How To Cut Your Cake ... Approximately

Paper: **Fair and Efficient Cake Division with Connected Pieces (WINE 2019)**

Authors: E R Arunachaleswaran, S Barman, R Kumar, and N Rathi

Speaker : Shreyas Srinivas

May 28, 2024

CISPA Helmholtz Center for Information Security

Understanding the Problem

You have a Cake!

Image by jemastock on Freepik

You have a Cake!

A Computer Scientist's Cake

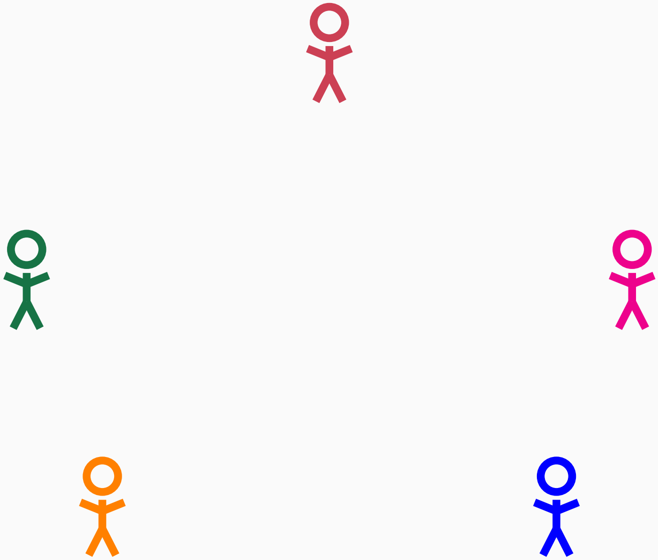
You have a Cake!

- In Mathspeak : $[0, 1] \subset \mathbb{R}$
- Currently unallocated

You have a Cake!

- In Mathspeak : $[0, 1] \subset \mathbb{R}$
- Currently unallocated

There are n Agents!



The Agents love Cake!

$v_r : \mathcal{I}[0,1] \mapsto \mathbb{R}$

$v_g : \mathcal{I} \mapsto \mathbb{R}$

Valuations

$v_m : \mathcal{I}[0,1] \mapsto \mathbb{R}$

$v_o : \mathcal{I}[0,1] \mapsto \mathbb{R}$

$v_b : \mathcal{I}[0,1] \mapsto \mathbb{R}$

The Agents love Cake!

$v_r : \mathcal{I}[0,1] \mapsto \mathbb{R}$

$v_g : \mathcal{I} \mapsto \mathbb{R}$

Heterogeneous

Valuations

$v_m : \mathcal{I}[0,1] \mapsto \mathbb{R}$

$v_o : \mathcal{I}[0,1] \mapsto \mathbb{R}$

$v_b : \mathcal{I}[0,1] \mapsto \mathbb{R}$

The Agents are Reasonable!

$v_m : \mathcal{I}[0,1] \mapsto \mathbb{R}$

Say Hi to Agent Magenta!

The valuation function v_m is

- **Non-negative**
- **Normalized**
- **Divisible**
- **Additive**

The Agents are Reasonable!

$v_m : \mathcal{I}[0,1] \mapsto \mathbb{R}$

Say Hi to Agent Magenta!

The valuation function v_m is

- **Non-negative**
  $\forall I \in \mathcal{I}[0,1], v_m(I) \geq 0$
- **Normalized**
- **Divisible**
- **Additive**

The Agents are Reasonable!

$v_m : \mathcal{I}[0,1] \mapsto \mathbb{R}$

Say Hi to Agent Magenta!

The valuation function v_m is

- Non-negative
- **Normalized**
  $v_m([0,1]) = 1$
- Divisible
- Additive

The Agents are Reasonable!

$v_m : \mathcal{I}[0,1] \mapsto \mathbb{R}$

Say Hi to Agent Magenta!

The valuation function v_m is

- **Non-negative**
- **Normalized**
- **Divisible**
  For every interval $I = [x, y]$
  and $\lambda \in (0,1)$
  There is a subinterval
  $I' = [x, y'] \subseteq I$
  s.t. $v_m(I') = \lambda v_m(I)$
- **Additive**

The Agents are Reasonable!

$v_m : \mathcal{I}[0,1] \mapsto \mathbb{R}$

Say Hi to Agent Magenta!

The valuation function v_m is

- Non-negative
- Normalized
- Divisible
- **Additive**

  For every pair of disjoint intervals I and J, $v_m(I \sqcup J) = v_m(I) + v_m(J)$

The Agents are Reasonable!

$v_m : \mathcal{I}[0,1] \mapsto \mathbb{R}$

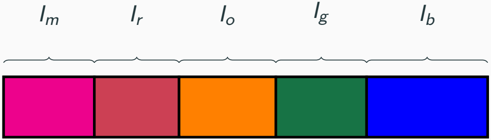
Say Hi to Agent Magenta!

The valuation function v_m is

- **Non-negative**
- **Normalized**
- **Divisible**
- **Additive**

Goal

Split the cake into contiguous pieces. One per agent.

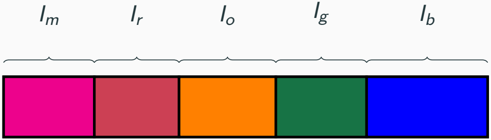


An allocation

- *Fairly*
- *Efficiently*

Goal

Split the cake into contiguous pieces. One per agent.



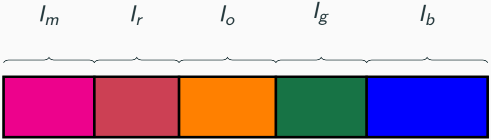
An allocation

- *Fairly*
- *Efficiently*

But How?

Goal

Split the cake into contiguous pieces. One per agent.

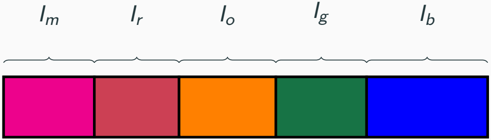


An allocation

- *Fairly* **Envy Freeness (EF)**
- *Efficiently* **Nash Social Welfare (NSW)**

Goal

Split the cake into contiguous pieces. One per agent.



