

Fair Allocations I

COMP4418 Knowledge
Representation and Reasoning

SHIVIKA NARANG (SHE/HER)

CSE, UNSW

Previously

Games: Non-cooperative and Cooperative

Matchings:

- Two Sided: One-one and Many-to-one matchings
- One Sided: House Allocation and Kidney Exchange

Today

Allocations: One-sided many-to-one matchings

Efficient Allocations

Fair Allocations

- What does it mean to be fair?
- Do fair allocations exist?
- Can we compute fair allocations?

What and Why Allocations?

Allocation Settings

Week 4: matched students to colleges

- Made pairs of agents belonging to two groups

Week 5: matched agents to houses

- Again pairs from two groups: agents and houses

Why is a house not an agent?

- Does not have preferences (agency)

This week: match agents to bundles of *items*

Allocation Settings

Items:

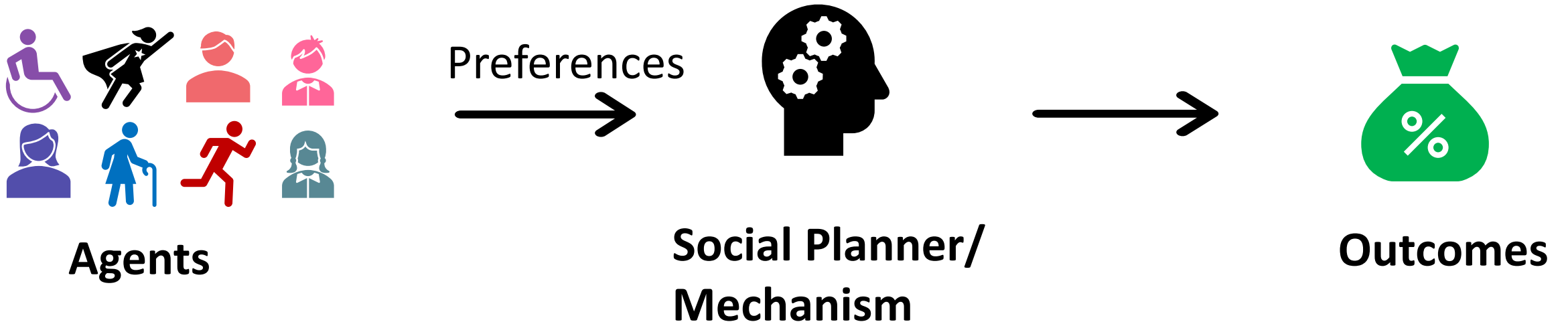
- Have no preferences
- Cannot be split between agents
- Each agent can get multiple items



Motivation:

- Inheritance/asset division
- Splitting food donations between food banks
- Allocating delivery packages to delivery drivers/postal workers

Recall



What are the preferences?

Agent Preferences

Assume:

- I. Agents' happiness depends only on the items allocated to them
- II. Agents have complete preferences
- III. No restrictions on how many items one agent can/must get

How should agents express preferences?

Earlier: Ordinal

- Need to specify an ordering over ALL subsets of items
- Can't achieve much more than PO

Need a more expressive model of preferences: Cardinal

Agent Preferences

Cardinal Valuations: Agents express a *numerical value* for each item/subset of items $v_i(B) \geq 0$.

- Assume $v_i(\emptyset) = 0$






Additive Valuations: each agent i has value v_i for a bundle B of items such that:

$$v_i(B) = \sum_{g \in B} v_i(g)$$

Value for bundle is the sum of value for individual items in bundle

Example



					
Agent/Item	g_1	g_2	g_3	g_4	g_5
1	100	30	5	10	5
2	100	10	5	5	40
3	30	10	50	50	5

Consider above allocation A : $A_1 = \{g_1\}$, $A_2 = \{g_2, g_5\}$, $A_3 = \{g_3, g_4\}$
 $v_1(A_1) = 100$, $v_2(A_2) = 50$, $v_3(A_3) = 100$.

Model

Allocation instance: $\langle N, M, v \rangle$ where

- Set of n agents: N
- Set of m items M
- Valuations of agents $v = (v_i)_{i \in N}$

An allocation $A = (A_1, \dots, A_n)$ is an n -partition of the set of items

- A_i : Bundle allocated to $i \in N$
- For each $i \in N$, $A_i \subseteq M$ --- each agent is allocated a subset of items
- For each $i \neq j$, $A_i \cap A_j = \emptyset$ --- no two agents allocated the same item
- $\bigcup_{i \in N} A_i = M$ --- each item is allocated to some agent

What Type of Allocations are Good?

Efficiency

What would be good allocations?

What properties have we seen before?

- Stability --- needs two sided preferences
- Pareto Optimality









Pareto Optimal (PO): An allocation A is PO if there is NO other allocation A' s.t. $v_i(A_i) \leq v_i(A'_i)$ and for some i^* , $v_{i^*}(A_{i^*}) < v_{i^*}(A'_{i^*})$.

- An allocation that is not Pareto dominated by another.

Pareto Optimality

Do Pareto Optimal allocations always exist?









- Yes
- Multiple allocations may be PO

						
  	Agent/Item	g_1	g_2	g_3	g_4	g_5
1	100	30	5	10	5	
2	100	10	5	5	40	
3	30	10	50	50	5	

Pareto Optimality

Do Pareto Optimal allocations always exist?


- Yes
- Multiple allocations may be PO






						
  	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	100	30	5	10	5
	2	100	10	5	5	40
	3	30	10	50	50	5

Pareto Optimality

Do Pareto Optimal allocations always exist?

- Yes
- Multiple allocations may be PO









					
Agent/Item	g_1	g_2	g_3	g_4	g_5
1	100	30	5	10	5
2	100	10	5	5	40
3	30	10	50	50	5

Pareto Optimality

Do Pareto Optimal allocations always exist?

- Yes
- Multiple allocations may be PO









					
Agent/Item	g_1	g_2	g_3	g_4	g_5
1	100	30	5	10	5
2	100	10	5	5	40
3	30	10	50	50	5

Pareto Optimality

Do Pareto Optimal allocations always exist?

- Yes
- Multiple allocations may be PO



					
Agent/Item	g_1	g_2	g_3	g_4	g_5
1	100	30	5	10	5
2	100	10	5	5	40
3	30	10	50	50	5

Is there a stronger guarantee than
Pareto Optimality?

Doing better than PO









PO alone is not meaningful with cardinal valuations

Can we do anything better?

Utilitarian Social Welfare (USW): Sum of agents' valuations for the allocation

$$\text{USW}(A) = \sum_{i \in N} v_i(A_i)$$









Example

					
Agent/Item	g_1	g_2	g_3	g_4	g_5
 1	100	30	5	10	5
 2	100	10	5	5	40
 3	30	10	50	50	5

Here, $v_1(A_1) = 150$, $v_2(A_2) = 0$, $v_3(A_3) = 0$.

So, here $USW(A) = 150$









Example

					
Agent/Item	g_1	g_2	g_3	g_4	g_5
 1	100	30	5	10	5
 2	100	10	5	5	40
 3	30	10	50	50	5

Here, $v_1(A_1) = 15$, $v_2(A_2) = 40$, and $v_3(A_3) = 40$

So, $USW(A) = 95$.









Example

					
Agent/Item	g_1	g_2	g_3	g_4	g_5
 1	100	30	5	10	5
 2	100	10	5	5	40
 3	30	10	50	50	5

Here, $v_1(A_1) = 130$, $v_2(A_2) = 40$ and $v_3(A_3) = 100$

So, $USW(A) = 270$.

Example

					
Agent/Item	g_1	g_2	g_3	g_4	g_5
 i_1	100	30	5	10	5
 i_2	100	10	5	5	40
 i_3	30	10	50	50	5

Can any other allocation have higher USW?

All items go to an agent with max value for them.

Can't do better.

USW: Existence and Computation

Is a maximum USW allocation guaranteed to **exist**?


- Yes. Finite number of possible allocations: m^n






Can we **find** a maximum USW allocation in polynomial time?

- Yes. Give each item to an agent with maximum value for it
- This relies on additive valuations.

Does maximizing USW make the agents *happy*?

Example











					
Agent/Item	g_1	g_2	g_3	g_4	g_5
1	50	35	0	5	0
2	10	10	5	5	5
3	50	10	6	6	0

Can maximum USW lead to even more disparity?

- Unfortunately, yes.

Example

						
  	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	50	35	7	7	10
	2	49	10	5	5	5
	3	49	10	6	6	0

Maximum USW can be very unfair on an individual level

- Can we do better?

What does it mean to be fair?

Finding Fair Solutions









Max USW might give some agents very low value

- Can we prevent low values?

Egalitarian Social Welfare (ESW): ESW of an allocation is the lowest value received by an agent from it.

$$\text{ESW}(A) = \min_{i \in N} v_i(A_i)$$









Example

					
Agent/Item	g_1	g_2	g_3	g_4	g_5
 1	100	30	5	10	5
 2	100	10	5	5	40
 3	30	10	50	50	5

Here, $v_1(A_1) = 150$, $v_2(A_2) = 0$, $v_3(A_3) = 0$.

So, here $ESW(A) = 0$









Example

					
Agent/Item	g_1	g_2	g_3	g_4	g_5
 1	100	30	5	10	5
 2	100	10	5	5	40
 3	30	10	50	50	5

Here, $v_1(A_1) = 130$, $v_2(A_2) = 40$ and $v_3(A_3) = 100$

So, $ESW(A) = 40$.









Example

					
Agent/Item	g_1	g_2	g_3	g_4	g_5
 1	100	30	5	10	5
 2	100	10	5	5	40
 3	30	10	50	50	5

Here, $v_1(A_1) = 100$, $v_2(A_2) = 55$ and $v_3(A_3) = 50$

So, $ESW(A) = 50$.

Example

					
Agent/Item	g_1	g_2	g_3	g_4	g_5
 1	100	30	5	10	5
 2	100	10	5	5	40
 3	30	10	50	50	5

Here, $v_1(A_1) = 100$, $v_2(A_2) = 50$ and $v_3(A_3) = 100$

So, $ESW(A) = 50$.

Example

Which is better:

$A = (\{g_1\}, \{g_2, g_4, g_5\}, \{g_3\})$ or $A' = (\{g_1\}, \{g_2, g_5\}, \{g_3, g_4\})$

- $v_1(A_1) = 100, v_2(A_2) = 55, v_3(A_3) = 50$ and
- $v_1(A'_1) = 100, v_2(A'_2) = 50, v_3(A'_3) = 100$.

Both A and A' have ESW of 50.

