who's not in *L* - then *i* is added to *L*;
Each agent *not* in *L* gets to pick one more unallocated good;

Algorithm 4: RR and ECE hybrid

$L \leftarrow \emptyset$;
While there is some agent *i* without any goods:
    Agent *i* chooses to either:
        (a) Take an unallocated good;
        (b) Steal the good from another agent who's not in *L* - then *i* is added to *L*;
Each agent *not* in *L* gets to pick one more unallocated good;
Run ECE to allocate the rest of the goods;

Algorithm 4: RR and ECE hybrid

$L \leftarrow \emptyset$;
While there is some agent *i* without any goods:
    Agent *i* chooses to either:
        (a) Take an unallocated good;
        (b) Steal the good from another agent who's not in *L* - then *i* is added to *L*;
Each agent *not* in *L* gets to pick one more unallocated good;
Run ECE to allocate the rest of the goods;

Policy: Assume players choose (b) when they can steal a good worth more than ϕ times any unallocated good.

Algorithm 4: RR and ECE hybrid

$L \leftarrow \emptyset$;
While there is some agent *i* without any goods:
    Agent *i* chooses to either:
        (a) Take an unallocated good;
        (b) Steal the good from another agent who's not in *L* - then *i* is added to *L*;
Each agent *not* in *L* gets to pick one more unallocated good;
Run ECE to allocate the rest of the goods;

Policy: Assume players choose (b) when they can steal a good worth more than ϕ times any unallocated good.

Example

$L = \{\}$

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	1000	2	1
Agent 2	0.01	5	6	100	2	1
Agent 3	0.01	7	6	100	2	1

Algorithm 4: RR and ECE hybrid

$L \leftarrow \emptyset$;
While there is some agent *i* without any goods:
    Agent *i* chooses to either:
        (a) Take an unallocated good;
        (b) Steal the good from another agent who's not in *L* - then *i* is added to *L*;
Each agent *not* in *L* gets to pick one more unallocated good;
Run ECE to allocate the rest of the goods;

Policy: Assume players choose (b) when they can steal a good worth more than ϕ times any unallocated good.

Example

$L = \{\}$

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	1000	2	1
Agent 2	0.01	5	6	100	2	1
Agent 3	0.01	7	6	100	2	1

Algorithm 4: RR and ECE hybrid

$L \leftarrow \emptyset$;
While there is some agent *i* without any goods:
    Agent *i* chooses to either:
        (a) Take an unallocated good;
        (b) Steal the good from another agent who's not in *L* - then *i* is added to *L*;
Each agent *not* in *L* gets to pick one more unallocated good;
Run ECE to allocate the rest of the goods;

Policy: Assume players choose (b) when they can steal a good worth more than ϕ times any unallocated good.

Example

$L = \{2\}$

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	**1000**	2	1
Agent 2	**0.01**	5	6	**100**	2	1
Agent 3	**0.01**	7	6	**100**	2	1

Algorithm 4: RR and ECE hybrid

$L \leftarrow \emptyset$;
While there is some agent i without any goods:
    Agent i chooses to either:
        (a) Take an unallocated good;
        (b) Steal the good from another agent who's not in L - then i is added to L;
Each agent *not* in L gets to pick one more unallocated good;
Run ECE to allocate the rest of the goods;

Policy: Assume players choose (b) when they can steal a good worth more than ϕ times any unallocated good.

Example

$L = \{2\}$

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	1000	2	1
Agent 2	0.01	5	6	100	2	1
Agent 3	0.01	7	6	100	2	1

Algorithm 4: RR and ECE hybrid

$L \leftarrow \emptyset$;
While there is some agent i without any goods:
    Agent i chooses to either:
        (a) Take an unallocated good;
        (b) Steal the good from another agent who's not in L - then i is added to L;
Each agent *not* in L gets to pick one more unallocated good;
Run ECE to allocate the rest of the goods;

Policy: Assume players choose (b) when they can steal a good worth more than ϕ times any unallocated good.