An allocation

- *Fairly* Approximate **Envy Freeness (EF)**
- *Efficiently* Approximate **Nash Social Welfare (NSW)**

A Quick Recap

Fairness Notion : Envy Freeness

- Exact Envy freeness
- c-Additive Approximate Envy Freeness ($c > 0$)
- α-Approximate Envy Freeness ($\alpha > 1$)

Fairness Notion : Envy Freeness

- Exact Envy freeness

$$\forall a, b, v_a(I_a) \geq v_a(I_b)$$

- c-Additive Approximate Envy Freeness ($c > 0$)
- α-Approximate Envy Freeness ($\alpha > 1$)

Fairness Notion : Envy Freeness

- Exact Envy freeness
- c-Additive Approximate Envy Freeness ($c > 0$)

$$\forall a, b, v_a(I_a) \geq v_a(I_b) - c$$

- α-Approximate Envy Freeness ($\alpha > 1$)

Fairness Notion : Envy Freeness

- Exact Envy freeness
- c-Additive Approximate Envy Freeness ($c > 0$)
- α-Approximate Envy Freeness ($\alpha > 1$)

$$\forall a, b, v_a(I_a) \geq \frac{v_a(I_b)}{\alpha}$$

Efficiency Notion : Maximise Nash Social Welfare (NSW)

- NSW of an allocation A:

$$NSW(A) = (\Pi_{a \in Agents} v_a(I_a^A))^{1/n}$$

- Exact NSW objective : Find allocation A such that
- α-NSW approximation objective : Find an allocation A such that

Efficiency Notion : Maximise Nash Social Welfare (NSW)

- NSW of an allocation A:
- Exact NSW objective : Find allocation A such that

$$NSW(A) = \sup_{A' \in Allocations} NSW(A')$$

- α-NSW approximation objective : Find an allocation A such that

Efficiency Notion : Maximise Nash Social Welfare (NSW)

- NSW of an allocation A:
- Exact NSW objective : Find allocation A such that
- α-NSW approximation objective : Find an allocation A such that

$$NSW(A) \geq \frac{1}{\alpha}\left(\sup_{A' \in Allocations} NSW(A')\right)$$

Our Model

Robertson-Webb Model

An oracle with two types of queries

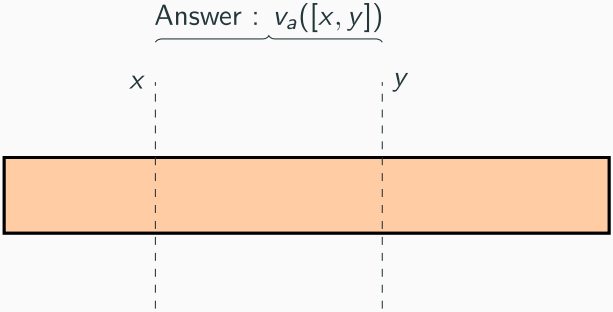
- $eval(a, [x, y])$
- $cut(a, x, \alpha)$

Robertson-Webb Model

An oracle with two types of queries

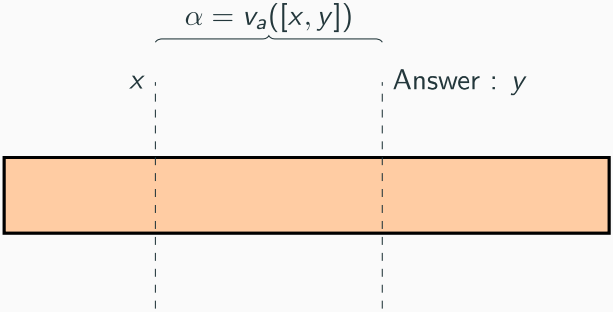
- $\mathit{eval}(a, [x, y])$
- $\mathit{cut}(a, x, \alpha)$

Robertson-Webb Model: Eval Query



$$eval(a, [x, y]) = v_a([x, y])$$

Robertson-Webb Model : Cut Query



Cut query $cut(a, x, \alpha)$ returns y such that $v_a([x, y]) = \alpha$

What's on the Menu Today

- An Efficient algorithm for $(3 + o(1))$-*EF* allocations
- $(3 + o(1))$-*NSW* allocation.
- $(2 + o(1))$ - *EF* allocation.
- Briefly mention some of the excluded results.

What's on the Menu Today

- An Efficient algorithm for $(3 + o(1))$-EF allocations
- $(3 + o(1))$-NSW allocation.
- $(2 + o(1))$ - EF allocation.
- Briefly mention some of the excluded results.

If time permits...

What's on the Menu Today

- An Efficient algorithm for $(3 + o(1))$-*EF* allocations
- $(3 + o(1))$-*NSW* allocation.
- $(2 + o(1))$ - *EF* allocation.
- Briefly mention some of the excluded results.

If time permits...

What's on the Menu Today

- An Efficient algorithm for $(3 + o(1))$-EF allocations
- $(3 + o(1))$-NSW allocation.
- $(2 + o(1))$ - EF allocation.
- Briefly mention some of the excluded results.

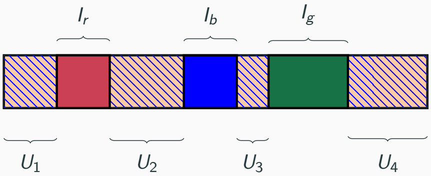
If time permits...

What's on the Menu Today

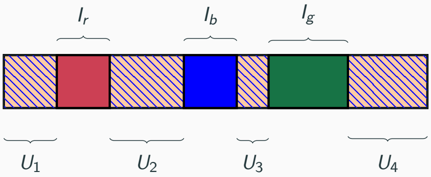
- An Efficient algorithm for $(3 + o(1))$-*EF* allocations
- $(3 + o(1))$-*NSW* allocation.
- $(2 + o(1))$ - *EF* allocation.
- Briefly mention some of the excluded results.