2 prefers A , 3 prefers A' and 1 gets equal value in both.

A' maximizes the second lowest value

Degrees of Egalitarian Welfare

How can we capture these degrees of fairness?

Leximin Tuple: Given allocation A its leximin tuple L_A lists the agents values from A in non-decreasing order.

For $A = (\{g_1\}, \{g_2, g_4, g_5\}, \{g_3\})$, $L_A = (50, 55, 100)$

For $A' = (\{g_1\}, \{g_2, g_5\}, \{g_3, g_4\})$, $L_{A'} = (50, 100, 100)$

Clearly A' is better than A .

Degrees of Egalitarian Welfare









Can we generalize this approach? Want to:

- Maximize ESW
- Out of those that maximize ESW maximize the second lowest value
- Out of these maximize the third lowest value
- So on...

Notation: $L_A[t]$ - t^{th} entry/index in leximin tuple of A

Leximin Domination: Allocation A *leximin dominates* A' if there exist $t \in [n]$ s.t. $L_A[t] > L_{A'}[t]$ and for all $t' < t$, $L_A[t'] = L_{A'}[t']$.

Example






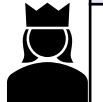


						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	100	30	5	10	5
	2	100	10	5	5	40
	3	30	10	50	50	5

$$A = (\{g_1\}, \{g_2, g_3, g_5\}, \{g_4\}) \quad L_A = (50, 55, 100)$$

$$A' = (\{g_1\}, \{g_2, g_5\}, \{g_3, g_4\}) \quad L_{A'} = (50, 100, 100)$$

A' leximin dominates A

Example

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	50	35	7	7	10
	2	49	10	5	5	5
	3	49	10	6	6	0

$A = (\{g_1\}, \{g_2, g_5\}, \{g_3, g_4\}) \quad L_A = (12, 15, 50)$

$A' = (\{g_2\}, \{g_3, g_4, g_5\}, \{g_1\}) \quad L_{A'} = (15, 35, 49) \quad A' \text{ leximin dominates } A$

Leximin Optimal: An allocation that is not leximin dominated.

Leximin Optimality

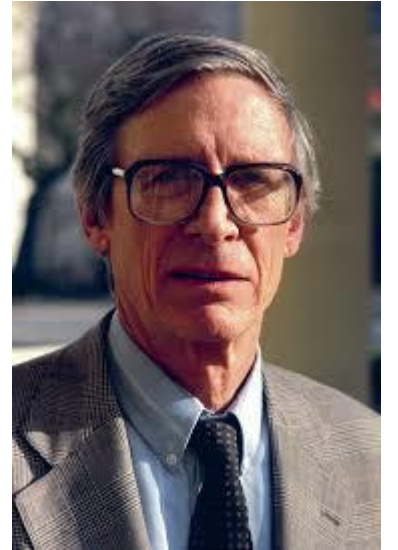
Properties:

- Guaranteed to exist
- Must be Pareto Optimal
- Maximizes ESW

Often called Rawlsian Fairness after John Rawls

Are we done?

- Sadly, no









John Rawls

Drawbacks

Computation: Strongly NP-hard to find a leximin optimal allocation

- Reduction from 3-Partition

Agent Satisfaction: If one agent has very low values compared to others, will get majority of items.

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	10	10	70	70	90
	2	10	10	70	40	50
	3	1	1	2	2	2

Drawbacks

Computation: Strongly NP-hard to find a leximin optimal allocation

- Reduction from 3-Partition

Agent Satisfaction: If one agent has very low values compared to others, will get majority of items.

Verifiability: Hard for agents to check if allocation is fair with limited information.

- Agents typically only know own preferences

Alternate Approach?

What *e/se* can it mean to be fair?

Envy-Freeness









Want allocations where no agent prefers a different bundle.

Definition. An allocation A is envy-free (EF) if for any two agents $i, j \in N$, $v_i(A_i) \geq v_i(A_j)$.

Why envy-free allocations:

- No agent wants to swap bundles
- Can be verified by an agent with only knowledge of their own bundle

Example









					
Agent/Item	g_1	g_2	g_3	g_4	g_5
 1	10	10	70	70	90
 2	10	10	70	40	50
 3	1	1	2	2	2

Is this EF?

$v_1(A_1) = 10, v_1(A_2) = 10, v_1(A_3) = 230$ 1 envies 3

$v_2(A_1) = 10, v_2(A_2) = 10, v_2(A_3) = 160$ 2 envies 3

Example

					
Agent/Item	g_1	g_2	g_3	g_4	g_5
 1	10	10	70	70	90
 2	10	10	70	40	50
 3	1	1	2	2	2

$v_1(A_1) = 90, v_1(A_2) = 70, v_1(A_3) = 90$ 1 has no envy

$v_2(A_1) = 50, v_2(A_2) = 70, v_2(A_3) = 60$ 2 has no envy

$v_3(A_1) = 2, v_3(A_2) = 2, v_3(A_3) = 4$ 3 has no envy

A is envy-free!

Existence and Computation

Do Envy-free allocations always **exist**?

- No. Consider two agents and a car

If items are *divisible* then envy-free allocations always exist

- Divisible items: cake, land, money

Can we **find** envy-free allocations when they do exist?

- Strongly NP-hard. Reduction from 3-Partition

What can we do?

Can we be imperfectly fair?

Relaxing Envy-freeness

Can we ensure we find allocations that are close to envy-free?

How do we define “close to envy-free”?

Approximately Envy-free: Allocation A is α -EF if for any two agents $i, j \in N$, $v_i(A_i) \geq \alpha v_i(A_j)$ for $\alpha \in [0,1]$.

- Verifiable

Existence: Need not exist for any $\alpha > 0$.

Computation: NP-hard.

Counter Example

Fix $\alpha \in (0,1]$.

Choose ϵ s.t. $0 < \epsilon < \alpha$.

No $\alpha - EF$ allocation.



Agent/Item	g_1	g_2
1	1	$\alpha - \epsilon$
2	1	$\alpha - \epsilon$

Relaxing Envy-freeness

How can we actually meaningfully relax envy-freeness?

- Multiplicative approximation didn't work
- Combinatorial?



Eric Budish

Envy-free up to one item: An allocation A is EF1 if for any two agents $i, j \in N$, there exists $g \in A_j$ s.t. $v_i(A_i) \geq v_i(A_j \setminus g)$.

- Proposed by Eric Budish in 2011



Any allocation that is EF is EF1 but not vice versa



The Combinatorial Assignment Problem: Approximate Competitive Equilibrium from Equal Incomes

Eric Budish

Journal of Political Economy, Vol. 119, No. 6 (December 2011), pp. 1061-1103 (43 pages)









Example



 	Agent/Item	g_1	g_2
	1	1	$\alpha - \epsilon$
	2	1	$\alpha - \epsilon$

$v_2(A_1) = 1 > v_2(A_2) = \alpha - \epsilon$ but
 $v_2(A_1 \setminus \{g_1\}) = 0 < v_2(A_2)$. Hence EF1.

Example

					
Agent/Item	g_1	g_2	g_3	g_4	g_5
 1	10	10	70	70	90
 2	10	10	70	40	50
 3	1	1	2	2	2









Is this EF1?

$$v_1(A_1) = 10, v_1(A_2) = 10, v_1(A_3) = 230,$$

$$v_1(A_3 \setminus \{g_3\}) = 160 = v_1(A_3 \setminus \{g_4\}), v_1(A_3 \setminus \{g_5\}) = 140$$

Not EF1 for 1 nor 2

Example

					
Agent/Item	g_1	g_2	g_3	g_4	g_5
 1	10	10	70	70	90
 2	10	10	70	40	50
 3	1	1	2	2	2

$$v_1(A_1) = 80, v_1(A_2) = 70, v_1(A_3) = 100, v_1(A_3 \setminus \{g_5\}) = 10$$

$$v_2(A_1) = 50, v_2(A_2) = 70, v_2(A_3) = 60$$

$$v_3(A_1) = 3, v_3(A_2) = 3, v_3(A_3) = 3$$

This is EF1.

Finding EF1 Allocations

Eric Budish defined EF1 in 2011

Lipton, Markakis, Mossel and Saberi gave an algorithm to find EF1 allocations **in 2004**

- Use envy graphs

On approximately fair allocations of indivisible goods

Authors:  [R. J. Lipton](#),  [E. Markakis](#),  [E. Mossel](#),  [A. Saberi](#)







EC '04: Proceedings of the 5th ACM conference on Electronic commerce • Pages 125 - 131 • [htt](#)

Envy Graphs

For a given allocation, we can define its envy graph

Definition. Given allocation A , it's envy graph $G_A = (V, E)$ where $V = N$ and a directed edge $(i, j) \in E$ if i envies j , that is $v_i(A_i) < v_i(A_j)$.

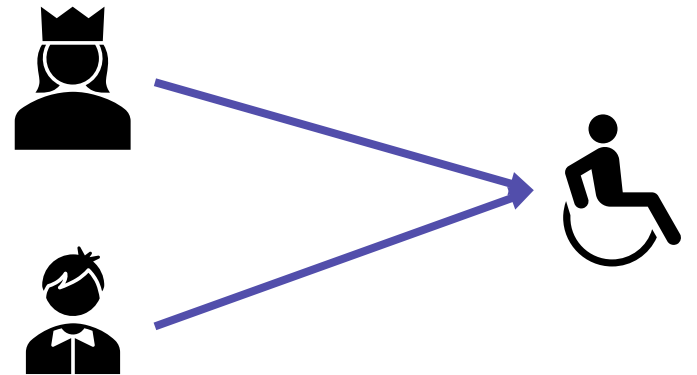
Example

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
1		10	10	70	70	90
2		10	10	70	40	50
3		1	1	2	2	2









$$v_1(A_1) = 10, v_1(A_2) = 10, v_1(A_3) = 230$$

$$v_2(A_1) = 10, v_2(A_2) = 10, v_2(A_3) = 160$$

$$v_3(A_1) = 1, v_3(A_2) = 1, v_3(A_3) = 6$$



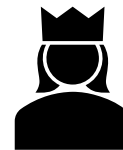
Example

						
  	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	10	10	70	70	90
	2	10	10	70	40	50
	3	1	1	2	2	2

$$v_1(A_1) = 90, v_1(A_2) = 70, v_1(A_3) = 90$$

$$v_2(A_1) = 50, v_2(A_2) = 70, v_2(A_3) = 60$$

$$v_3(A_1) = 2, v_3(A_2) = 2, v_3(A_3) = 4$$



Envy Graphs

For a given allocation, we can define its envy graph

Definition. Given allocation A , it's envy graph $G_A = (V, E)$ where $V = N$ and a directed edge $(i, j) \in E$ if i envies j , that is $v_i(A_i) < v_i(A_j)$.