Example

$L = \{2\}$

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	1000	2	1
Agent 2	0.01	5	6	100	2	1
Agent 3	0.01	7	6	100	2	1

Algorithm 4: RR and ECE hybrid

$L \leftarrow \emptyset$;
While there is some agent i without any goods:
    Agent i chooses to either:
        (a) Take an unallocated good;
        (b) Steal the good from another agent who's not in L - then i is added to L;
Each agent *not* in L gets to pick one more unallocated good;
Run ECE to allocate the rest of the goods;

Policy: Assume players choose (b) when they can steal a good worth more than ϕ times any unallocated good.

Example

$L = \{2\}$

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	1000	2	1
Agent 2	0.01	5	6	100	2	1
Agent 3	0.01	7	6	100	2	1

Algorithm 4: RR and ECE hybrid

$L \leftarrow \emptyset$;
While there is some agent i without any goods:
    Agent i chooses to either:
        (a) Take an unallocated good;
        (b) Steal the good from another agent who's not in L - then i is added to L;
Each agent *not* in L gets to pick one more unallocated good;
Run ECE to allocate the rest of the goods;

Policy: Assume players choose (b) when they can steal a good worth more than ϕ times any unallocated good.

Example

$L = \{2\}$

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	1000	2	1
Agent 2	0.01	5	6	100	2	1
Agent 3	0.01	7	6	100	2	1

Algorithm 4: RR and ECE hybrid

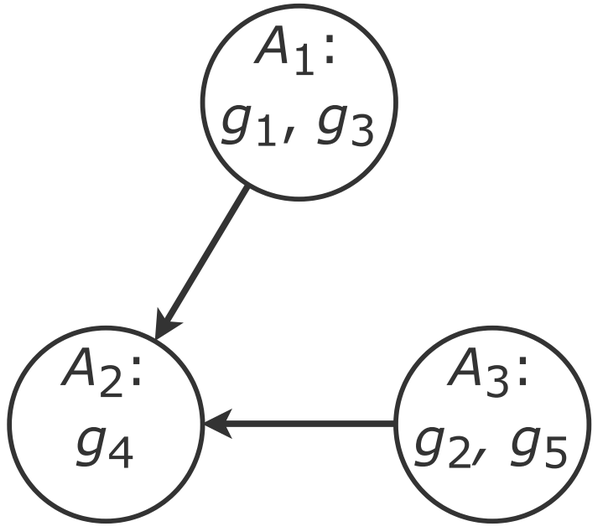
$L \leftarrow \emptyset$;
While there is some agent *i* without any goods:
    Agent *i* chooses to either:
        (a) Take an unallocated good;
        (b) Steal the good from another agent who's not in *L* - then *i* is added to *L*;
Each agent *not* in *L* gets to pick one more unallocated good;
Run ECE to allocate the rest of the goods;

Policy: Assume players choose (b) when they can steal a good worth more than ϕ times any unallocated good.

Example

$L = \{2\}$

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	1000	2	1
Agent 2	0.01	5	6	100	2	1
Agent 3	0.01	7	6	100	2	1



Algorithm 4: RR and ECE hybrid

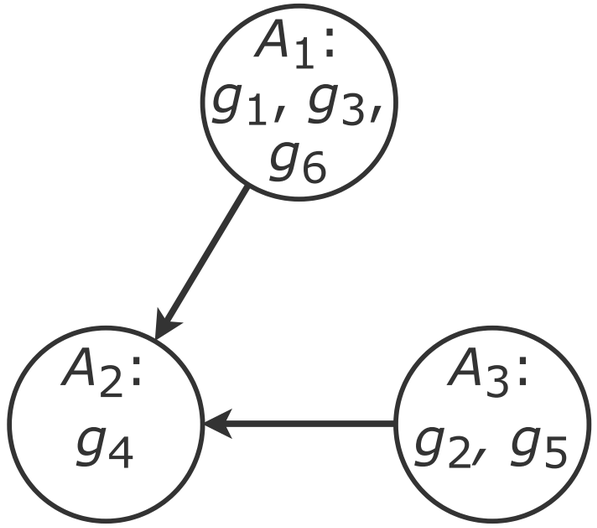
$L \leftarrow \emptyset$;
While there is some agent *i* without any goods:
    Agent *i* chooses to either:
        (a) Take an unallocated good;
        (b) Steal the good from another agent who's not in *L* - then *i* is added to *L*;
Each agent *not* in *L* gets to pick one more unallocated good;
Run ECE to allocate the rest of the goods;

Policy: Assume players choose (b) when they can steal a good worth more than ϕ times any unallocated good.

