How to Fairly Allocate Goods to Players: Mathematical Model and Algorithms

Trung Thanh Nguyen

ORLab, Phenikaa University Viasm

December 5, 2023

Outline

Combinatorial Optimization Problem (COP)

Basic definition

$$\max/\min \quad f(x) \tag{1}$$
$$\text{subject to} \quad x \in \mathcal{S}. \tag{2}$$

where S is a set of finite feasible solutions.

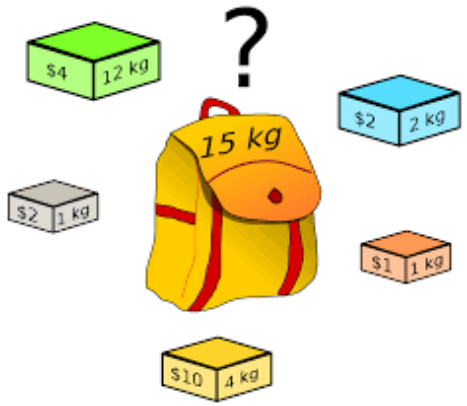
Facts:

- The size of S is often exponential large
- In general, COP is NP-hard to solve exactly

Example of COPs

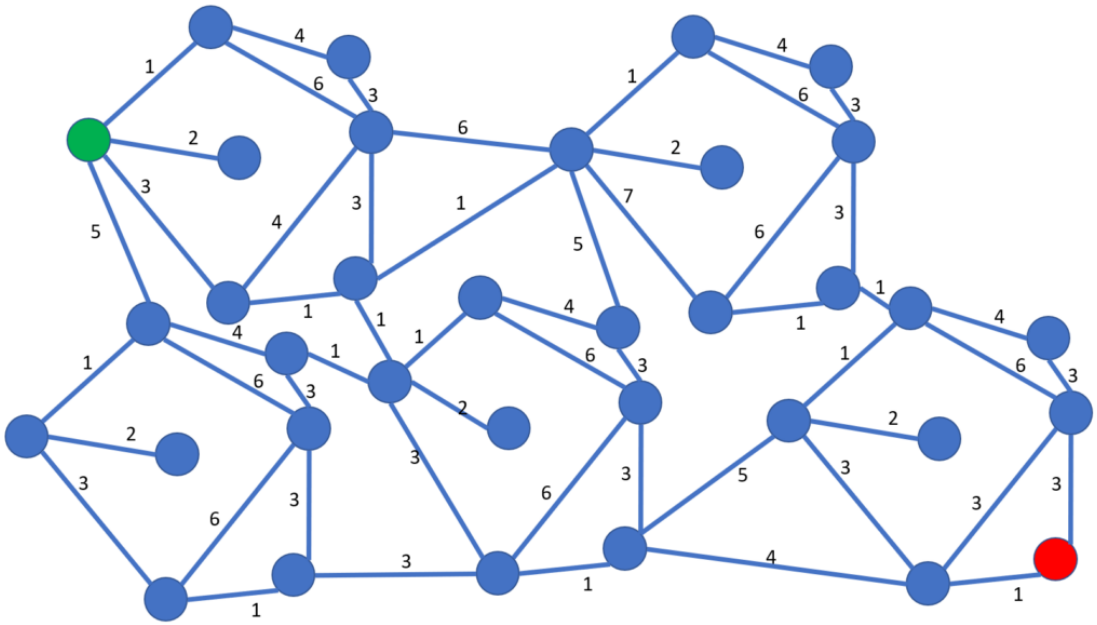
- Partition

- Knapsack



Example of COPs

- Shortest path, TSP



Resource Allocation Problem

Given:
- A set of goods (or objects) $[m] = \{1, \ldots, m\}$
- A set of agents $[n] = \{1, \ldots, n\}$
- Each agent j has a utility function $u_i : 2^{[m]} \to \mathbb{R}_+$
- Goods are indivisible and non-shareable

Resource Allocation Problem

Given:
- A set of goods (or objects) $[m] = \{1, \ldots, m\}$
- A set of agents $[n] = \{1, \ldots, n\}$
- Each agent j has a utility function $u_i : 2^{[m]} \to \mathbb{R}_+$
- Goods are indivisible and non-shareable

Allocation:
- $\pi = (\pi_1, \ldots, \pi_n)$: an ordered partition of $[m]$ into n subsets

Require:
- An allocation that is considered to be "fair" to every agent.