Computing $(3 + o(1))$-*approx EF* **allocations**

An example allocation : Some useful terms and notation



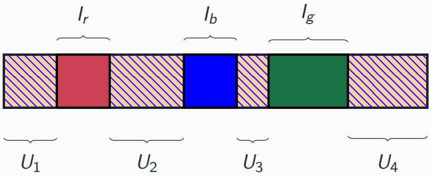
An example allocation : Some useful terms and notation



Unallocated intervals

$\mathcal{U} := \{U_1, U_2, U_3, U_4\}$

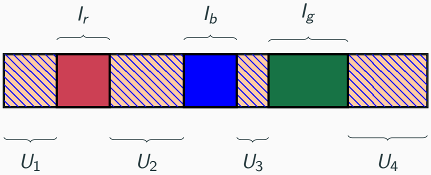
An example allocation : Some useful terms and notation



Allocated Intervals

$\mathcal{A} := \{I_r, I_b, I_g\}$

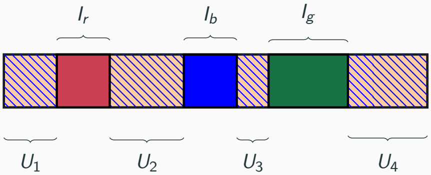
An example allocation : Some useful terms and notation



Intervals

$\mathcal{I} := \mathcal{A} \cup \mathcal{U}$

An example allocation : Some useful terms and notation

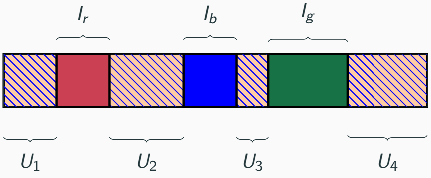


δ-additive envy

Agent $x \in \{r, g, b\}$ δ-additive envies interval $X \in \mathcal{I}$ if

$$v_x(I_x) < v_x(X) - \delta$$

Warmup Questions

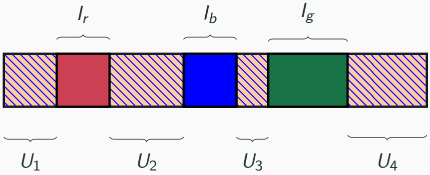


Questions

Suppose $|\mathcal{A}| = n$.

- An upper bound on $|\mathcal{U}|$?
- An upper bound on $|\mathcal{I}|$?

Warmup Questions

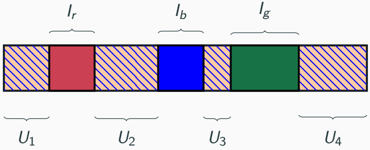


Questions

Suppose $|\mathcal{A}| = n$.

- An upper bound on $|\mathcal{U}|$? $n + 1$
- An upper bound on $|\mathcal{I}|$? $2n + 1$

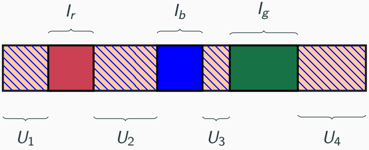
A nice example allocation: Are we done?



Suppose for a $\delta(n) > 0$ which we choose later

1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

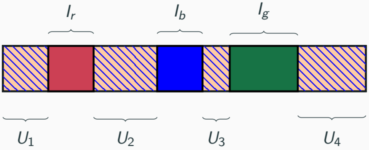
A nice example allocation: Are we done?



Suppose for a $\delta(n) > 0$ which we choose later

1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

A nice example allocation: Are we done?



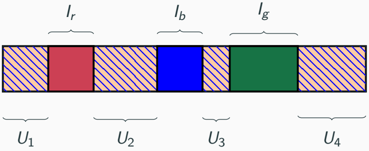
Suppose for a $\delta(n) > 0$ which we choose later

1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

Question

Can some agent r improve its value by more than δ if it swaps I_r with some $U_i \in \mathcal{U}$?

A nice example allocation: Are we done?



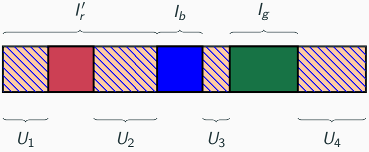
Suppose for a $\delta(n) > 0$ which we choose later

1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

Answer

No. By definition of δ-additive envy and assumption 2.

A nice example allocation: Are we done?



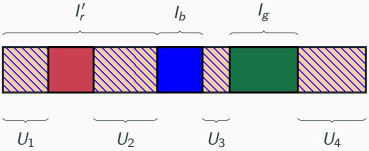
Suppose

1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

Another Question

Suppose agent r's allocation is expanded from I_r to I'_r (see figure). Upto what additive/multiplicative factors can any other agent b, be envy-free of r.

A nice example allocation: Are we done?



Suppose

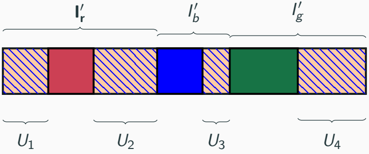
1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

Answer

Clearly $I_r' := U_1 \sqcup I_r \sqcup U_2$. Thus for any other agent s, using the assumptions and additivity of valuations,

$$3v_s(I_s) + 3\delta \geq v_s(I_r')$$

A nice example allocation: Are we done?



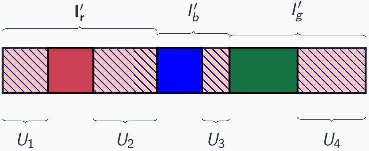
Suppose

1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

Another Question

And if all unassigned intervals are arbitrarily assigned in a similar manner?

A nice example allocation: Are we done?



Suppose

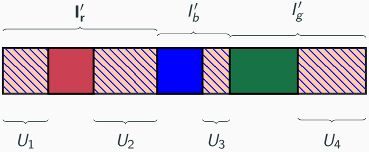
1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

Answer

Yes. For all agents r and s,

$$\begin{aligned} 3v_s(I'_s) + \delta &\geq 3v_s(I_s) + \delta \\ &\geq v_s(I'_r) \end{aligned}$$

A nice example allocation: Are we done?



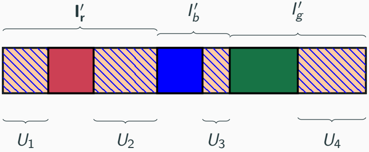
Suppose

1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

Almost

There is still that extra additive term of δ. We need more clues

A nice example allocation: Are we done?



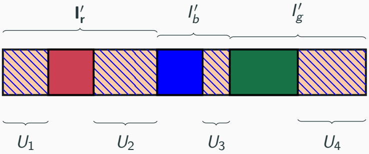
Suppose

1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

Another Question

What is the minimum value of allocation an arbitrary agent s gets?

A nice example allocation: Are we done?



Suppose

1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

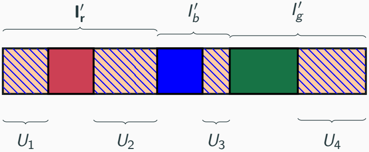
Answer

Can you see that the below is implied by assumption 1 and 2?

$$\forall X \in \mathcal{I},\ v_s(I_s) + \delta \geq v_s(X)$$

Let's sum up this inequality over each $X \in \mathcal{I}$.