Example

$L = \{2\}$

	g_1	g_2	g_3	g_4	g_5	g_6
Agent 1	3	3	3	1000	2	1
Agent 2	0.01	5	6	100	2	1
Agent 3	0.01	7	6	100	2	1



Proof of $(\phi - 1)$-EFX, part 1

Lemma 1

Before beginning ECE, the partial allocation is $(\phi - 1)$-EFX.

Proof of $(\phi - 1)$-EFX, part 1

Lemma 1

Before beginning ECE, the partial allocation is $(\phi - 1)$-EFX.

Proof. We don't have to worry about anyone envying agents in L since they each only get one good. Thus the only cases we have to worry about are:

Case 1: An agent $i \in L$ envying an agent $j \notin L$.

Case 2: An agent $i \notin L$ envying an agent $j \notin L$.

Proof of $(\phi - 1)$-EFX, part 1

Lemma 1

Before beginning ECE, the partial allocation is $(\phi - 1)$-EFX.

Proof. We don't have to worry about anyone envying agents in L since they each only get one good. Thus the only cases we have to worry about are:

Case 1: An agent $i \in L$ envying an agent $j \notin L$.
Since i stole their favorite good, they prefer it to both of j's goods, so it is 1-EFX.

Case 2: An agent $i \notin L$ envying an agent $j \notin L$.

Proof of $(\phi - 1)$-EFX, part 1

Lemma 1

Before beginning ECE, the partial allocation is $(\phi - 1)$-EFX.

Proof. We don't have to worry about anyone envying agents in L since they each only get one good. Thus the only cases we have to worry about are:

Case 1: An agent $i \in L$ envying an agent $j \notin L$.
Since i stole their favorite good, they prefer it to both of j's goods, so it is 1-EFX.

Case 2: An agent $i \notin L$ envying an agent $j \notin L$.
Let g_1 be the good agent i recieved in the first RR run, and let g_2 be their favorite of the two goods j has.

Proof of $(\phi - 1)$-EFX, part 1

Lemma 1

Before beginning ECE, the partial allocation is $(\phi - 1)$-EFX.

Proof. We don't have to worry about anyone envying agents in L since they each only get one good. Thus the only cases we have to worry about are:

Case 1: An agent $i \in L$ envying an agent $j \notin L$.
Since i stole their favorite good, they prefer it to both of j's goods, so it is 1-EFX.

Case 2: An agent $i \notin L$ envying an agent $j \notin L$.
Let g_1 be the good agent i recieved in the first RR run, and let g_2 be their favorite of the two goods j has. We know g_2 is worth at most ϕ times as much as g_1 to agent i, otherwise i would have taken/stolen g_2 in the first RR run instead of g_1.

Proof of $(\phi - 1)$-EFX, part 1

Lemma 1

Before beginning ECE, the partial allocation is $(\phi - 1)$-EFX.

Proof. We don't have to worry about anyone envying agents in L since they each only get one good. Thus the only cases we have to worry about are:

Case 1: An agent $i \in L$ envying an agent $j \notin L$.
Since i stole their favorite good, they prefer it to both of j's goods, so it is 1-EFX.

Case 2: An agent $i \notin L$ envying an agent $j \notin L$.
Let g_1 be the good agent i recieved in the first RR run, and let g_2 be their favorite of the two goods j has. We know g_2 is worth at most ϕ times as much as g_1 to agent i, otherwise i would have taken/stolen g_2 in the first RR run instead of g_1. Thus,

$$v_i(g_2) \leq \phi v_i(g_1) \leq \phi v_i(A_i)$$

Proof of $(\phi - 1)$-EFX, part 1

Lemma 1

Before beginning ECE, the partial allocation is $(\phi - 1)$-EFX.

Proof. We don't have to worry about anyone envying agents in L since they each only get one good. Thus the only cases we have to worry about are:

Case 1: An agent $i \in L$ envying an agent $j \notin L$.
Since i stole their favorite good, they prefer it to both of j's goods, so it is 1-EFX.