Resource Allocation Problem

Given:
- A set of goods (or objects) $[m] = \{1, \ldots, m\}$
- A set of agents $[n] = \{1, \ldots, n\}$
- Each agent j has a utility function $u_i : 2^{[m]} \to \mathbb{R}_+$
- Goods are indivisible and non-shareable

Allocation:
- $\pi = (\pi_1, \ldots, \pi_n)$: an ordered partition of $[m]$ into n subsets

Require:
- An allocation that is considered to be "fair" to every agent.

Applications: assign bandwidth to users in network, courses to students, jobs to machines, etc.
(`https://en.wikipedia.org/wiki/Fair_division_experiments`)

Resource Allocation Problem

Given: - A set of goods (or objects) $[m] = \{1, \ldots, m\}$
- A set of agents $[n] = \{1, \ldots, n\}$
- Each agent j has a utility function $u_i : 2^{[m]} \to \mathbb{R}_+$
- Goods are indivisible and non-shareable

Allocation: - $\pi = (\pi_1, \ldots, \pi_n)$: an ordered partition of $[m]$ into n subsets

Require: - An allocation that is considered to be "fair" to every agent.

Applications: assign bandwidth to users in network, courses to students, jobs to machines, etc.
(`https://en.wikipedia.org/wiki/Fair_division_experiments`)

The concern of economists: - What is a "fair" allocation?

The concern of mathematicians: - How hard is it to find a "fair" allocation?

Concepts of Fairness

Using Collective Utility Function (CUF)

Let $p = (u_1(\pi_1), \ldots, u_n(\pi_n))$ be the vector of agents utilities

Rawls function: $\min_i p_i$ - the utility of the worst-off agents

Concepts of Fairness

Using Collective Utility Function (CUF)

Let $p = (u_1(\pi_1), \ldots, u_n(\pi_n))$ be the vector of agents utilities

Rawls function: $\min_i p_i$ - the utility of the worst-off agents

Lexi function: the biggest p^* over all vectors p, based on lexicographic order

Concepts of Fairness

Using Collective Utility Function (CUF)

Let $p = (u_1(\pi_1), \ldots, u_n(\pi_n))$ be the vector of agents utilities

Rawls function: $\min_i p_i$ - the utility of the worst-off agents

Lexi function: the biggest p^* over all vectors p, based on lexicographic order

Nash function: $\prod_i p_i$ (or $(\prod_i p_i)^{1/n}$) - the product of agents utilities

Concepts of Fairness

Defining fair share of agents

An allocation is fair if the utility of every agent is at least her fair share

Concepts of Fairness

Defining fair share of agents

An allocation is fair if the utility of every agent is at least her fair share

maximin share (MMS): $\text{MMS}_i = \max_\pi \min_j u_i(\pi_j)$

Concepts of Fairness

Defining fair share of agents

An allocation is fair if the utility of every agent is at least her fair share

maximin share (MMS): $\text{MMS}_i = \max_\pi \min_j u_i(\pi_j)$

proportional share (PROP): $\text{PS}_i = u_i([m])/n$

Concepts of Fairness

Defining fair share of agents

An allocation is fair if the utility of every agent is at least her fair share

maximin share (MMS): $\text{MMS}_i = \max_\pi \min_j u_i(\pi_j)$

proportional share (PROP): $\text{PS}_i = u_i([m])/n$

envy-freeness (EF): $u_i(\pi_i) \geq u_i(\pi_j)$ for all i, j.

Concepts of Fairness

Defining fair share of agents

An allocation is fair if the utility of every agent is at least her fair share

maximin share (MMS): $\text{MMS}_i = \max_\pi \min_j u_i(\pi_j)$

proportional share (PROP): $\text{PS}_i = u_i([m])/n$

envy-freeness (EF): $u_i(\pi_i) \geq u_i(\pi_j)$ for all i, j.

MMS allocation $\leftarrow$ PROP allocation $\leftarrow$ EF allocation

Concepts of Fairness

Defining fair share of agents

An allocation is fair if the utility of every agent is at least her fair share

maximin share (MMS): $\text{MMS}_i = \max_\pi \min_j u_i(\pi_j)$

proportional share (PROP): $\text{PS}_i = u_i([m])/n$

envy-freeness (EF): $u_i(\pi_i) \geq u_i(\pi_j)$ for all i, j.