A nice example allocation: Are we done?



Suppose

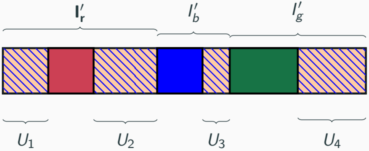
1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

Answer(Contd)

Here's the result

$$(2n+1)v_s(I_s) \geq 1 - 2n\delta$$

A nice example allocation: Are we done?



Suppose

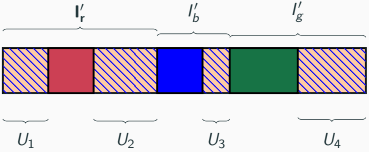
1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

Answer(Contd)

Using $v_s(I_s') > v_s(I_s)$

$$(2n+1)v_s(I_s') \geq 1 - 2n\delta$$

A nice example allocation: Are we done?



Suppose

1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

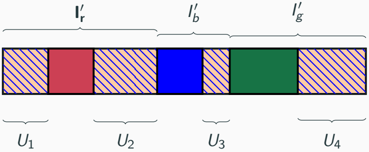
Answer(Contd): Allocations have min value

Simplify the above with $\delta = \Omega(1/n^2)$ to get

$$3n\delta v_s(I'_s) > \delta$$

for sufficiently large n depending on the constant.

A nice example allocation: Are we done?



Suppose

1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

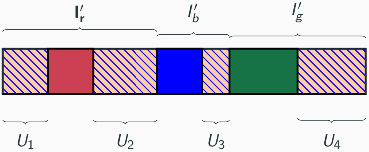
Answer(Contd) : Multiplicative Approx

Now we can turn the additive δ in $3v_s(I'_s) + \delta \geq v_s(I'_r)$ into a multiplicative form.

$$(3 + n\delta)v_s(I'_s) \geq v_s(I'_r)$$

Thus δ must be chosen as $\Omega(1/n^2)$

Now we are done ... almost



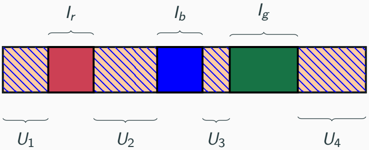
Suppose

1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

Procedure NICE-ALLOC

By assigning unallocated intervals to adjacent agents by arbitrary choice in a nice allocation, we get a $(3 + o(1))$-approximate EF allocation.

About those assumptions...



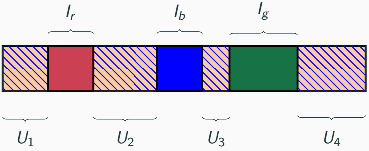
Suppose

1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

Question

How trivial is it to produce an allocation satisfying assumptions 1 and/or 2?

About those assumptions...



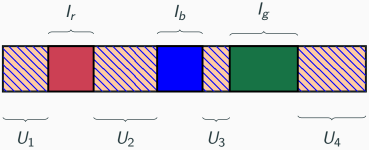
Suppose

1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

Question

How about assumption 1 alone?

About those assumptions...



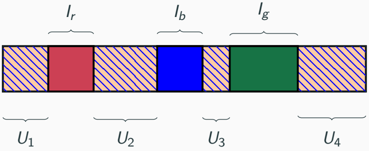
Suppose

1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

Answer

Easy : Give all agents the empty interval

About those assumptions...



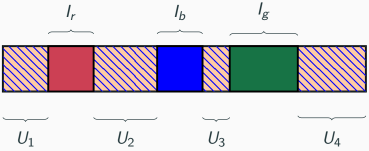
Suppose

1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

Question

And Assumption 2?

About those assumptions...



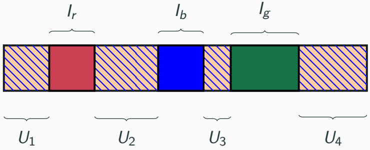
Suppose

1. No agent δ-additive envies any other agent's allocation.
2. No agent δ-additive envies any $X \in \mathcal{U}$.

Answer

Not so easy...

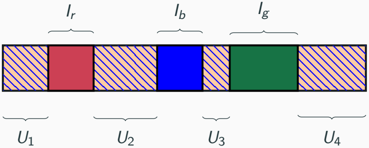
When Only Assumption 1 Holds



Suppose

1. No agent δ-additive envies any other agent's allocation.

When Only Assumption 1 Holds



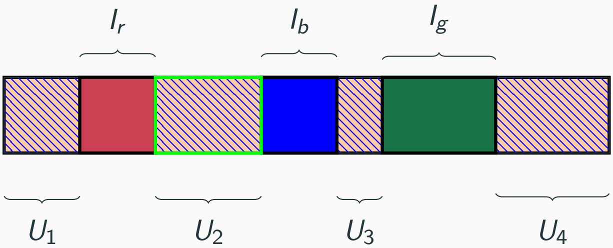
Suppose

1. No agent δ-additive envies any other agent's allocation.

Meaning?

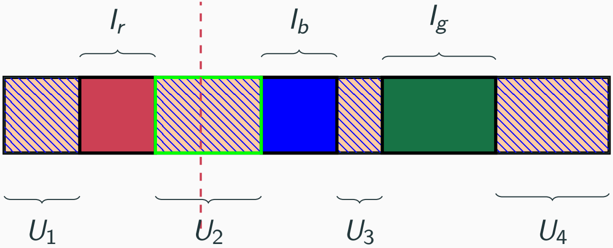
There is some $U \in \mathcal{U}$ say U_2, and some agents (say r and b), for whom U is atleast δ units more valuable than their own.