Case 2: An agent $i \notin L$ envying an agent $j \notin L$.
Let g_1 be the good agent i recieved in the first RR run, and let g_2 be their favorite of the two goods j has. We know g_2 is worth at most ϕ times as much as g_1 to agent i, otherwise i would have taken/stolen g_2 in the first RR run instead of g_1. Thus,

$$v_i(g_2) \leq \phi v_i(g_1) \leq \phi v_i(A_i) \implies v_i(A_i) \geq \frac{1}{\phi} v_i(g_2)$$

Proof of $(\phi - 1)$-EFX, part 1

Lemma 1

Before beginning ECE, the partial allocation is $(\phi - 1)$-EFX.

Proof. We don't have to worry about anyone envying agents in L since they each only get one good. Thus the only cases we have to worry about are:

Case 1: An agent $i \in L$ envying an agent $j \notin L$.
Since i stole their favorite good, they prefer it to both of j's goods, so it is 1-EFX.

Case 2: An agent $i \notin L$ envying an agent $j \notin L$.
Let g_1 be the good agent i recieved in the first RR run, and let g_2 be their favorite of the two goods j has. We know g_2 is worth at most ϕ times as much as g_1 to agent i, otherwise i would have taken/stolen g_2 in the first RR run instead of g_1. Thus,

$$v_i(g_2) \leq \phi v_i(g_1) \leq \phi v_i(A_i) \implies v_i(A_i) \geq \frac{1}{\phi} v_i(g_2) = (\phi - 1) v_i(g_2). \blacksquare$$

Proof of $(\phi - 1)$-EFX, part 2

Lemma 2

Before beginning ECE, every agent values their own bundle at least ϕ times as much as any unallocated good.

Proof of $(\phi - 1)$-EFX, part 2

Lemma 2

Before beginning ECE, every agent values their own bundle at least ϕ times as much as any unallocated good.

Proof. For an agent in L, this follows because they chose to steal a good worth ϕ times as much as any unallocated goods.

Proof of $(\phi - 1)$-EFX, part 2

Lemma 2

Before beginning ECE, every agent values their own bundle at least ϕ times as much as any unallocated good.

Proof. For an agent in L, this follows because they chose to steal a good worth ϕ times as much as any unallocated goods.

For an agent not in L, observe that both of their goods are preferred to any unallocated goods. Thus, they value their own bundle higher by a factor of at least $2 > \phi$. ■

Proof of $(\phi - 1)$-EFX, part 2

Lemma 2

Before beginning ECE, every agent values their own bundle at least ϕ times as much as any unallocated good.

Proof. For an agent in L, this follows because they chose to steal a good worth ϕ times as much as any unallocated goods.

For an agent not in L, observe that both of their goods are preferred to any unallocated goods. Thus, they value their own bundle higher by a factor of at least $2 > \phi$. ■

Proof of Theorem. We now claim that Lemmas 1 and 2 continue to hold after every iteration of ECE.

Proof of $(\phi - 1)$-EFX, part 2

Lemma 2

Before beginning ECE, every agent values their own bundle at least ϕ times as much as any unallocated good.

Proof. For an agent in L, this follows because they chose to steal a good worth ϕ times as much as any unallocated goods.

For an agent not in L, observe that both of their goods are preferred to any unallocated goods. Thus, they value their own bundle higher by a factor of at least $2 > \phi$. ■

Proof of Theorem. We now claim that Lemmas 1 and 2 continue to hold after every iteration of ECE. For Lemma 2, this is obvious, since utilities are increasing throughout ECE.

Proof of $(\phi - 1)$-EFX, part 3

Suppose *A* is a $(\phi - 1)$-EFX partial allocation, then we perform an iteration of ECE to obtain a new partial allocation *A'*. There are two cases:

Case 1: An envy cycle gets rotated.

Case 2: A new good g gets allocated to some agent j.

Proof of $(\phi - 1)$-EFX, part 3

Suppose *A* is a $(\phi - 1)$-EFX partial allocation, then we perform an iteration of ECE to obtain a new partial allocation *A′*. There are two cases:

Case 1: An envy cycle gets rotated.
This doesn't make anyone newly envious by the same reason as in the ECE EF1 proof.