MMS allocation $\leftarrow$ PROP allocation $\leftarrow$ EF allocation

Bad news: Fair allocations do not always exist.

Relaxing concepts of fair share

An allocation is fair if the utility of every agent is at least her fair share

Concepts of Fairness

Defining fair share of agents

An allocation is fair if the utility of every agent is at least her fair share

maximin share (MMS): $\text{MMS}_i = \max_\pi \min_j u_i(\pi_j)$

proportional share (PROP): $\text{PS}_i = u_i([m])/n$

envy-freeness (EF): $u_i(\pi_i) \geq u_i(\pi_j)$ for all i, j.

MMS allocation $\leftarrow$ PROP allocation $\leftarrow$ EF allocation

Bad news: Fair allocations do not always exist.

Relaxing concepts of fair share

An allocation is fair if the utility of every agent is at least her fair share

1-out-of-$(n+1)$-MMS: $\max_{\pi=(\pi_1,\ldots,\pi_{n+1})} \min_{k=1}^{n+1} u_i(\pi_k)$

Concepts of Fairness

Defining fair share of agents

An allocation is fair if the utility of every agent is at least her fair share

maximin share (MMS): $\text{MMS}_i = \max_\pi \min_j u_i(\pi_j)$

proportional share (PROP): $\text{PS}_i = u_i([m])/n$

envy-freeness (EF): $u_i(\pi_i) \geq u_i(\pi_j)$ for all i, j.

MMS allocation $\leftarrow$ PROP allocation $\leftarrow$ EF allocation

Bad news: Fair allocations do not always exist.

Relaxing concepts of fair share

An allocation is fair if the utility of every agent is at least her fair share

1-out-of-$(n+1)$-MMS: $\max_{\pi=(\pi_1,\ldots,\pi_{n+1})} \min_{k=1}^{n+1} u_i(\pi_k)$

PS up to one item (PROP1): for every i, there exists an item j such that $u_i(\pi_i \cup \{j\}) \geq u_i([m])/n$

Concepts of Fairness

Defining fair share of agents

An allocation is fair if the utility of every agent is at least her fair share

maximin share (MMS): $\text{MMS}_i = \max_\pi \min_j u_i(\pi_j)$

proportional share (PROP): $\text{PS}_i = u_i([m])/n$

envy-freeness (EF): $u_i(\pi_i) \geq u_i(\pi_j)$ for all i, j.

MMS allocation $\leftarrow$ PROP allocation $\leftarrow$ EF allocation

Bad news: Fair allocations do not always exist.

Relaxing concepts of fair share

An allocation is fair if the utility of every agent is at least her fair share

1-out-of-$(n+1)$-MMS: $\max_{\pi=(\pi_1,\ldots,\pi_{n+1})} \min_{k=1}^{n+1} u_i(\pi_k)$

PS up to one item (PROP1): for every i, there exists an item j such that $u_i(\pi_i \cup \{j\}) \geq u_i([m])/n$

envy-freeness (EF1): for every i, $u_i(\pi_i) \geq u_i(\pi_j \setminus \{j\})$, $\exists j \in \pi_k$, $\forall k$.

Concepts of Fairness

Defining fair share of agents

An allocation is fair if the utility of every agent is at least her fair share

maximin share (MMS): $\text{MMS}_i = \max_\pi \min_j u_i(\pi_j)$

proportional share (PROP): $\text{PS}_i = u_i([m])/n$

envy-freeness (EF): $u_i(\pi_i) \geq u_i(\pi_j)$ for all i, j.

MMS allocation $\leftarrow$ PROP allocation $\leftarrow$ EF allocation

Bad news: Fair allocations do not always exist.

Relaxing concepts of fair share

An allocation is fair if the utility of every agent is at least her fair share

1-out-of-$(n+1)$-MMS: $\max_{\pi=(\pi_1,\ldots,\pi_{n+1})} \min_{k=1}^{n+1} u_i(\pi_k)$

PS up to one item (PROP1): for every i, there exists an item j such that $u_i(\pi_i \cup \{j\}) \geq u_i([m])/n$

envy-freeness (EF1): for every i, $u_i(\pi_i) \geq u_i(\pi_j \setminus \{j\})$, $\exists j \in \pi_k$, $\forall k$.