An envy-free allocation has an “empty” envy graph

- A graph with vertices but no edges

Envy Graphs as a Tool

Envy Graphs

How do we use envy graphs to build EF1 allocations?

Consider a procedure which allocates one item at a time.

- Recall we assume $v_i(g) \geq 0$ for all $i \in N, g \in M$.

Example

Consider $n=3$ agents with *identical values* for $m = 7$ items

$v_i(g_k) = k$ for all $i \in N, k = 1, \dots, m$.

We shall allocate items one at a time and try to maintain EF1



$$A_1 = \{ \}$$

$$v(A_1) = 0$$



$$A_2 = \{ \}$$

$$v(A_2) = 0$$



$$A_3 = \{ \}$$

$$v(A_3) = 0$$



g_1



g_2



g_3



g_4



g_5



g_6



g_7

Example

Consider $n=3$ agents with *identical values* for $m = 7$ items

$$v_i(g_k) = k \text{ for all } i \in N, k = 1, \dots, m.$$

We shall allocate items one at a time and try to maintain EF1



$$A_1 = \{ \}$$

$$v(A_1) = 0$$



$$A_2 = \{ \}$$

$$v(A_2) = 0$$



$$A_3 = \{ \}$$

$$v(A_3) = 0$$

 g_1  g_2  g_3  g_4  g_5  g_6  g_7

Example

Consider $n=3$ agents with *identical values* for $m = 7$ items

$v_i(g_k) = k$ for all $i \in N, k = 1, \dots, m$.

We shall allocate items one at a time and try to maintain EF1



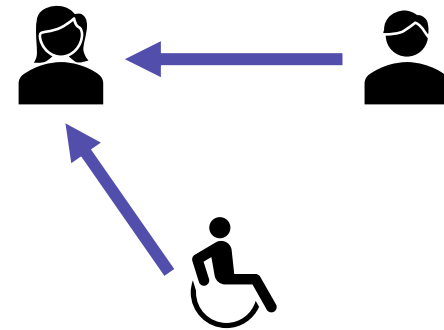
$$A_1 = \{g_4\} \quad v(A_1) = 4$$



$$A_2 = \{\} \quad v(A_2) = 0$$



$$A_3 = \{\} \quad v(A_3) = 0$$



g_1



g_2



g_3



g_4



g_5



g_6



g_7

Example

Consider $n=3$ agents with *identical values* for $m = 7$ items

$v_i(g_k) = k$ for all $i \in N$, $k = 1, \dots, m$.

We shall allocate items one at a time and try to maintain EF1



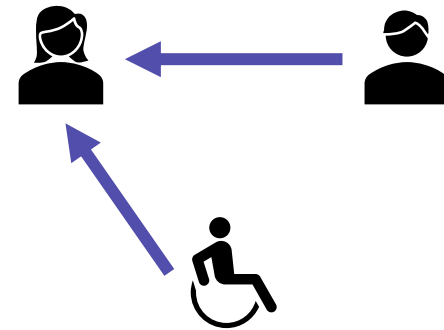
$$A_1 = \{g_4\} \quad v(A_1) = 4$$



$$A_2 = \{\} \quad v(A_2) = 0$$



$$A_3 = \{\} \quad v(A_3) = 0$$



g_1



g_2



g_3



g_4



g_5



g_6



g_7

Example

Consider $n=3$ agents with *identical values* for $m = 7$ items

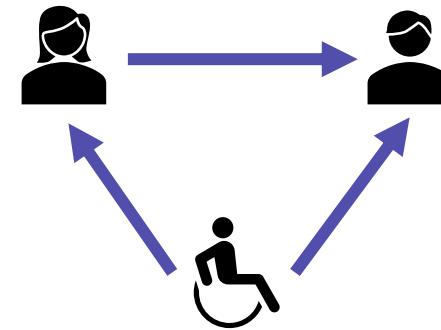
$v_i(g_k) = k$ for all $i \in N$, $k = 1, \dots, m$.

We shall allocate items one at a time and try to maintain EF1

 $A_1 = \{g_4\}$ $v(A_1) = 4$

 $A_2 = \{g_6\}$ $v(A_2) = 6$

 $A_3 = \{\}$ $v(A_3) = 0$



Example

Consider $n=3$ agents with *identical values* for $m = 7$ items

$v_i(g_k) = k$ for all $i \in N, k = 1, \dots, m$.

We shall allocate items one at a time and try to maintain EF1



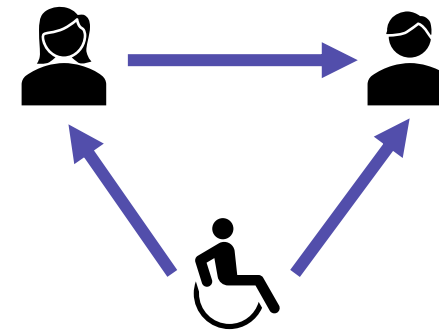
$$A_1 = \{g_4\} \quad v(A_1) = 4$$



$$A_2 = \{g_6\} \quad v(A_2) = 6$$



$$A_3 = \{\} \quad v(A_3) = 0$$



g_1



g_2



g_3



g_4



g_5



g_6






g_7

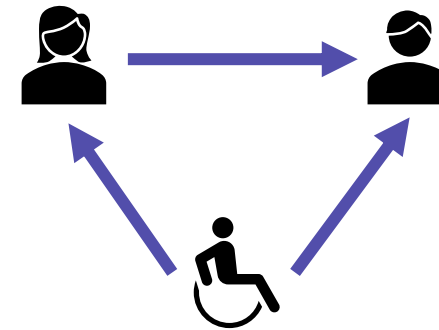
Example

Consider $n=3$ agents with *identical values* for $m = 7$ items

$v_i(g_k) = k$ for all $i \in N, k = 1, \dots, m$.

We shall allocate items one at a time and try to maintain EF1

	$A_1 = \{g_4\}$	$v(A_1) = 4$
	$A_2 = \{g_6\}$	$v(A_2) = 6$
	$A_3 = \{g_1\}$	$v(A_3) = 1$






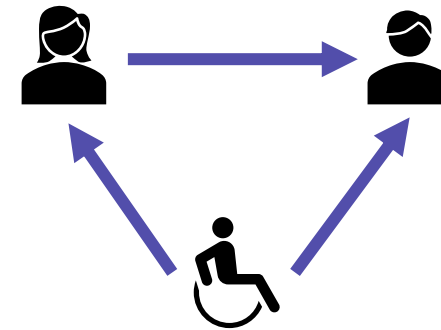
Example

Consider $n=3$ agents with *identical values* for $m = 7$ items

$v_i(g_k) = k$ for all $i \in N, k = 1, \dots, m$.

We shall allocate items one at a time and try to maintain EF1

	$A_1 = \{g_4\}$	$v(A_1) = 4$
	$A_2 = \{g_6\}$	$v(A_2) = 6$
	$A_3 = \{g_1\}$	$v(A_3) = 1$



g_1



g_2



g_3



g_4



g_5



g_6



g_7

Example

Consider $n=3$ agents with *identical values* for $m = 7$ items

$v_i(g_k) = k$ for all $i \in N, k = 1, \dots, m$.

We shall allocate items one at a time and try to maintain EF1



$$A_1 = \{g_4\}$$

$$v(A_1) = 4$$



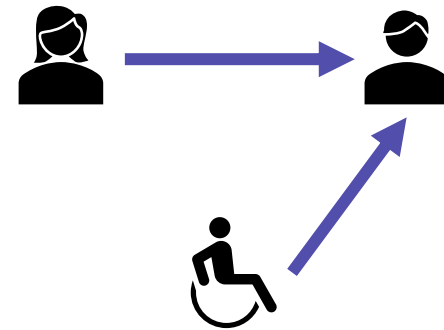
$$A_2 = \{g_6\}$$

$$v(A_2) = 6$$



$$A_3 = \{g_1, g_3\}$$

$$v(A_3) = 4$$



g_1



g_2



g_3



g_4



g_5



g_6



g_7

Example

Consider $n=3$ agents with *identical values* for $m = 7$ items

$v_i(g_k) = k$ for all $i \in N, k = 1, \dots, m$.

We shall allocate items one at a time and try to maintain EF1



$$A_1 = \{g_4\}$$

$$v(A_1) = 4$$



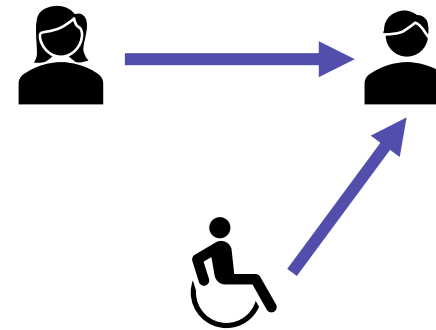
$$A_2 = \{g_6\}$$

$$v(A_2) = 6$$



$$A_3 = \{g_1, g_3\}$$

$$v(A_3) = 4$$



g_1



g_2



g_3



g_4



g_5



g_6






g_7

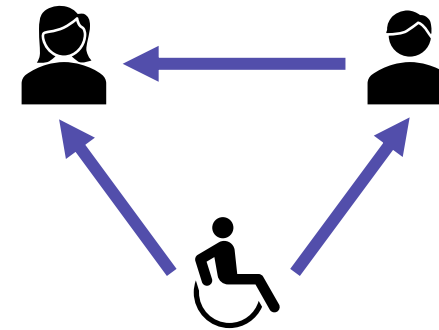
Example

Consider $n=3$ agents with *identical values* for $m = 7$ items

$v_i(g_k) = k$ for all $i \in N, k = 1, \dots, m$.

We shall allocate items one at a time and try to maintain EF1

	$A_1 = \{g_4, g_5\}$	$v(A_1) = 9$
	$A_2 = \{g_6\}$	$v(A_2) = 6$
	$A_3 = \{g_1, g_3\}$	$v(A_3) = 4$



g_1



g_2



g_3



g_4



g_5



g_6






g_7

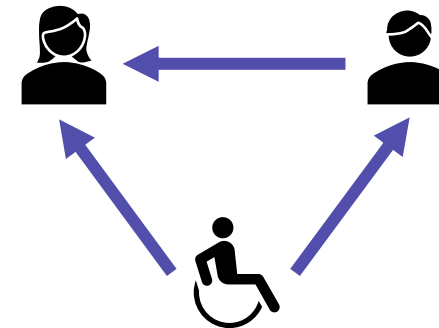
Example

Consider $n=3$ agents with *identical values* for $m = 7$ items

$v_i(g_k) = k$ for all $i \in N, k = 1, \dots, m$.