Allocating an unallocated interval U_2



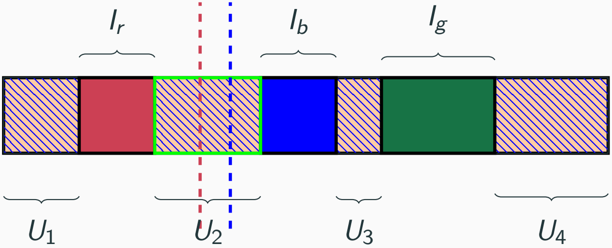
U_2 is δ-additively envied by r and b

Allocating an unallocated interval U_2



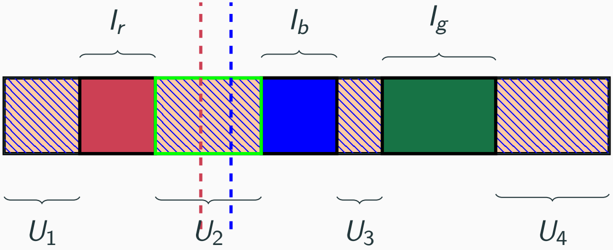
Red's claim is revealed

Allocating an unallocated interval U_2



Blue's claim is also revealed

Allocating an unallocated interval U_2

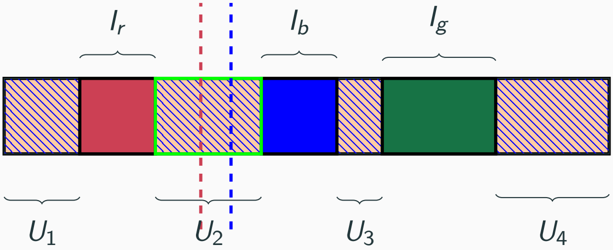


Blue's claim is also revealed

Question

Who's claim should we choose to allow

Allocating an unallocated interval U_2



Blue's claim is also revealed

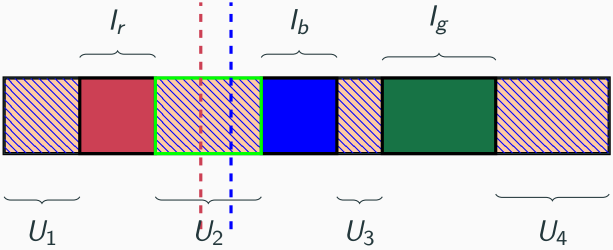
Question

Who's claim should we choose to allow

Consideration

The new allocation must respect δ-additive envy freeness of agents.

Allocating an unallocated interval U_2

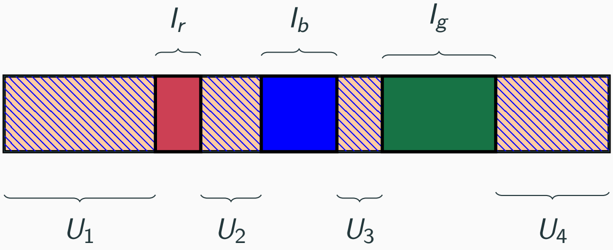


Blue's claim is also revealed

Answer

We honour red's claim. Discuss!

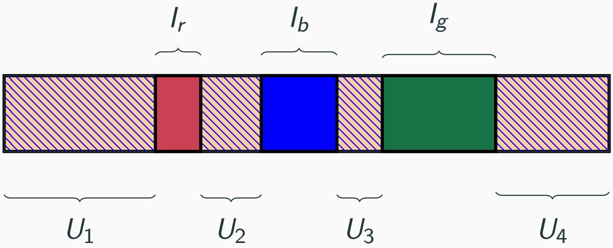
Allocating an unallocated interval U_2



r gets its new interval.

Strictly δ units more valuable than before according to v_r

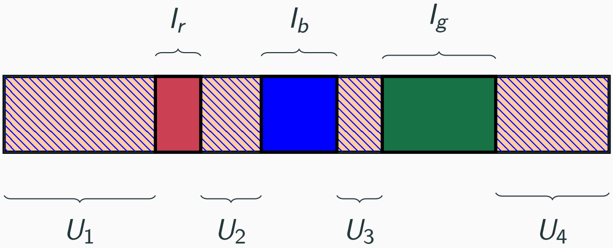
Allocating an unallocated interval U_2



Procedure SATURATE-ALLOC

If an unassigned interval is δ-additive envied by one or more agents, allocate the least envious agent, the exact amount to satisfy its δ-additive envy.

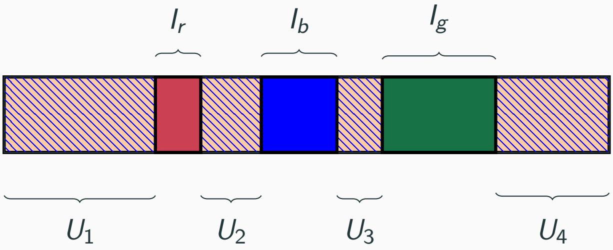
Allocating an unallocated interval U_2



Question

How many times can SATURATE-ALLOC be run starting from an empty allocation?

Allocating an unallocated interval U_2



Answer

$O(n^3)$ times. Why?

Putting it all together: Procedure ALG

- Start with empty allocation.

What's Applicable

- Agents are δ-additive envy free of each other.

- If there is an unallocated interval and an agent who δ-additive envies it, then, run SATURATE-ALLOC
- After $O(1/n^3)$ steps, there are no more unallocated intervals to assign.
- Use procedure NICE-ALLOC.

Putting it all together: Procedure ALG

- Start with empty allocation.
- If there is an unallocated interval and an agent who δ-additive envies it, then, run SATURATE-ALLOC

What's Applicable

- Agents remain δ-additive envy free of each other.

- After $O(1/n^3)$ steps, there are no more unallocated intervals to assign.
- Use procedure NICE-ALLOC.

Putting it all together: Procedure ALG

- Start with empty allocation.
- If there is an unallocated interval and an agent who δ-additive envies it, then, run SATURATE-ALLOC
- After $O(1/n^3)$ steps, there are no more unallocated intervals to assign.

What's Applicable

- Agents remain δ-additive envy free of each other.
- Agents are now δ-additive envy free of all unallocated intervals

- Use procedure NICE-ALLOC.

Putting it all together: Procedure ALG

- Start with empty allocation.
- If there is an unallocated interval and an agent who δ-additive envies it, then, run SATURATE-ALLOC
- After $O(1/n^3)$ steps, there are no more unallocated intervals to assign.
- Use procedure NICE-ALLOC.

What's Applicable

- All agents are $(3 + o(1))$-envy free
- The algorithm has terminated in $O(n^3)$ steps.