Good news: PROP1 allocations and EF1 allocations does always exist.

Computational Problems

Objectives based on CUF

Given a problem instance, find an allocation that maximizes either one of the functions: Rawls, Lexi, and Nash functions.

Computational Problems

Objectives based on CUF

Given a problem instance, find an allocation that maximizes either one of the functions: Rawls, Lexi, and Nash functions.

Objectives based on fair share

Given a problem instance, find an allocation that is maximin fair, proportional fair, or envy-free.

Sub-problems:

- Computing fair share for each agent.
- Deciding whether a fair allocation exists.

Types of Utility Functions

Hardness of fair allocation problems depends heavily on types of utility functions.

additive: $u_i(S) = \sum_{j \in S} u_i(j)$ for every $S \subseteq [m]$, $u_i(j) \equiv u_{ij}$

submodular: $u_i(S \cup V) + u_i(S \cap V) \leq u_i(S) + u_i(V)$, for every $S, V \subseteq [m]$

subadditive: $u_i(S \cup V) \leq u_i(S) + u_i(V)$, for every $S, V \subseteq [m]$

Mathematical models

Max-Min Allocation (MMA) as a Binary Linear Program

$$\text{maximize} \quad \min_{i=1}^{n} \left\{ \sum_{j=1}^{m} u_{ij} x_{ij} \right\} \tag{3}$$

$$\text{subject to} \quad \sum_{i=1}^{n} x_{ij} = 1, \qquad j \in [m], \tag{4}$$

$$x_{ij} \in \{0, 1\} \qquad i \in [n], j \in [m], \tag{5}$$

where $x_{ij} = 1$ if good j is assigned to agent i, and $x_{ij} = 0$ otherwise.

Mathematical models

Max-Min Allocation (MMA) as a Binary Linear Program

$$\text{maximize} \quad \min_{i=1}^{n} \left\{ \sum_{j=1}^{m} u_{ij} x_{ij} \right\} \tag{3}$$

$$\text{subject to} \quad \sum_{i=1}^{n} x_{ij} = 1, \qquad j \in [m], \tag{4}$$

$$x_{ij} \in \{0, 1\} \qquad i \in [n], j \in [m], \tag{5}$$

where $x_{ij} = 1$ if good j is assigned to agent i, and $x_{ij} = 0$ otherwise.

Partition problem - A special case of MMA

Given a set S of positive number, divide the set into two disjoint subsets so that the difference between the subsets is as small as possible.

$$\text{minimize}_{\emptyset \neq V \subset S} \left| \sum_{j \in V} j - \sum_{j \in S \setminus V} j \right|$$

Known Results

How hard to solve MMA

- MMA is at least as hard as Partition problem, which is NP-hard
- There is no polynomial time algorithm for exactly solving MMA
- The current best algorithm, based on Dynamic Programming, has exponential run-time in n.

Known Results

How hard to solve MMA

- MMA is at least as hard as Partition problem, which is NP-hard
- There is no polynomial time algorithm for exactly solving MMA
- The current best algorithm, based on Dynamic Programming, has exponential run-time in n.

Solve MMA approximately by polynomial time algorithms

- There is no $(\frac{1}{2} + \epsilon)$-approximation for MMA, for any $\epsilon > 0$ (Bezakova and Dani 2005)
- The current best algorithm has an approximation factor of $\frac{1}{c.\sqrt{n \ln n}}$ Saha and Srinivasan (2018)

Known Results

How hard to solve MMA

- MMA is at least as hard as Partition problem, which is NP-hard
- There is no polynomial time algorithm for exactly solving MMA
- The current best algorithm, based on Dynamic Programming, has exponential run-time in n.

Solve MMA approximately by polynomial time algorithms

- There is no $(\frac{1}{2} + \epsilon)$-approximation for MMA, for any $\epsilon > 0$ (Bezakova and Dani 2005)
- The current best algorithm has an approximation factor of $\frac{1}{c.\sqrt{n \ln n}}$ Saha and Srinivasan (2018)

Existing works have mainly focused on special cases where better algorithms can be designed.