We shall allocate items one at a time and try to maintain EF1

	$A_1 = \{g_4, g_5\}$	$v(A_1) = 9$
	$A_2 = \{g_6\}$	$v(A_2) = 6$
	$A_3 = \{g_1, g_3\}$	$v(A_3) = 4$



g_1



g_2



g_3



g_4



g_5



g_6






g_7

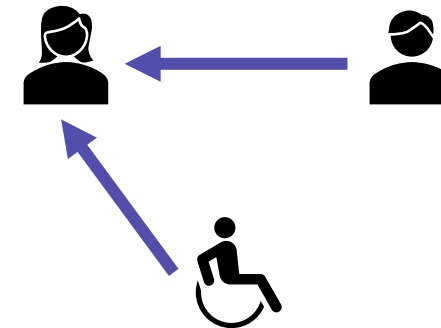
Example

Consider $n=3$ agents with *identical values* for $m = 7$ items

$v_i(g_k) = k$ for all $i \in N, k = 1, \dots, m$.

We shall allocate items one at a time and try to maintain EF1

	$A_1 = \{g_4, g_5\}$	$v(A_1) = 9$
	$A_2 = \{g_6\}$	$v(A_2) = 6$
	$A_3 = \{g_1, g_3, g_2\}$	$v(A_3) = 6$



g_1



g_2



g_3



g_4



g_5



g_6






g_7

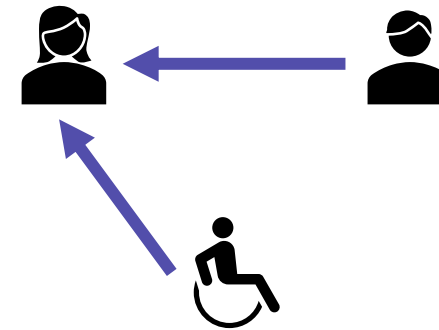
Example

Consider $n=3$ agents with *identical values* for $m = 7$ items

$v_i(g_k) = k$ for all $i \in N, k = 1, \dots, m$.

We shall allocate items one at a time and try to maintain EF1

	$A_1 = \{g_4, g_5\}$	$v(A_1) = 9$
	$A_2 = \{g_6\}$	$v(A_2) = 6$
	$A_3 = \{g_1, g_3, g_2\}$	$v(A_3) = 6$



g_1



g_2



g_3



g_4



g_5



g_6



g_7

Example

Consider $n=3$ agents with *identical values* for $m = 7$ items

$v_i(g_k) = k$ for all $i \in N, k = 1, \dots, m$.

We shall allocate items one at a time and try to maintain EF1



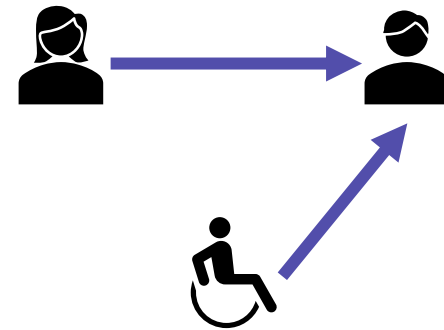
$$A_1 = \{g_4, g_5\} \quad v(A_1) = 9$$



$$A_2 = \{g_6, g_7\} \quad v(A_2) = 13$$



$$A_3 = \{g_1, g_3, g_2\} \quad v(A_3) = 6$$



g_1



g_2



g_3



g_4



g_5



g_6



g_7

Envy Graphs

How do we use envy graphs to build EF1 allocations?








Consider a procedure which allocates one item at a time.

- Recall we assume $v_i(g) \geq 0$ for all $i \in N, g \in M$.

Observations:

- An agent allocated an empty set will not be envied by anyone
- Giving an agent with an incoming edge in envy graph an item may cause EF1 violation
- Safe to give a source of the envy graph an item --- do these always exist?

Example

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
1		100	30	5	10	5
	2	100	10	5	5	40

$$A_1 = \{g_5\}$$

$$A_2 = \{g_4\}$$



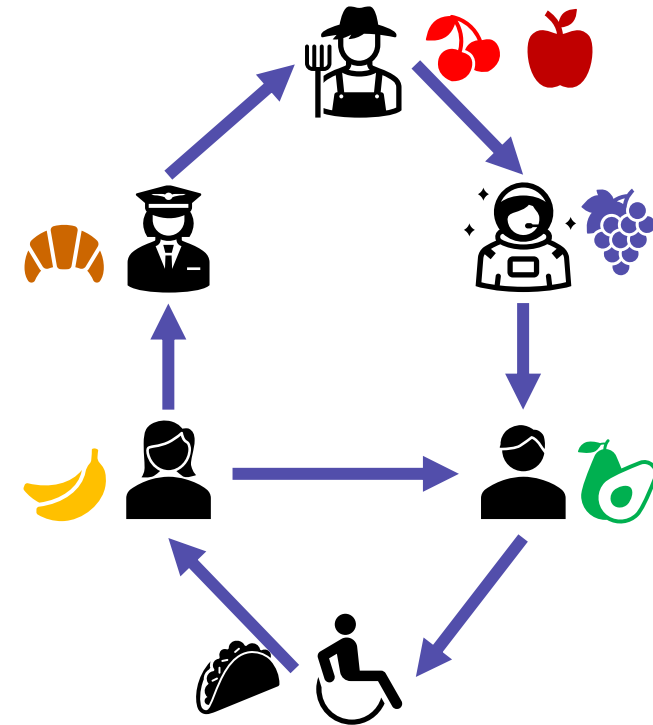
Envy cycle.

No source to allocate to!

Envy Cycle Elimination

What can we do if we encounter an envy cycle?

Resolve the cycle: swap bundles along the cycle



Envy Cycle Elimination

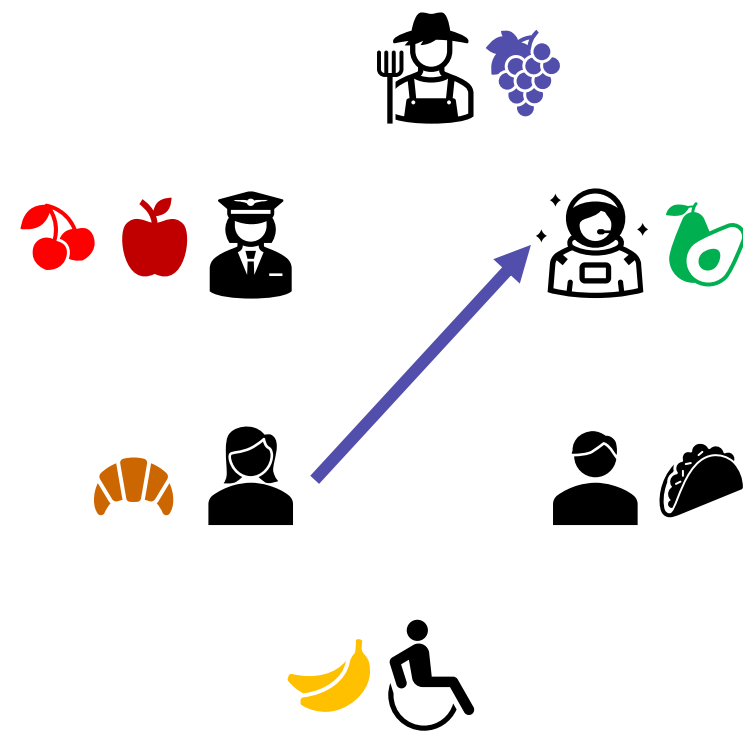
What can we do if we encounter an envy cycle?

Resolve the cycle: swap bundles along the cycle

Can this **increase** the number of envy edges?

- New items are not added, bundles stay same
- Each agent's value can only increase
- No additional edges

Can we now **find** EF1 allocations?



Envy Cycle Elimination: Algorithm

Algorithm: (N, M, v, A)

Let G_A be the envy graph of A

While(G_A contains cycle C)

- For $i \in C$ let $\text{succes}(i)$ be i 's successor in C
- Define A' where $A'_i = \begin{cases} A_i & \text{if } i \notin C \\ A_{\text{succes}(i)} & \text{if } i \in C \end{cases}$
- $A \leftarrow A'$

Return A

Envy Graph Procedure

Algorithm: $\langle N, M, v \rangle$

Initialize unallocated goods $P \leftarrow M$








Initialize allocation A where $A_i \leftarrow \emptyset$

While($P \neq \emptyset$)

- Let G_A be the envy-graph of A
- Choose arbitrary $g \in M$
- Choose $i \leftarrow \text{source}(G_A)$
- $A_i \leftarrow A_i \cup \{g\}$, $P \leftarrow P \setminus \{g\}$
- $A \leftarrow \text{EnvyCycleElimination}(\langle N, M, v \rangle, A)$

Return A

Example








						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	100	30	5	10	5
	2	100	10	5	5	40

$$A_1 = \{\}$$

$$A_2 = \{\}$$



Example








						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	100	30	5	10	5
	2	100	10	5	5	40

$$A_1 = \{\}$$

$$A_2 = \{\}$$



Example

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
1		100	30	5	10	5
	2	100	10	5	5	40

$$A_1 = \{g_5\}$$

$$A_2 = \{\}$$








No envy cycle

$$v_1(A_1) = 5, \quad v_1(A_2) = 0$$

$$v_2(A_1) = 40, \quad v_2(A_2) = 0$$



Example

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
1		100	30	5	10	5
	2	100	10	5	5	40

$$A_1 = \{g_5\}$$








$$A_2 = \{\}$$

$$v_1(A_1) = 5, \quad v_1(A_2) = 0$$

$$v_2(A_1) = 40, \quad v_2(A_2) = 0$$



Example

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
1		100	30	5	10	5
	2	100	10	5	5	40

$$A_1 = \{g_5\}$$

$$A_2 = \{g_4\}$$








$$v_1(A_1) = 5, v_1(A_2) = 10$$

$$v_2(A_1) = 40, v_2(A_2) = 5$$



Resolve envy cycle.