Computing $(3 + o(1))$-*approx NSW* **allocations**

$(3 + o(1))$-NSW allocation for free

Claim

The allocation computed by procedure ALG is a $(3 + o(1))$-NSW allocation

Recall Nash Social Welfare

For an allocation A of the cake $[0, 1]$ to n agents, with agent a getting interval N_a,

$$NSW(A) := (\Pi_{a \in Agents} v_a(N_a))^{1/n}$$

$(3+o(1))$-NSW allocation for free

Claim

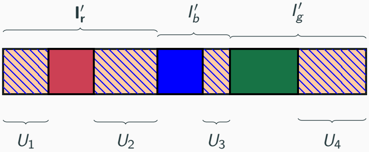
The allocation computed by procedure ALG is a $(3+o(1))$-NSW allocation

Recall Nash Social Welfare

For an allocation A of the cake $[0,1]$ to n agents, with agent a getting interval N_a,

$$NSW(A) := (\Pi_{a \in Agents} v_a(N_a))^{1/n}$$

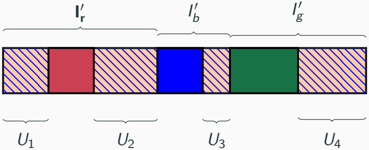
Proof Sketch : Take two allocations



A nice allocation A computed by ALG before NICE-ALLOC is applied

An Optimal NSW allocation A^* that ALG doesn't know of

Proof Sketch : Take two allocations



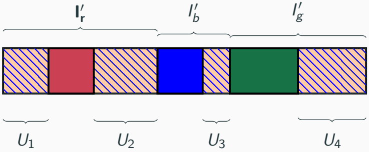
A nice allocation A computed by ALG before NICE-ALLOC is applied

An Optimal NSW allocation A^* that ALG doesn't know of

Question

How would you estimate $NSW(A^*)$ from A?

Proof Sketch : Take two allocations



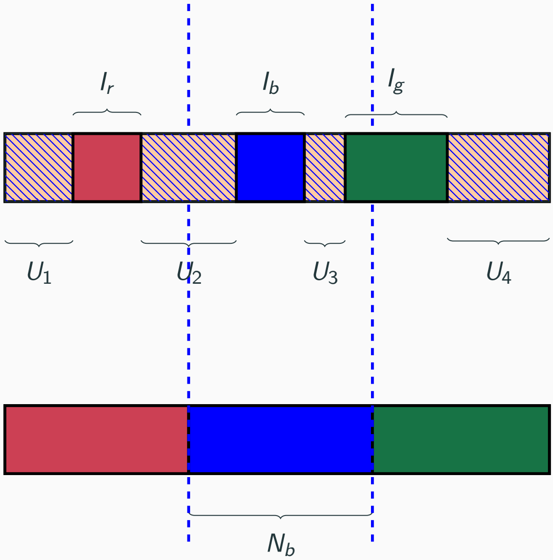
A nice allocation A computed by ALG before NICE-ALLOC is applied

An Optimal NSW allocation A^* that ALG doesn't know of

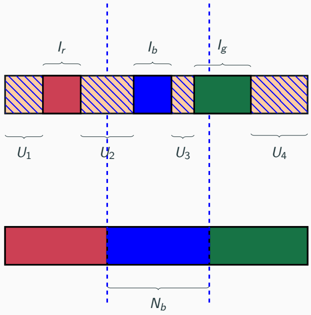
Hint

Both allocations are being done on the same cake.

Proof Sketch : Approximate one with the other



Proof Sketch : Approximate one with the other



- N_b is covered by U_2, I_b, U_3, and I_g.
- We already have for any X:

$$v_b(I_b) \geq v_b(X) - \delta$$

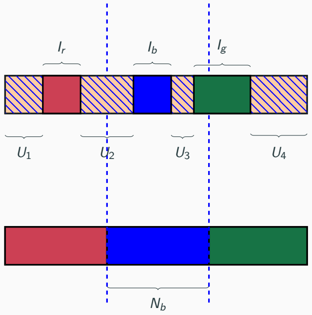
- Recall that allocations have minimum value:

$$3n\delta v_b(I_b) > \delta$$

- Thus taken over all covering intervals

$$v_b(I_b) \geq 3(1 + 3\delta)v_b(X)$$

Proof Sketch : Approximate one with the other



- N_b is covered by U_2, I_b, U_3, and I_g.
- We already have for any X:

$$v_b(I_b) \geq v_b(X) - \delta$$

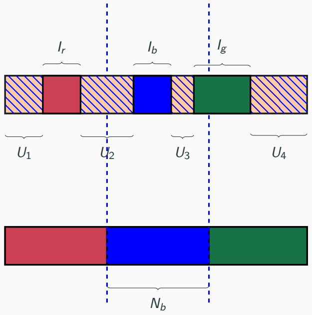
- Recall that allocations have minimum value:

$$3n\delta v_b(I_b) > \delta$$

- Thus taken over all covering intervals

$$v_b(I_b) \geq 3(1 + 3\delta)v_b(X)$$

Proof Sketch : Approximate one with the other



- N_b is covered by U_2, I_b, U_3, and I_g.
- We already have for any X:

$$v_b(I_b) \geq v_b(X) - \delta$$

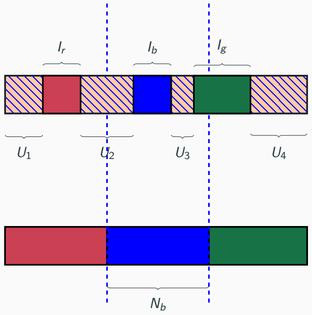
- Recall that allocations have minimum value:

$$3n\delta v_b(I_b) > \delta$$

- Thus taken over all covering intervals

$$v_b(I_b) \geq 3(1 + 3\delta)v_b(X)$$

Proof Sketch : Approximate one with the other



- N_b is covered by U_2, I_b, U_3, and I_g.
- We already have for any X:

$$v_b(I_b) \geq v_b(X) - \delta$$

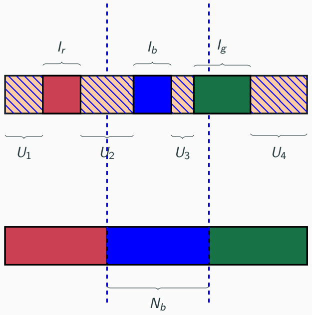
- Recall that allocations have minimum value:

$$3n\delta v_b(I_b) > \delta$$

- Thus taken over all covering intervals

$$v_b(I_b) \geq 3(1 + 3\delta)v_b(X)$$

Proof Sketch : Approximate one with the other



- N_b is covered by U_2, I_b, U_3, and I_g.
- We already have for any X:

$$v_b(I_b) \geq v_b(X) - \delta$$

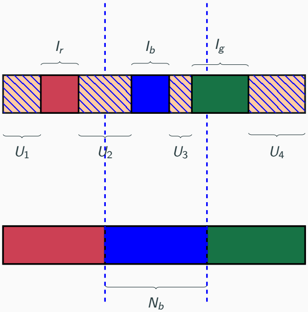
- Recall that allocations have minimum value:

$$3n\delta v_b(I_b) > \delta$$

- Thus taken over all covering intervals

$$v_b(I_b) \geq 3(1 + 3\delta) v_b(X)$$

Proof Sketch : Approximate one with the other



Observation

If the A^* allocation for agent r, N_r, spans k_r A-allocated intervals,

$$k_a v(I'_a)(1 + 3\delta) \geq v(N_a)$$

Proof Sketch : A bit of tedious math

Remember this

$$k_r v_r(I'_r)(1 + 3\delta) \geq v(N_r)$$

$$NSW(A^*) \leq (1 + 3\delta) \left(\Pi_{r \in Agents} v_r(I'_r)\right)^{1/n} \left(\Pi_{r \in Agents} k_r\right)^{1/n}$$

Proof Sketch : A bit of tedious math

Remember this

$$k_r v_r(I'_r)(1+3\delta) \geq v(N_r)$$

$$NSW(A^*) \leq (1+3\delta)\left(\Pi_{r \in Agents} k_r\right)^{1/n} \left(\Pi_{r \in Agents} v_r(I'_r)\right)^{1/n}$$
$$\leq (1+3\delta)\left(\Pi_{r \in Agents} k_r\right)^{1/n} NSW(A)$$

Proof Sketch : A bit of tedious math

Remember this

$$k_r v_r(I'_r)(1+3\delta) \geq v(N_r)$$

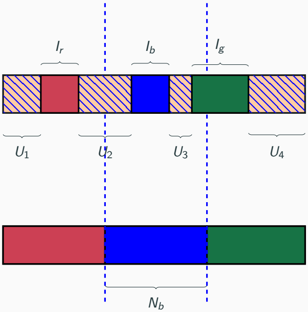
$$NSW(A^*) \leq (1+3\delta)\left(\Pi_{r\in Agents} k_r\right)^{1/n} NSW(A)$$
$$\leq (1+3\delta)(3+o(1))NSW(A)$$
$$\text{if } \left(\Pi_{r\in Agents} k_r\right)^{1/n} = 3+o(1) \text{ and } \delta = o(1)$$

Bounding the Geometric Mean of k_r's

Observation

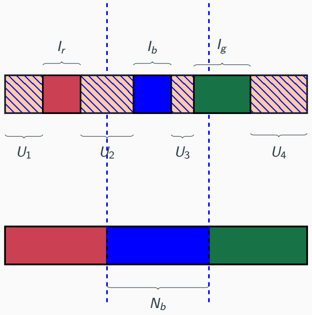
After applying AM-GM inequality, it suffices to bound the AM of $k_r's$

Bounding the Arithmetic Mean of k_r's



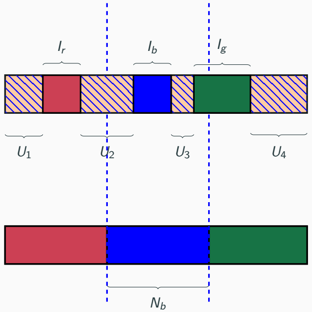
- At most $2n + 1$ intervals in A.
- At most $n - 1$ intervals appear in the covers of two adjacent NSW intervals in A^*.
- Conclusion : $\Sigma_{r \in Agents} k_r < 3n + 1$
- Conclusion $AM(k_r's) = 3 + o(1)$.

Bounding the Arithmetic Mean of k_r's



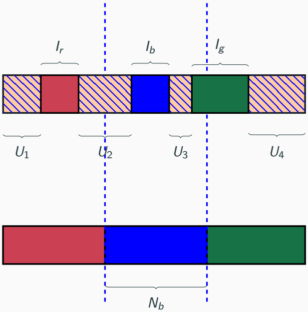
- At most $2n+1$ intervals in A.
- At most $n-1$ intervals appear in the covers of two adjacent NSW intervals in A^*.
- Conclusion : $\Sigma_{r \in Agents} k_r < 3n+1$
- Conclusion $AM(k_r' s) = 3 + o(1)$.

Bounding the Arithmetic Mean of k_r's



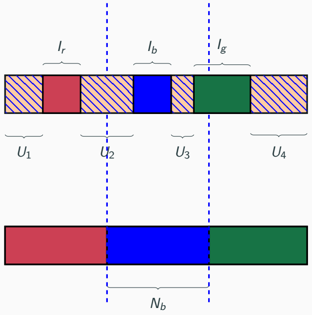
- At most $2n+1$ intervals in A.
- At most $n-1$ intervals appear in the covers of two adjacent NSW intervals in A^*.
- Conclusion : $\Sigma_{r \in Agents} k_r < 3n+1$
- Conclusion $AM(k_r' s) = 3 + o(1)$.

Bounding the Arithmetic Mean of k_r's



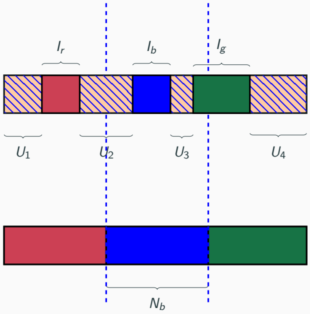
- At most $2n + 1$ intervals in A.
- At most $n - 1$ intervals appear in the covers of two adjacent NSW intervals in A^*.
- Conclusion : $\Sigma_{r \in Agents} k_r < 3n + 1$
- Conclusion $AM(k_r's) = 3 + o(1)$.

Bounding the Arithmetic Mean of k_r's



- At most $2n + 1$ intervals in A.
- At most $n - 1$ intervals appear in the covers of two adjacent NSW intervals in A^*.
- Conclusion : $\Sigma_{r \in Agents} k_r < 3n + 1$
- Conclusion $AM(k_r's) = 3 + o(1)$.