Mathematical models

Nash Allocation (NA)

$$\text{maximize} \quad \prod_{i=1}^{n} \left\{ \sum_{j=1}^{m} u_{ij} x_{ij} \right\} \tag{6}$$

$$\text{subject to} \quad \sum\nolimits_{i=1}^{n} x_{ij} = 1, \qquad j \in [m], \tag{7}$$

$$x_{ij} =\in \{0,1\} \qquad i \in [n], j \in [m], \tag{8}$$

where $x_{ij} = 1$ if good j is assigned to agent i, and $x_{ij} = 0$ otherwise.

Mathematical models

Nash Allocation (NA)

$$\text{maximize} \quad \prod_{i=1}^{n} \left\{ \sum_{j=1}^{m} u_{ij} x_{ij} \right\} \tag{6}$$

$$\text{subject to} \quad \sum_{i=1}^{n} x_{ij} = 1, \qquad j \in [m], \tag{7}$$

$$x_{ij} = \in \{0, 1\} \qquad i \in [n], j \in [m], \tag{8}$$

where $x_{ij} = 1$ if good j is assigned to agent i, and $x_{ij} = 0$ otherwise.

A special case of NA

Given a set S of positive number, divide the set into two disjoint subsets to maximize the following objective

$$\sum_{j \in V} j \cdot \sum_{j \in S \setminus V} j$$

over all V.

Known Results

How hard to solve NA

- We was the first to study NA in 2013
- NA has a similar computational complexity as MMA

Known Results

How hard to solve NA

- We was the first to study NA in 2013
- NA has a similar computational complexity as MMA

Solve NA approximately by polynomial time algorithms

- There is $\frac{1}{m}$-approximation for NA (even for subadditive functions) (Nguyen and Rothe 2014)
- There is $\frac{1}{2.89}$-approximation for NA (Cole and Gkatzelis 2015)
- There is no 0.99-approximation for NA. (Lee 2017)
- The current best algorithm has an approximation factor of $\frac{1}{1.45} \approx 0.68$ (Barman et al. 2017)

Known Results

How hard to solve NA

- We was the first to study NA in 2013
- NA has a similar computational complexity as MMA

Solve NA approximately by polynomial time algorithms

- There is $\frac{1}{m}$-approximation for NA (even for subadditive functions) (Nguyen and Rothe 2014)
- There is $\frac{1}{2.89}$-approximation for NA (Cole and Gkatzelis 2015)
- There is no 0.99-approximation for NA. (Lee 2017)
- The current best algorithm has an approximation factor of $\frac{1}{1.45} \approx 0.68$ (Barman et al. 2017)

Open question: Narrowing the gap $[0.68; 0.99]$

Maximin Fair Allocation Problem (MFA)

Known results

- Computing MMS for each agent is an NP-hard problem
- An allocation that is maximin fair may not exist for $n \geq 3$. (Kurokawa et al. 2016)
- A $\frac{3}{4}$-maximin fair allocation always exist and can be efficiently computed. (Garg and Taki 2020)

Proportional Fair Allocation Problem (PFA)

Known results

- Computing PS is easy
- PROP allocations do not always exist, even for $n = 2$.
- Deciding if there is a PROP-allocation is an NP-hard problem.
- PROP1-allocations can be computed in strongly polynomial time (Barman, Siddharth; Krishnamurthy 2019)

Envy-Free Allocation Problem (EFA)

Known results

- EF-allocations do not always exist, even for $n = 2$.
- Deciding if there is an EF-allocation is an NP-hard problem.
- EF1-allocations can be computed by using the Round-Robin algorithm

Approximation Schemes for MMA and NA

The case when n is small

Design a polynomial time algorithm that return an allocation whose value is within a factor of $1 - \epsilon$ of the optimum.

Algorithm 1 Pseudo-polynomial-time algorithm

```
1: V_0 := {0};
2: for j := 1 to m do
3:     V_j := ∅;
4:     for each v ∈ V_{j-1} do
5:         V_j := V_j ∪ {v + s_ij · e_i | i = 1, ..., n};
6:     end for
7: end for
8: return Vector v ∈ V_m such that the product of its coordinates is maximal;
```

- $(e_1, \ldots, e_n)$ be the canonical basis of the vector space $\mathbb{R}^n$, where e_i denotes the vector with a 1 in the ith coordinate and 0's elsewhere
- $s_{ij} = u_{ij}$
- V_j the set of all possible assignments of the first j goods to agents, where each assignment is encoded by an n-dimensional vector v.