Example

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	100	30	5	10	5
	2	100	10	5	5	40

$$A_1 = \{g_4\}$$

$$A_2 = \{g_5\}$$








No envy cycle.

$$v_1(A_1) = 10, v_1(A_2) = 5$$

$$v_2(A_1) = 5, v_2(A_2) = 40$$



Example

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	100	30	5	10	5
	2	100	10	5	5	40

$$A_1 = \{g_4\}$$








$$A_2 = \{g_5\}$$

$$v_1(A_1) = 10, v_1(A_2) = 5$$

$$v_2(A_1) = 5, v_2(A_2) = 40$$



Example

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	100	30	5	10	5
	2	100	10	5	5	40

$$A_1 = \{g_4, g_3\}$$

$$A_2 = \{g_5\}$$








No envy cycle.

$$v_1(A_1) = 15, v_1(A_2) = 5$$

$$v_2(A_1) = 10, v_2(A_2) = 40$$



Example

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	100	30	5	10	5
	2	100	10	5	5	40

$$A_1 = \{g_4, g_3\}$$








$$A_2 = \{g_5\}$$

$$v_1(A_1) = 15, v_1(A_2) = 5$$

$$v_2(A_1) = 10, v_2(A_2) = 40$$



Example

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
1		100	30	5	10	5
	2	100	10	5	5	40

$$A_1 = \{g_4, g_3\}$$

$$A_2 = \{g_5, g_2\}$$








No envy cycle

$$v_1(A_1) = 15, v_1(A_2) = 35$$

$$v_2(A_1) = 10, v_2(A_2) = 50$$



Example

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	100	30	5	10	5
	2	100	10	5	5	40

$$A_1 = \{g_4, g_3\}$$








$$A_2 = \{g_5, g_2\}$$

$$v_1(A_1) = 15, v_1(A_2) = 35$$

$$v_2(A_1) = 10, v_2(A_2) = 50$$



Example

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	100	30	5	10	5
	2	100	10	5	5	40

$$A_1 = \{g_4, g_3, g_1\} \quad v_1(A_1) = 115, v_1(A_2) = 35$$

$$A_2 = \{g_5, g_2\} \quad v_2(A_1) = 110, v_2(A_2) = 50$$



No more items.

A is EF1

Envy Graph Algorithm

Run Time Analysis:

- Each time an item is allocated adds at most $n - 1$ edges
- At most $m(n - 1)$ edges added across algorithm
- Envy Cycle Elimination removes at least two edges
- Worst case run Envy Cycle Elimination at most $\frac{m(n-1)}{2}$ times
- A cycle can be found in time $O(n^2)$ using Depth First Search

Worst Case run time: $O(mn^3)$.

Envy Graph Algorithm

Thm. Given an allocation instance $\langle N, M, v \rangle$, envy graph algorithm always returns an EF1 allocation for it.

Proof. Let $M = \{g_1, \dots, g_m\}$ s.t. g^t was the t^{th} item to be allocated.

Let i_t be the agent to be **initially** allocated g_t

Will show by induction that EF1 is maintained throughout.

Base Case: $t = 1$. Here $A_{i_1} = \{g_1\}$. EF1 by default. No envy cycles, so no change in allocation. Allocation is EF1

Induction Hypothesis: For $1 < t \leq m$ at the end of envy cycle elimination after allocating g_{t-1} , allocation was EF1

Envy Graph Algorithm

Induction Case: Consider g_t . **Before Envy Cycle Elimination (ECE)**, any EF1 violations could only be towards i_t .

As i_t was a source of the envy graph, when g_t is allocated, for any $j \in N$, $v_j(A_j) \geq v_j(A_{i_t} \setminus \{g_t\})$. Thus, at this point EF1 is satisfied.

After ECE, no agent sees a decrease in value. As the bundles remain same, an EF1 violation after ECE implies an EF1 violation before ECE.

As we had EF1 before ECE, EF1 continues to be satisfied after ECE.

Recap

EF1 allocations:

- Verifiable by agents
- Guaranteed to exist
- Can be found by in polynomial time

Envy Graph Procedure:

- Works for many types of valuations including additive
- Just needs that $v(S \cup g) \geq v(S)$ --- monotonicity
- Runs in time $O(mn^3)$

Can we find EF1 allocations any faster?

Alternate Algorithm for EF1

Round Robin Algorithm

Envy Graph procedure arbitrarily picks the next item to allocate

What if we were smarter?

Round Robin Idea:

- Choose an arbitrary ordering over agents
- Call them one by one.
- At an agents turn, they pick their favourite unassigned item, if any
- Once all agents have picked k items, restart from first agent



Eric Budish

Round robin for EF1 was first studied by Eric Budish

Round Robin Algorithm

Algorithm: (N, M, v)

Initialize set of unassigned items $P \leftarrow M$

Initialize empty allocation A , where $A_i \leftarrow \emptyset$ for all $i \in N$

Choose arbitrary ordering over agents: i_1, \dots, i_n








Set $j \leftarrow 1, t \leftarrow 1$

While $(P \neq \emptyset)$

- $g_t^{i_j} \leftarrow \operatorname{argmax}_{g \in P} v_{i_j}(g)$
- $A_{i_j} \leftarrow A_{i_j} \cup \{g_t^{i_j}\}, P \leftarrow P \setminus \{g_t^{i_j}\}$
- **If** $(j < n)$
 - $j \leftarrow j + 1$
- **Else**
 - $j \leftarrow 1, t \leftarrow t + 1$

Return A








Example

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	100	30	5	10	5
	2	100	10	5	5	40

$$A_1 = \{\}$$

$$A_2 = \{\}$$

Example

 	Agent/Item	 g_1	 g_2	 g_3	 g_4	 g_5
	1	100	30	5	10	5
	2	100	10	5	5	40








$$A_1 = \{g_1\}$$

$$A_2 = \{\}$$

$$v_1(A_1) = 100, \quad v_1(A_2) = 0$$

$$v_2(A_1) = 100, \quad v_2(A_2) = 0$$

Example

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	100	30	5	10	5
	2	100	10	5	5	40








$$A_1 = \{g_1\}$$

$$A_2 = \{\}$$

$$v_1(A_1) = 100, \quad v_1(A_2) = 0$$

$$v_2(A_1) = 100, \quad v_2(A_2) = 0$$

Example

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	100	30	5	10	5
	2	100	10	5	5	40








$$A_1 = \{g_1\}$$

$$A_2 = \{g_5\}$$

$$v_1(A_1) = 100, \quad v_1(A_2) = 5$$

$$v_2(A_1) = 100, \quad v_2(A_2) = 40$$

Example

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
1	100	30	5	10	5	
	2	100	10	5	5	40








$$A_1 = \{g_1\}$$

$$A_2 = \{g_5\}$$

$$v_1(A_1) = 100, \quad v_1(A_2) = 5$$

$$v_2(A_1) = 100, \quad v_2(A_2) = 40$$

Example

						
 	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	100	30	5	10	5
	2	100	10	5	5	40








$$A_1 = \{g_1, g_2\}$$

$$A_2 = \{g_5\}$$

$$v_1(A_1) = 130, \quad v_1(A_2) = 5$$

$$v_2(A_1) = 110, \quad v_2(A_2) = 40$$

Example

 	Agent/Item	 g_1	 g_2	 g_3	 g_4	 g_5
	1	100	30	5	10	5
	2	100	10	5	5	40








$$A_1 = \{g_1, g_2\}$$

$$A_2 = \{g_5\}$$

$$v_1(A_1) = 130, \quad v_1(A_2) = 5$$

$$v_2(A_1) = 110, \quad v_2(A_2) = 40$$

Example

 	Agent/Item	 g_1	 g_2	 g_3	 g_4	 g_5
	1	100	30	5	10	5
	2	100	10	5	5	40








$$A_1 = \{g_1, g_2\}$$

$$A_2 = \{g_5, g_4\}$$

$$v_1(A_1) = 130, \quad v_1(A_2) = 15$$

$$v_2(A_1) = 110, \quad v_2(A_2) = 45$$

Example

 	Agent/Item	 g_1	 g_2	 g_3	 g_4	 g_5
	1	100	30	5	10	5
	2	100	10	5	5	40








$$A_1 = \{g_1, g_2\}$$

$$A_2 = \{g_5, g_4\}$$

$$v_1(A_1) = 130, \quad v_1(A_2) = 15$$

$$v_2(A_1) = 110, \quad v_2(A_2) = 45$$

Example

 	Agent/Item	 g_1	 g_2	 g_3	 g_4	 g_5
	1	100	30	5	10	5
	2	100	10	5	5	40

$$A_1 = \{g_1, g_2, g_3\}$$

$$A_2 = \{g_5, g_4\}$$

$$v_1(A_1) = 135, \quad v_1(A_2) = 15$$

$$v_2(A_1) = 115, \quad v_2(A_2) = 45$$

Round Robin Analysis

Observations:

- Each agent gets either $\left\lfloor \frac{m}{n} \right\rfloor$ or $\left\lceil \frac{m}{n} \right\rceil$ items
- For each $i \in N$, g_t^i is the t^{th} item picked by i .
- For each $i \in N$, $t = 1, \dots, \left\lceil \frac{m}{n} \right\rceil$ we have that $v_i(g_t^i) \geq v_i(g_{t+1}^i)$
- In fact, for any $i, j \in N$ and $t < t'$, we have $v_i(g_t^i) \geq v_i(g_{t'}^j)$

HW: Show that Round robin runs in time $O(mn)$

Round Robin Analysis

Thm. Round Robin algorithm always returns an EF1 allocation.

Proof. Fix any $i, j \in N$ s.t. $i \neq j$. We shall show that i is EF1 towards j
If i (or j) picked $\left\lfloor \frac{m}{n} \right\rfloor$ items let $g_{\left\lfloor \frac{m}{n} \right\rfloor}^i = \emptyset$ and $v_i \left(g_{\left\lfloor \frac{m}{n} \right\rfloor}^i \right) = 0$ (Same for j)

Case 1: i picked before j

we have that $v_i(g_t^i) \geq v_i(g_t^j)$ for all $t = 1, \dots, \left\lfloor \frac{m}{n} \right\rfloor$

Hence $v_i(A_i) = \sum_t v_i(g_t^i) \geq \sum_t v_i(g_t^j) = v_i(A_j)$

Thus, i is EF towards j and hence i is EF1 towards j .

Round Robin Analysis

Case 2: j picked before i

Recall for all $t > 1$, we have that $v_i(g_{t-1}^i) \geq v_i(g_t^j)$

For EF1, consider $A_j \setminus \{g_1^j\}$

$$v_i(A_j \setminus \{g_1^j\}) = \sum_{t>1} v_i(g_t^j) \leq \sum_{t>1} v_i(g_{t-1}^i) \leq \sum_t v_i(g_t^i) = v_i(A_i)$$

Hence, $v_i(A_i) \geq v_i(A_j \setminus \{g_1^j\})$.

Thus, i is EF1 towards j .