Case 2: A new good g gets allocated to some agent j.

Proof of $(\phi - 1)$-EFX, part 3

Suppose *A* is a $(\phi - 1)$-EFX partial allocation, then we perform an iteration of ECE to obtain a new partial allocation *A'*. There are two cases:

Case 1: An envy cycle gets rotated.
This doesn't make anyone newly envious by the same reason as in the ECE EF1 proof.

Case 2: A new good g gets allocated to some agent j.
Then j is the only new agent we have to worry about other players envying. Consider an arbitrary other agent i.

Proof of $(\phi - 1)$-EFX, part 3

Suppose *A* is a $(\phi - 1)$-EFX partial allocation, then we perform an iteration of ECE to obtain a new partial allocation *A′*. There are two cases:

Case 1: An envy cycle gets rotated.
This doesn't make anyone newly envious by the same reason as in the ECE EF1 proof.

Case 2: A new good g gets allocated to some agent j.
Then j is the only new agent we have to worry about other players envying. Consider an arbitrary other agent i.

$$v_i(A'_j) = v_i(A_j) + v_i(g)$$

Proof of $(\phi - 1)$-EFX, part 3

Suppose *A* is a $(\phi - 1)$-EFX partial allocation, then we perform an iteration of ECE to obtain a new partial allocation *A′*. There are two cases:

Case 1: An envy cycle gets rotated.
This doesn't make anyone newly envious by the same reason as in the ECE EF1 proof.

Case 2: A new good g gets allocated to some agent j.
Then j is the only new agent we have to worry about other players envying. Consider an arbitrary other agent i.

$$v_i(A'_j) = v_i(A_j) + v_i(g)$$

$$\leq v_i(A_i) +$$

Proof of $(\phi - 1)$-EFX, part 3

Suppose *A* is a $(\phi - 1)$-EFX partial allocation, then we perform an iteration of ECE to obtain a new partial allocation *A′*. There are two cases:

Case 1: An envy cycle gets rotated.
This doesn't make anyone newly envious by the same reason as in the ECE EF1 proof.

Case 2: A new good g gets allocated to some agent j.
Then j is the only new agent we have to worry about other players envying. Consider an arbitrary other agent i.

$$\begin{aligned} v_i(A'_j) &= v_i(A_j) + v_i(g) \\ &\leq v_i(A_i) + \frac{v_i(A_i)}{\phi} \end{aligned}$$

Proof of $(\phi - 1)$-EFX, part 3

Suppose *A* is a $(\phi - 1)$-EFX partial allocation, then we perform an iteration of ECE to obtain a new partial allocation *A′*. There are two cases:

Case 1: An envy cycle gets rotated.
This doesn't make anyone newly envious by the same reason as in the ECE EF1 proof.

Case 2: A new good g gets allocated to some agent j.
Then j is the only new agent we have to worry about other players envying. Consider an arbitrary other agent i.

$$\begin{aligned} v_i(A'_j) &= v_i(A_j) + v_i(g) \\ &\leq v_i(A_i) + \frac{v_i(A_i)}{\phi} \\ &= \left(1 + \frac{1}{\phi}\right) v_i(A'_i) \end{aligned}$$

Proof of $(\phi - 1)$-EFX, part 3

Suppose *A* is a $(\phi - 1)$-EFX partial allocation, then we perform an iteration of ECE to obtain a new partial allocation *A′*. There are two cases:

Case 1: An envy cycle gets rotated.
This doesn't make anyone newly envious by the same reason as in the ECE EF1 proof.

Case 2: A new good g gets allocated to some agent j.
Then j is the only new agent we have to worry about other players envying. Consider an arbitrary other agent i.

$$\begin{aligned} v_i(A'_j) &= v_i(A_j) + v_i(g) \\ &\leq v_i(A_i) + \frac{v_i(A_i)}{\phi} \\ &= \left(1 + \frac{1}{\phi}\right) v_i(A'_i) = \phi v_i(A'_i) \end{aligned}$$

Proof of $(\phi - 1)$-EFX, part 3

Suppose *A* is a $(\phi - 1)$-EFX partial allocation, then we perform an iteration of ECE to obtain a new partial allocation *A′*. There are two cases:

Case 1: An envy cycle gets rotated.
This doesn't make anyone newly envious by the same reason as in the ECE EF1 proof.