Approximation Schemes for MMA and NA

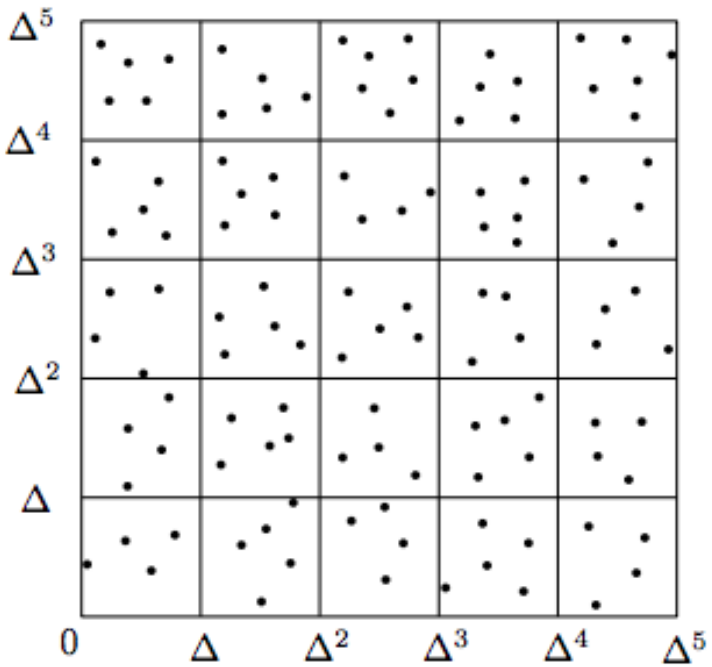
Agent	r_1	r_2	r_3	r_4	r_5
a_1	0	1	2	0	1
a_2	1	1	0	1	0

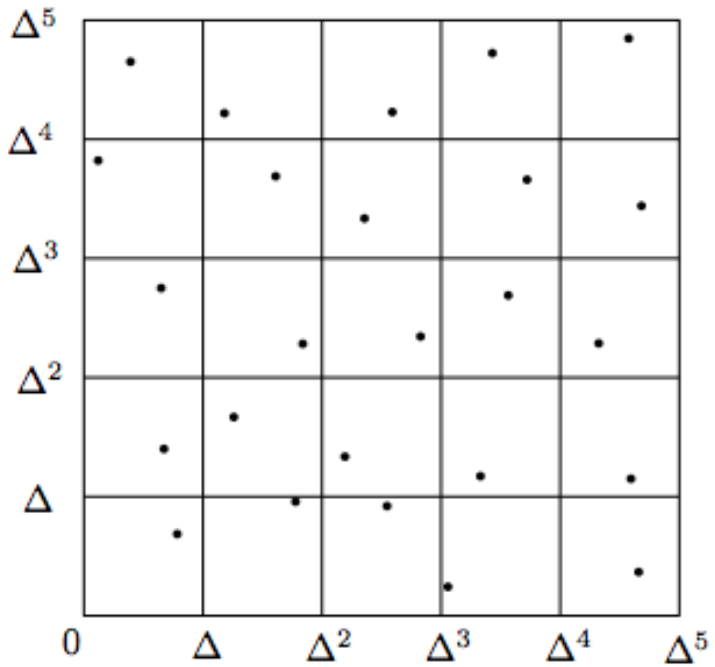
Step	V_j	Size of V_j
$j = 1$	(0, 0), (0, 1)	2
$j = 2$	(1, 0), (0, 1), (1, 1), (0, 2)	4
$j = 3$	(3, 0), (1, 0), (2, 1), (0, 1), (3, 1), (1, 1), (2, 2), (0, 2)	8
$j = 4$	(3, 0), (3, 1), (1, 0), (1, 1), (2, 1), (2, 2), (0, 1), (0, 2) (3, 1), (3, 2), (1, 1), (1, 2), (2, 2), (2, 3), (0, 2), (0, 3)	16
$j = 5$	(4, 0), (3, 0), (4, 1), (3, 1) (2, 0), (1, 0), (2, 1), (1, 1) (3, 1), (2, 1), (3, 2), (2, 2) (1, 1), (0, 1), (1, 2), (0, 2) (4, 1), (3, 1), (4, 2), (3, 2) (2, 1), (1, 1), (2, 2), (1, 2) (3, 2), (2, 2), (3, 3), (2, 3) (1, 2), (0, 2), (1, 3), (0, 3)	32