Envy-freeness and Beyond

Envy Freeness is:









- Verifiable fairness property
- Well-motivated

Anything else?

- EF satisfies Proportionality

Definition. Allocation A satisfies proportionality (PROP) if for each $i \in N$ we have that
$$v_i(A_i) \geq \frac{1}{n} v_i(M)$$

Example

					
Agent/Item	g_1	g_2	g_3	g_4	g_5
 1	100	10	50	90	50
 2	100	10	10	10	70
 3	100	20	30	50	100

$$v_1(M) = 300 \text{ and } v_1(A_1) = 100$$

$$v_2(M) = 200 \text{ and } v_2(A_2) = 70$$

$$v_3(M) = 300 \text{ and } v_3(A_3) = 100$$

A is PROP

Proportionality and Envy

Proportional Allocations

Thm. Any envy-free allocation satisfies PROP.

Proof. Let A be envy-free

Thus, we have that for each $i \in N$, $v_i(A_i) \geq v_i(A_j)$.

Now $v_i(M) = \sum_{g \in M} v_i(g) = \sum_{j \in N} \sum_{g \in A_j} v_i(g)$

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

Hence, $v_i(A_i) \geq \frac{1}{n} v_i(M)$.

Proportional Allocations

Do PROP allocations always exist?

- No. Same example as EF

Can we find a PROP allocation when it does exist?

- NP-hard to check if a PROP allocation exists. Same reduction as EF

Relaxing to PROP1?

Definition. Allocation A is said to satisfy proportionality up to one item (**PROP1**) if for each $i \in N$ there exists $g \in M$ s.t.

$$v_i(A_i \cup \{g\}) \geq \frac{1}{n} v_i(M).$$

PROP1 via EF1

Thm. Any EF1 allocation is PROP1.

Proof. Let allocation A be EF1. Fix arbitrary $i \in N$.

Choose $g^* \in \operatorname{argmax}_{M \setminus A_i} v_i(g)$.

As A is EF1, for any $j \in N$, there exists g_j s.t. $v_i(A_i) \geq v_i(A_j) - v_i(g_j)$.

Thus, $v_i(A_i) + v_i(g^*) \geq v_i(A_i) + v_i(g_j) \geq v_i(A_j)$.

Consequently, $v_i(M) = \sum_{j \in N} v_i(A_j) \leq \sum_j (v_i(A_i) + v_i(g^*))$

Hence, A is PROP1.

Is Proportionality Interesting?

Why Proportionality

Important real-life goal

- Many countries have laws for proportionally fair solutions








Threshold based notion

- easy to verify even *without knowledge of whole allocation*

Drawback: Exact Proportionality impossible to guarantee with indivisible items

- PROP1 can allow for very bad allocations

Example

						
Agent/Item		g_1	g_2	g_3	g_4	g_5
	1	100	30	5	10	5
	2	100	10	5	5	40

Here, $v_1(A_1) = 150$ and $v_2(A_2) = 0$

This allocation is PROP1

$$v_2(A_2 \cup \{g_1\}) = 100 > \frac{v_2(M)}{2} = 80$$

Improving on PROP1

Is there a more meaningful way to relax Proportionality?

Multiplicative approximations?

$$\alpha\text{-PROP: } v_i(A_i) \geq \alpha \frac{v_i(M)}{n} \text{ for } \alpha \in (0,1]$$

Don't work for the same reason as α -EF.

Alternate Approach to Almost PROP

Recall

Splitting a cake in half without complaints:



Cut and Choose Mechanism: Have one child cut the cake and the other choose first

Maximin Fair Allocations

What if an agent got to partition the items into bundles but got to pick their bundle last?

- Find an allocation which **maximizes** the value of the **minimum** valued bundle








Definition. An agent's maximin fair share is

$$MMS_i = \max_A \min_{j \in N} v_i(A_j)$$

MMS_i is also called agent i 's MMS value/threshold.

Definition. A is maximin fair (**MMS**) if for all $i \in N$, $v_i(A_i) \geq MMS_i$

Example







						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	100	30	5	10	5
	2	100	10	5	5	40

Here,

$$MMS_1 = 50 = v_1(g_2) + v_1(g_3) + v_1(g_4) + v_1(g_5)$$

$$MMS_2 = 60 = v_2(g_2) + v_2(g_3) + v_2(g_4) + v_2(g_5)$$

Example

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	50	35	7	7	10
	2	49	10	5	5	5
3		6	6	49	0	10

$$MMS_1 = 24 = v_1(g_3) + v_1(g_4) + v_1(g_5)$$

$$MMS_2 = 10 = v_2(g_2)$$

$$MMS_3 = 10 = v_3(g_5)$$

Maximin Fair Allocations

MMS allocations were first defined by Eric Budish.

Observations:

- MMS value of an agent doesn't take other agents' valuations into account
- Finding the allocation that gives agent i value MMS_i is equivalent to maximizing ESW on instance where all agents have valuation v_i
- Allocations that define the MMS value for different agents may be different.



Eric Budish

HW: Show that $MMS_i \leq \frac{v_i(M)}{n}$

The Combinatorial Assignment Problem: Approximate Competitive Equilibrium from Equal Incomes

Eric Budish

Journal of Political Economy, Vol. 119, No. 6 (December 2011), pp. 1061-1103 (43 pages)

Maximin Fair Allocations

What affects the MMS value of an agent?

Let $MMS_i = MMS_i(n, M)$ the MMS value when dividing the items in M among n agents.

Observations:

- $MMS_i(n - 1, M) \geq MMS_i(n, M)$
- For any $g \in M$, $MMS_i(n, M \setminus \{g\}) \leq MMS_i(n, M)$.
- What about $MMS_i(n - 1, M \setminus \{g\})$?

Lemma. For any agent $i \in N$ and any item $g \in M$
 $MMS_i(n, M) \leq MMS_i(n - 1, M \setminus \{g\})$.

Maximin Fair Allocations

Lemma. For any agent $i \in N$ and any item $g \in M$
 $MMS_i(n, M) \leq MMS_i(n - 1, M \setminus \{g\})$.

Proof. For agent $i \in N$, let $A = (A_1, \dots, A_n)$ be s.t. for each $j \in N$,
 $v_i(A_j) \geq MMS_i(n, M)$.

WLOG, let $g \in A_1$.

Let A' be an alternate allocation where $A'_1 = \{g\}$, $A'_2 = A_2 \cup (A_1 \setminus \{g\})$ and for all $j \neq 1, 2$, $A'_j = A_j$.

Clearly, $MMS_i(n - 1, M \setminus \{g\}) \geq \min_{j \geq 1} v_i(A'_j) \geq MMS_i(n, M)$.

Maximin Fair Allocations

Do MMS allocations exist?

For $n \geq 3$: MMS allocations need not exist, even with 9 items

For $n = 2$: Can use Cut and Choose

Approximation Algorithms for Computing Maximin Share Allocations

Authors:  [Georgios Amanatidis](#),  [Evangelos Markakis](#),  [Afshin Nikzad](#),  [Amin Saberi](#)

ACM Transactions on Algorithms (TALG), Volume 13, Issue 4 • Article No.: 52, Pages 1 - 28 • <https://doi.org/10.1145/3147173>

Cut and Choose

Algorithm: (N, M, v_1, v_2) where $|N| = 2$

Let $A' = (A'_1, A'_2)$ s.t. $v_1(A'_1) \geq MMS_1$ and $v_1(A'_2) \geq MMS_1$

If $(v_2(A'_1) \geq v_2(A'_2))$

- Set allocation A where $A_1 = A'_2$ and $A_2 = A'_1$

Else

- Set allocation $A = A'$.

Return A

Cut and Choose: Analysis

Thm. Cut and Choose returns an allocation that is MMS for both agents.

Proof. For agent 1, A satisfies MMS by construction.

Let both bundles in A' not satisfy MMS for agent 2.

Thus, $v_2(A_1) < MMS_2$.

Let A^* be an allocation s.t. $v_2(A_1^*) \geq v_2(A_2^*) = MMS_2$

Hence, $v_2(A_2) = v_2(M) - v_2(A_1) > v_2(M) - v_2(A_2^*) = v_2(A_1^*)$.

Consequently, $v_2(A_2) > MMS_2$.

Cut and Choose: Analysis








Run Time: Depends on the time taken to find the value of the MMS threshold.

Finding the MMS value of an agent is NP-hard, even with $n = 2$.

- Reduction from Balanced Partition

Unless $P=NP$, an MMS allocation cannot be found in polytime.

Example








						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	100	30	5	10	5
	2	10	10	5	5	40

Here,

$$MMS_1 = 50 = v_1(g_2) + v_1(g_3) + v_1(g_4) + v_1(g_5)$$

$$MMS_2 = 30 = v_2(g_1) + v_2(g_2) + v_2(g_3) + v_2(g_4)$$

Example

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	100	30	5	10	5
	2	10	10	5	5	40

Consider $A'_1 = \{g_1\}$ and $A'_2 = \{g_2, g_3, g_4, g_5\}$

$$v_2(A'_2) = 60 > MMS_2$$


Approximating MMS

Can we find something close to MMS?


Definition. Allocation A is α -MMS if for each $i \in N$, $v_i(A_i) \geq \alpha MMS_i$ for $\alpha \in (0,1]$.

Best known bounds

- $\left(\frac{3}{4} + \frac{1}{12n}\right)$ -MMS can be found in polynomial time [Garg and Taki 2020]
- There is an instance where no allocation can be better than $\frac{39}{40}$ -MMS [Feige, Sapir, Tauber 2021]



Artificial Intelligence
Volume 300, November 2021, 103547



An improved approximation algorithm for maximin shares ☆, ☆☆

Jugal Garg ✉, Setareh Taki ✉

A Tight Negative Example for MMS Fair Allocations

Authors:  [Uriel Feige](#),  [Ariel Sapir](#),  [Laliv Tauber](#)

Web and Internet Economics: 17th International Conference, WINE 2021, Potsdam, Germany, December 14–17, 2021, Proceedings
Pages 355 - 372 • https://doi.org/10.1007/978-3-030-94676-0_20

Approximating MMS

Goal: Polynomial time algorithm for $\frac{1}{2}$ - MMS








Can we use some tools from earlier?