Conclusion : Free $(3 + o(1))$-Approx NSW



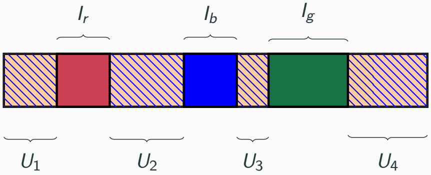
Conclusion

$NSW(A^*) \leq (3 + o(1))NSW(A)$

Computing $(2 + o(1)) - \mathit{approx}\ \mathit{EF}$ **allocations**

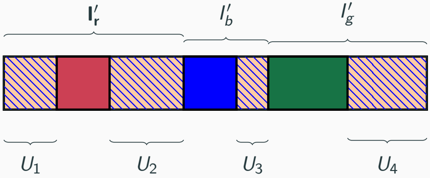
Look back at $(3 + o(1))$-EF

First we construct a nice δ-additive envy free allocation like below.



Look back at $(3 + o(1))$-EF

Then we use procedure NICE-ALLOC to assign the unallocated intervals to neighbours

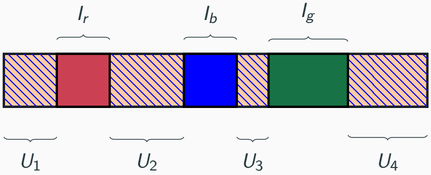


The source of our 3-factor approx

Let's rollback a bit to before we apply NICE-ALLOC.

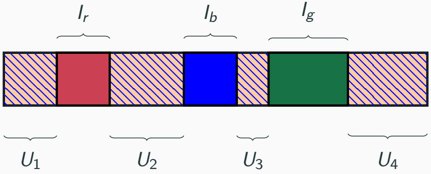
The source of our 3-factor approx

Let's rollback a bit to before we apply NICE-ALLOC.



The source of our 3-factor approx

Let's rollback a bit to before we apply NICE-ALLOC.



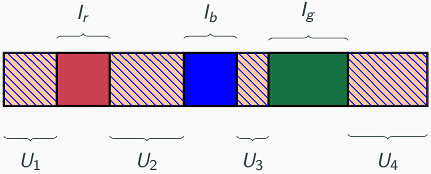
Recall : There can be $n + 1$ unallocated intervals

The source of our woes

At least one agent can get 2 unallocated intervals in NICE-ALLOC

The source of our 3-factor approx

Let's rollback a bit to before we apply NICE-ALLOC.

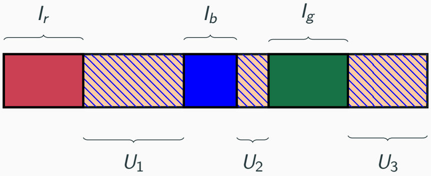


Question

Can we ensure that there are only n unallocated intervals in SATURATE-ALLOC

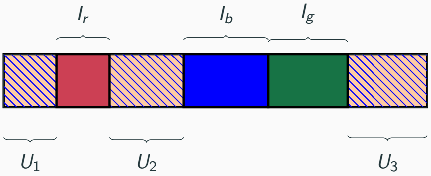
When do we get $\leq n$ unallocated blocks

When do we get $\leq n$ unallocated blocks



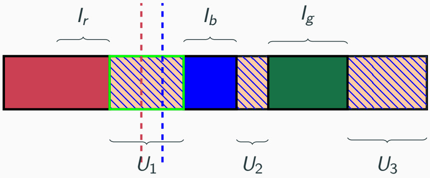
Case 1 : One or more allocations are along the edge

When do we get $\leq n$ unallocated blocks



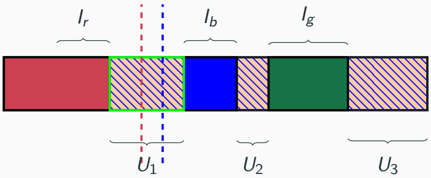
Case 2 : There are adjacent allocations

Adapting SATURATE-ALLOC



Case 1 : How should the green unallocated interval be allocated

Adapting SATURATE-ALLOC

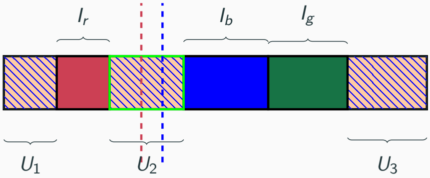


Case 1 : How should the green unallocated interval be allocated

Solution

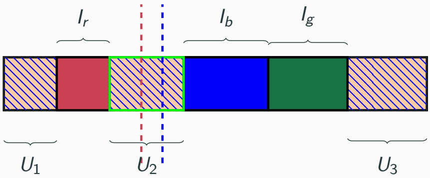
Cut from the right of the green interval

Adapting SATURATE-ALLOC



Case 2 : How should the green unallocated interval be allocated

Adapting SATURATE-ALLOC



Case 2 : How should the green unallocated interval be allocated

Solution

Cut from the right of the green interval

Conclusion

Conclusion

We can use ALG to obtain a $(2 + o(1))$-approx EF allocation if: whenever allocating a cut of an interval from the left produces $(n + 1)$ unallocated intervals, we allocate a cut from the right.

Things left unsaid

Things I left out

- Even without ALG, if you get an α-
- It is hard to do much better than constant factor approximation of envy freeness for contiguous cake Division. No PTAS expected.
- The results on NSW can be generalised to other kinds of central measures.
- Exact max NSW allocation is hard to compute.

Things I left out

- Even without ALG, if you get an α-
- It is hard to do much better than constant factor approximation of envy freeness for contiguous cake Division. No PTAS expected.
- The results on NSW can be generalised to other kinds of central measures.
- Exact max NSW allocation is hard to compute.

Things I left out

- Even without ALG, if you get an α-
- It is hard to do much better than constant factor approximation of envy freeness for contiguous cake Division. No PTAS expected.
- The results on NSW can be generalised to other kinds of central measures.
- Exact max NSW allocation is hard to compute.

Things I left out

- Even without ALG, if you get an α-
- It is hard to do much better than constant factor approximation of envy freeness for contiguous cake Division. No PTAS expected.
- The results on NSW can be generalised to other kinds of central measures.
- Exact max NSW allocation is hard to compute.

Things I left out

- Even without ALG, if you get an α-
- It is hard to do much better than constant factor approximation of envy freeness for contiguous cake Division. No PTAS expected.
- The results on NSW can be generalised to other kinds of central measures.
- Exact max NSW allocation is hard to compute.

Thank You