Approximation Schemes for MMA and NA

Running time of the pseudo-algorithm

$$O(mB^n), \quad \text{where } B = \max_i u_i([m])$$



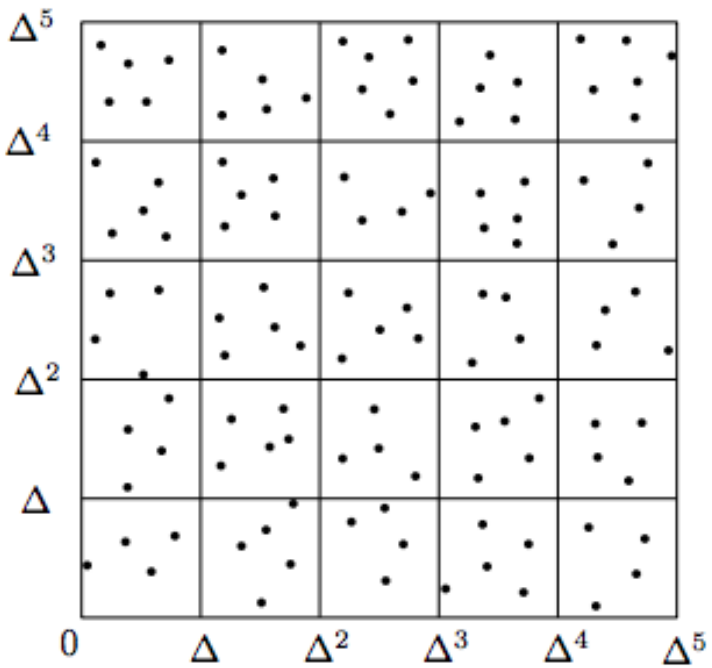


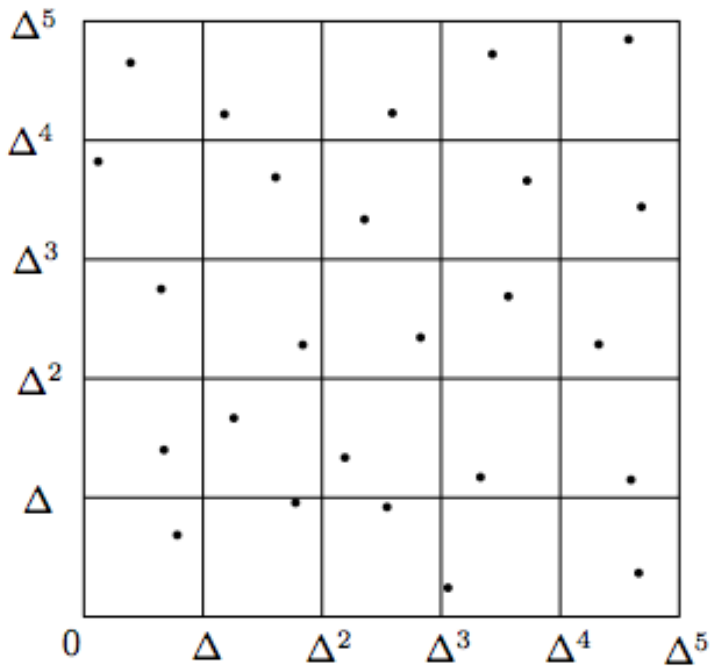
$$\Delta = 1 + \frac{\epsilon}{2m}$$

Approximation Schemes for MMA and NA

Running time of the modified pseudo-algorithm

$$O(mK^{2n}), \quad \text{where } K \leq \left(1 + \frac{2m}{\epsilon}\right) \cdot |I|$$





$$\Delta = 1 + \frac{\epsilon}{2m}$$

Open problems

- Designing a constant approximation algorithm for MMA.
- Narrowing the gap of $[0.68; 0.99]$ for NA.
- Improving $\frac{3}{4}$-maximin fair allocation

THANK YOU FOR YOUR ATTENTION!