Recall:

- $MMS_i \leq \frac{v_i(M)}{n}$
- $MMS_i(n, M) \leq MMS_i(n - 1, M \setminus \{g\})$ for any $g \in M$

Idea: If some agent has a single good that gives a good approximation to MMS, can give them the high valued good easily

Example

						
	Agent/Item	g_1	g_2	g_3	g_4	g_5
	1	100	30	5	10	5
	2	10	10	5	5	40

Here,

$$MMS_1 = 50 = v_1(g_2) + v_1(g_3) + v_1(g_4) + v_1(g_5)$$

$$MMS_2 = 30 = v_2(g_1) + v_2(g_2) + v_2(g_3) + v_2(g_4)$$

$$v_1(g_1) = 100 > MMS_1$$

$$v_2(M \setminus \{g_1\}) = 60 > MMS_2$$

Approximating MMS

What about remaining agents with all small valued goods?

- Use round robin?

Idea:

- Let $\beta_i = \frac{v_i(M)}{n}$ for each $i \in N$
- While there exists an agent i and item g s.t. $v_i(g) \geq \frac{\beta_i}{2}$
 - Give g to i and remove them from instance
 - Recalculate β values
- Run round robin on remaining agents and items

Greedy Round Robin

Algorithm: (N, M, v)

Initialize $I \leftarrow N, P \leftarrow M$

Initialize empty allocation A , where $A_i \leftarrow \emptyset$ for all $i \in N$

Set $\beta_i = v_i(M)/n$











While(there exist $i \in I, g \in P$ s.t. $v_i(g) \geq \beta_i/2$)

- $A_i \leftarrow \{g\}$
- $I \leftarrow I \setminus \{i\}, P \leftarrow P \setminus \{g\}$
- For all $j \in I$, set $\beta_j = v_j(P)/|I|$

$A' \leftarrow \text{RoundRobin}(I, P, v)$. For all $i \in I, A_i \leftarrow A'_i$

Return A

Example










						
 Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6
 1	100	10	20	90	50	30
 2	100	20	20	20	20	20
 3	100	23	17	17	24	19

$$v_1(M) = 300 \text{ and } MMS_1 = 100$$

$$v_2(M) = 200 \text{ and } MMS_2 = 40$$

$$v_3(M) = 200 \text{ and } MMS_3 = 47$$

Example

								
	Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6	$\beta_i/2$
	1	100	10	20	90	50	30	50
	2	100	20	20	20	20	20	33
	3	100	23	17	17	24	19	33










Phase 1: Allocate high valued goods

$$A_1 = \{\}$$

$$A_2 = \{\}$$

$$A_3 = \{\}$$

Example

							
 Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6	$\beta_i/2$
1	100	10	20	90	50	30	50
 2	100	20	20	20	20	20	33
 3	100	23	17	17	24	19	33










Phase 1: Allocate high valued goods

$$A_1 = \{g_1\}$$

$$A_2 = \{\}$$

$$A_3 = \{\}$$

Example

								
	Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6	$\beta_i/2$
	1	100	10	20	90	50	30	50
	2	100	20	20	20	20	20	25
	3	100	23	17	17	24	19	25








Phase 2: Round Robin

$$A_1 = \{g_1\}$$

$$A_2 = \{\}$$

$$A_3 = \{\}$$

Example

							
 Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6	$\beta_i/2$
1	100	10	20	90	50	30	50
2	100	20	20	20	20	20	25
3	100	23	17	17	24	19	25










Phase 2: Round Robin

$$A_1 = \{g_1\}$$

$$A_2 = \{g_2\}$$

$$A_3 = \{\}$$

Example

								
	Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6	$\beta_i/2$
	1	100	10	20	90	50	30	50
	2	100	20	20	20	20	20	25
	3	100	23	17	17	24	19	25










Phase 2: Round Robin

$$A_1 = \{g_1\}$$

$$A_2 = \{g_2\}$$

$$A_3 = \{g_5\}$$

Example

								
	Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6	$\beta_i/2$
	1	100	10	20	90	50	30	50
	2	100	20	20	20	20	20	25
	3	100	23	17	17	24	19	25










Phase 2: Round Robin

$$A_1 = \{g_1\}$$

$$A_2 = \{g_2\}$$

$$A_3 = \{g_5\}$$

Example

								
	Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6	$\beta_i/2$
	1	100	10	20	90	50	30	50
	2	100	20	20	20	20	20	25
	3	100	23	17	17	24	19	25











Phase 2: Round Robin

$$A_1 = \{g_1\}$$

$$A_2 = \{g_2, g_3\}$$

$$A_3 = \{g_5\}$$

Example

							
 Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6	$\beta_i/2$
 1	100	10	20	90	50	30	50
 2	100	20	20	20	20	20	25
 3	100	23	17	17	24	19	25











Phase 2: Round Robin

$$A_1 = \{g_1\}$$

$$A_2 = \{g_2, g_3\}$$

$$A_3 = \{g_5\}$$

Example

							
 Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6	$\beta_i/2$
 1	100	10	20	90	50	30	50
 2	100	20	20	20	20	20	25
 3	100	23	17	17	24	19	25











Phase 2: Round Robin

$$A_1 = \{g_1\}$$

$$A_2 = \{g_2, g_3\}$$

$$A_3 = \{g_5, g_6\}$$

Example

							
 Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6	$\beta_i/2$
 1	100	10	20	90	50	30	50
 2	100	20	20	20	20	20	25
 3	100	23	17	17	24	19	25










Phase 2: Round Robin

$$A_1 = \{g_1\}$$

$$A_2 = \{g_2, g_3\}$$

$$A_3 = \{g_5, g_6\}$$

Example

								
	Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6	$\beta_i/2$
	1	100	10	20	90	50	30	50
	2	100	20	20	20	20	20	25
	3	100	23	17	17	24	19	25










Phase 2: Round Robin

$$A_1 = \{g_1\}$$

$$A_2 = \{g_2, g_3, g_4\}$$

$$A_3 = \{g_5, g_6\}$$

Example

							
	Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6
	1	100	10	20	90	50	30
	2	100	20	20	20	20	20
	3	100	23	17	17	24	19

$$v_1(A_1) = 100 > \frac{MMS_1}{2} \text{ where } MMS_1 = 100$$

$$v_2(A_2) = 60 > \frac{MMS_2}{2} \text{ where } MMS_2 = 40$$

$$v_3(A_3) = 43 > \frac{MMS_3}{2} \text{ where } MMS_3 = 50$$

Greedy Round Robin Analysis

Thm. Greedy Round Robin returns an allocation that is $\frac{1}{2} - MMS$.

Proof. Fix arbitrary $i \in N$

Case 1: i was assigned a single item in the while loop

Let β_i, I and P be as they were before allocating $A_i = g$.

Here, we have that $v_i(g) \geq \frac{\beta_i}{2} = \frac{v_i(P)}{2|I|}$

Recall that $\frac{v_i(P)}{|I|} \geq MMS_i(|I|, P) \geq MMS_i(n, M)$.

Thus, $v_i(A_i) = v_i(g) \geq \frac{MMS_i}{2}$

Greedy Round Robin Analysis

Case 2: i was assigned a single item during round robin

Let I and P be as they were during *round robin*.

Let $g_i \in \operatorname{argmax}_{g \in P} v_i(g)$.

As A' is EF1 for (I, P, v) , for any $j \in I$: $v_i(A'_i) + v_i(g_i) \geq v_i(A'_j)$.

Consequently, $v_i(A'_i) \geq \frac{v_i(P)}{|I|} - v_i(g_i) = \frac{\sum_j v_i(A'_j)}{|I|} - v_i(g_i)$.

Recall $v_i(g_i) \leq \frac{v_i(P)}{2|I|}$.

Hence, $v_i(A'_i) \geq \frac{\sum_j v_i(A'_j)}{2|I|} \geq \frac{MMS_i(|I|, P)}{2} \geq \frac{MMS_i(n, M)}{2}$.

Recap

Envy Graph Algorithm/Standard Round Robin: Returns EF1 allocation

- Also PROP1

Greedy Round Robin: Returns $\frac{1}{2}$ —MMS allocation

Reminder

Assignment 2 is due on 25th October

TA help sessions happen **twice a week**:

- Wednesday: 1:00 PM - 2:00 PM (Weeks 7-10) at Quad G031
- Thursday: 10:30 AM - 11:30 AM (Weeks 7-10) online via Teams (check Moodle announcements for link)

Any more queries: feel free to schedule a chat with instructors!

Exercise Sheet Solutions

Exercise 1

Consider an allocation instance $\langle N, M, v \rangle$ where agents have identical valuations. That is, for any $i, j \in N$ and any $g \in M$, $v_i(g) = v_j(g)$. Prove that under identical valuations:

- i. All allocations have the same USW
- ii. An allocation that maximizes ESW will satisfy MMS
- iii. No allocation can have an envy graph with a cycle.

Exercise 1

Consider an allocation instance $\langle N, M, v \rangle$ where agents have identical valuations. That is, for any $i, j \in N$ and any $g \in M$, $v_i(g) = v_j(g)$. Prove that under identical valuations:

- i. All allocations have the same USW

Proof. i) Consider two distinct allocations A and A' . For any item $g \in M$, let i_g and j_g be the agents who receive g in A and A' , resp.

$$\begin{aligned} \text{Now } USW(A) &= \sum_i \sum_{g \in A_i} v_i(g) = \sum_g v_{i_g}(g) \\ &= \sum_g v_{j_g}(g) = \sum_j \sum_{g \in A'_j} v_i(g) = USW(A'). \end{aligned}$$

As A and A' were two arbitrary allocations we have that all allocations have the same USW

Exercise 1

Consider an allocation instance $\langle N, M, v \rangle$ where agents have identical valuations. That is, for any $i, j \in N$ and any $g \in M$, $v_i(g) = v_j(g)$. Prove that under identical valuations:

- i. All allocations have the same USW
- ii. An allocation that maximizes ESW will satisfy MMS

Proof. ii) Consider an allocation A with maximum ESW.

As agents are identical, for any $i \in N$,

$$ESW(A) = \min_{j \in N} v_j(A_j) = \min_{j \in N} v_i(A_j) \leq v_i(A_i)$$

That is, the ESW of an allocation will be i 's value for the worst bundle in A . Thus, a maximum ESW allocation must be MMS for i .

Exercise 1

Proof.

iii) Let A be an arbitrary allocation. Recall that i envies j only if $v_i(A_i) < v_i(A_j)$.

As agents are identical, $v_i(A_i) < v_i(A_j) \Rightarrow v_j(A_i) < v_j(A_j)$.

Consequently, if i envies j and j envies j' then i envies j' .

Thus, an envy cycle i_1, \dots, i_t implies

$$v_{i_1}(A_{i_1}) < v_{i_1}(A_{i_2}) < \dots < v_{i_1}(A_{i_t}) < v_{i_1}(A_{i_1}).$$

This is not possible. Thus, an envy cycle cannot exist.