Case 2: A new good g gets allocated to some agent j.
Then j is the only new agent we have to worry about other players envying. Consider an arbitrary other agent i.

$$
\begin{aligned}
v_i(A'_j) &= v_i(A_j) + v_i(g) \\
&\leq v_i(A_i) + \frac{v_i(A_i)}{\phi} \\
&= \left(1 + \frac{1}{\phi}\right) v_i(A'_i) = \phi v_i(A'_i) \implies v_i(A'_i) \geq \frac{1}{\phi} v_i(A'_j)
\end{aligned}
$$

Proof of $(\phi - 1)$-EFX, part 3

Suppose *A* is a $(\phi - 1)$-EFX partial allocation, then we perform an iteration of ECE to obtain a new partial allocation *A′*. There are two cases:

Case 1: An envy cycle gets rotated.
This doesn't make anyone newly envious by the same reason as in the ECE EF1 proof.

Case 2: A new good g gets allocated to some agent j.
Then j is the only new agent we have to worry about other players envying. Consider an arbitrary other agent i.

$$\begin{aligned} v_i(A'_j) &= v_i(A_j) + v_i(g) \\ &\leq v_i(A_i) + \frac{v_i(A_i)}{\phi} \\ &= \left(1 + \frac{1}{\phi}\right) v_i(A'_i) = \phi v_i(A'_i) \implies v_i(A'_i) \geq \frac{1}{\phi} v_i(A'_j) = (\phi - 1) v_i(A'_j) \end{aligned}$$

Proof of $(\phi - 1)$-EFX, part 3

Suppose *A* is a $(\phi - 1)$-EFX partial allocation, then we perform an iteration of ECE to obtain a new partial allocation *A′*. There are two cases:

Case 1: An envy cycle gets rotated.
This doesn't make anyone newly envious by the same reason as in the ECE EF1 proof.

Case 2: A new good g gets allocated to some agent j.
Then j is the only new agent we have to worry about other players envying. Consider an arbitrary other agent i.

$$\begin{aligned} v_i(A'_j) &= v_i(A_j) + v_i(g) \\ &\leq v_i(A_i) + \frac{v_i(A_i)}{\phi} \\ &= \left(1 + \frac{1}{\phi}\right) v_i(A'_i) = \phi v_i(A'_i) \implies v_i(A'_i) \geq \frac{1}{\phi} v_i(A'_j) = (\phi - 1) v_i(A'_j) \end{aligned}$$

So here we actually get $(\phi - 1)$-EF, which implies $(\phi - 1)$-EFX. ■

Algorithm 5: MNW and ECE hybrid

Step 1: Compute the partial allocation maximizing Nash welfare (product of utilities) with respect to the following utility functions:

$$u_i(A_i) := \begin{cases} \phi \cdot v_i(A_i) & \text{if } |A_i| = 1 \\ v_i(A_i) & \text{if } |A_i| = 2 \\ 0 & \text{if } |A_i| \geq 3 \end{cases}$$

Algorithm 5: MNW and ECE hybrid

Step 1: Compute the partial allocation maximizing Nash welfare (product of utilities) with respect to the following utility functions:

$$u_i(A_i) := \begin{cases} \phi \cdot v_i(A_i) & \text{if } |A_i| = 1 \\ v_i(A_i) & \text{if } |A_i| = 2 \\ 0 & \text{if } |A_i| \geq 3 \end{cases}$$

Step 2: Complete with ECE.

Algorithm 5: MNW and ECE hybrid

Step 1: Compute the partial allocation maximizing Nash welfare (product of utilities) with respect to the following utility functions:

$$u_i(A_i) := \begin{cases} \phi \cdot v_i(A_i) & \text{if } |A_i| = 1 \\ v_i(A_i) & \text{if } |A_i| = 2 \\ 0 & \text{if } |A_i| \geq 3 \end{cases}$$

Step 2: Complete with ECE.

The same proof approach shows that this procedure also finds a $(\phi - 1)$-EFX allocation! If the conclusions of Lemmas 1 and 2 do not hold, then you can increase Nash welfare.