Exercise 2

Give an example of an instance with indivisible items where:

- i. An envy free allocation does not exist, but a proportional allocation does.
- ii. A PROP allocation exists that is not EF, even though an EF allocation exists for the instance.
- iii. An allocation that is simultaneously PROP and EF1 exists
- iv. An EF1 allocation exists such that its envy graph has a cycle.

Exercise 2

Give an example of an instance with indivisible items where:

- An envy free allocation does not exist, but a proportional allocation does.

Proof. Consider the following instance with $n = 3$ and $m = 3$

No EF allocation:

If i does not get g_1 will be envious

Here $PROP_1 = \frac{1}{3}$ and $PROP_2 = PROP_3 = 1$

PROP allocation:

$A_i = \{g_i\}$ for $i \in N$.

Agent/Item	g_1	g_2	g_3
1	1	0	0
2	2	1	0
3	2	0	1

Exercise 2

Give an example of an instance with indivisible items where:

- A PROP allocation exists that is not EF, even though an EF allocation exists for the instance.

Proof. Consider the following instance with $n = 3$ and $m = 4$

Here $PROP_1 = \frac{2}{3}$ and $PROP_2 = PROP_3 = 1$

PROP not EF allocation

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

EF allocation:

$$A'_1 = \{g_3, g_4\}, A_2 = \{g_1\}, A_3 = \{g_2\}$$

Agent/Item	g_1	g_2	g_3	g_4
1	1	0	1	0
2	1	1	1	0
3	1	1	0	1

Exercise 2

Give an example of an instance with indivisible items where:

- An allocation that is simultaneously PROP and EF1 exists

Proof. Consider the following instance with $n = 3$ and $m = 3$

Here $PROP_1 = \frac{1}{3}$ and $PROP_2 = PROP_3 = 1$

PROP and EF1 allocation:

$A_i = \{g_i\}$ for $i \in N$.

Agent/Item	g_1	g_2	g_3
1	1	0	0
2	2	1	0
3	2	0	1

Exercise 2

Give an example of an instance with indivisible items where:

- An EF1 allocation exists such that its envy graph has a cycle.

Proof. Consider the following instance with $n = 3$ and $m = 3$

EF1 allocation with envy cycle:

$$A_i = \{g_i\} \text{ for } i \in N.$$

Agent/Item	g_1	g_2	g_3
1	1	0	2
2	2	1	0
3	0	2	1

Envy cycle: 1 envies 2, 2 envies 3, 3 envies 1

Exercise 3

Give an example of an instance with indivisible items where:

- Every leximin optimal allocation is MMS
- No leximin optimal allocation is MMS
- There exists an MMS allocation that maximizes ESW but isn't leximin optimal
- There is an MMS allocation that has maximum USW

Exercise 3

Give an example of an instance with indivisible items where:

- Every leximin optimal allocation is MMS

Proof. Consider the following instance with $n = 2$ and $m = 6$

Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6
1	1	1	1	1	1	1
2	3	2	2	0	0	0

Here, $MMS_1 = MMS_2 = 3$

Leximin Optimal Allocation: $A_1 = \{g_3, g_4, g_5, g_6\}$, $A_2 = \{g_1, g_2\}$

$L_A = (4, 5)$ where $v_1(A_1) = 4$ and $v_2(A_2) = 5$.

Exercise 3

Give an example of an instance with indivisible items where:

- No leximin optimal allocation is MMS

Proof. Consider the following instance with $n = 2$ and $m = 6$

Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6
1	1	1	1	1	1	1
2	20	20	20	20	20	20

Here, $MMS_1 = 3$ and $MMS_2 = 60$

Any leximin optimal allocation will give agent 2 exactly one item and the rest to agent 1. Cannot satisfy MMS for agent 2.

Exercise 3

Give an example of an instance with indivisible items where:

- There exists an MMS allocation that maximizes ESW but isn't leximin optimal

Proof. Consider the following instance with $n = 2$ and $m = 5$

Here, $MMS_1 = 3$ and $MMS_2 = 2$

Maximum possible $ESW = 3$

MMS and max ESW Allocation:

$A_1 = \{g_1, g_2\}, A_2 = \{g_3, g_4, g_5\}$ where $v_1(A_1) = 3 = v_2(A_2)$

Leximin Optimal Allocation: $A'_1 = \{g_1, g_5\}, A'_2 = \{g_2, g_3, g_4\}$

Agent/Item	g_1	g_2	g_3	g_4	g_5
1	3	0	0	0	3
2	1	1	1	1	1

Exercise 3

Give an example of an instance with indivisible items where:

- There exists an MMS allocation that has maximum USW

Proof. Consider the following instance with $n = 2$ and $m = 5$

Here, $MMS_1 = 4$ and $MMS_2 = 2$

Maximum possible $USW = 9$

MMS and max USW Allocation:

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

where $v_1(A_1) = 6$ and $v_2(A_2) = 3$.

Agent/Item	g_1	g_2	g_3	g_4	g_5
1	3	3	1	1	1
2	1	1	1	1	1

Exercise 4

Consider the following instance with $n = 3$ and $m = 6$

Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6
1	100	10	20	30	50	90
2	100	20	30	20	20	10
3	100	23	17	17	24	19

For this instance, identify:

- Two distinct maximum USW allocations
- A PROP allocation
- Three distinct $\frac{1}{2}$ —MMS allocations

Exercise 4

Consider the following instance with $n = 3$ and $m = 6$. For this instance, identify:

- Two distinct maximum USW allocations

Proof. Consider the following:

$$A_1 = \{g_4, g_5, g_6\},$$

$$A_2 = \{g_3\} \text{ and } A_3 = \{g_1, g_2\}$$

Here each item goes to an agent with max value for it, hence max USW

Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6
1	100	10	20	30	50	90
2	100	20	30	20	20	10
3	100	23	17	17	24	19

Exercise 4

Consider the following instance with $n = 3$ and $m = 6$. For this instance, identify:

- Two distinct maximum USW allocations

Proof. Alternately, consider:

$$A'_1 = \{g_4, g_5, g_6\},$$

$$A'_2 = \{g_1, g_3\} \text{ and } A'_3 = \{g_2\}$$

In this allocation also each item goes to an agent with max value for it, hence max USW

Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6
1	100	10	20	30	50	90
2	100	20	30	20	20	10
3	100	23	17	17	24	19

Exercise 4

Consider the following instance with $n = 3$ and $m = 6$. For this instance, identify:

- A PROP allocation

Proof. Here,

$PROP_1 = 100$ and

$PROP_2 = PROP_3 = 67$

Consider $A_1 = \{g_5, g_6\}$, $A_2 = \{g_2, g_3, g_4\}$, $A_3 = \{g_1\}$ where

$v_1(A_1) = 140$, $v_2(A_2) = 70$ and $v_3(A_3) = 100$.

Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6
1	100	10	20	30	50	90
2	100	20	30	20	20	10
3	100	23	17	17	24	19

Exercise 4

Consider the following instance with $n = 3$ and $m = 6$. For this instance, identify:

- Three distinct $\frac{1}{2}$ – MMS allocations

Proof. Here,

$$MMS_1 = 100 \text{ and}$$

$$MMS_2 = MMS_3 = 50$$

Consider $A_1 = \{g_1\}$, $A_2 = \{g_3\}$, $A_3 = \{g_2, g_4, g_5, g_6\}$ where

$$v_1(A_1) = 100, v_2(A_2) = 30 \text{ and } v_3(A_3) = 83.$$

This is $\frac{1}{2}$ -MMS.

Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6
1	100	10	20	30	50	90
2	100	20	30	20	20	10
3	100	23	17	17	24	19

Exercise 4

Consider the following instance with $n = 3$ and $m = 6$. For this instance, identify:

- Three distinct $\frac{1}{2}$ – MMS allocations

Proof. Here,

$$MMS_1 = 100 \text{ and}$$

$$MMS_2 = MMS_3 = 50$$

Consider $A'_1 = \{g_6\}$, $A'_2 = \{g_2, g_3, g_4, g_5\}$, $A'_3 = \{g_1\}$ where

$$v_1(A_1) = 90, v_2(A_2) = 90 \text{ and } v_3(A_3) = 100.$$

This is $\frac{1}{2}$ -MMS.

Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6
1	100	10	20	30	50	90
2	100	20	30	20	20	10
3	100	23	17	17	24	19

Exercise 4

Consider the following instance with $n = 3$ and $m = 6$. For this instance, identify:

- Three distinct $\frac{1}{2}$ – MMS allocations

Proof. Here,

$$MMS_1 = 100 \text{ and}$$

$$MMS_2 = MMS_3 = 50$$

Consider $A_1^\# = \{g_1\}$, $A_2^\# = \{g_3, g_4, g_6\}$, $A_3^\# = \{g_2, g_5\}$ where

$$v_1(A_1) = 100, v_2(A_2) = 60 \text{ and } v_3(A_3) = 47.$$

This is $\frac{1}{2}$ -MMS.

Agent/Item	g_1	g_2	g_3	g_4	g_5	g_6
1	100	10	20	30	50	90
2	100	20	30	20	20	10
3	100	23	17	17	24	19

Exercise 5

Consider an allocation instance $\langle N, M, v \rangle$ where $m = kn$. Suppose we add an additional constraint that each agent must receive exactly k items. For this setting, prove that:

- There exists an instance where the Envy Graph algorithm fails.
- Round Robin will return an EF1 allocation where each agent receives k items.

Exercise 5

Consider an allocation instance $\langle N, M, v \rangle$ where $m = kn$. Suppose we add an additional constraint that each agent must receive exactly k items. For this setting, prove that:

- There exists an instance where the Envy Graph algorithm fails.

Answer. Envy graph algorithm proceeds by allocating items to a source agent. If the source agent already has k items, the algorithm will result in an invalid allocation for this setting.

Consider the following instance with $n = 2$ and $m = 8$.

Let $v_1(g_1) = 100 = v_2(g_1)$ and for all other g , $v_1(g) = v_2(g) = 1$.

After giving g_1 to say agent 1, the algorithm will not be able to proceed after agent 2 receives 4 items.

Exercise 5

Consider an allocation instance $\langle N, M, v \rangle$ where $m = kn$. Suppose we add an additional constraint that each agent must receive exactly k items. For this setting, prove that:

- Round Robin will return an EF1 allocation where each agent receives k items.

Answer. Round robin always returns an EF1 allocation.

Round Robin by construction gives each agent $\left\lfloor \frac{m}{n} \right\rfloor$ or $\left\lceil \frac{m}{n} \right\rceil$ items.

When $m = kn$, this will mean round robin gives each agent k items.