Allocating Goods to Maximize Fairness

Deeparnab Chakrabarty  
U. of Pennsylvania

Julia Chuzhoy  
TTI-C

Sanjeev Khanna  
U. of Pennsylvania
Max Min Allocation

Input:

• Set $A$ of $m$ agents
• Set $I$ of $n$ items
• Utilities $u_{A,i}$ of agent $A$ for item $i$.

Output: assignment of items to agents.

• Utility of agent $A$: $\sum u_{A,i}$ for items $i$ assigned to agent $A$.

Goal: Maximize minimum utility of any agent.

Notation

$n$ - number of items
$m$ - number of agents
Example
Example

Solution value: 4
Max-Min Allocation

• Captures a natural notion of fairness in allocation of indivisible goods.
• Is related to the cake cutting theory.
• Approximation is still poorly understood.
• An interesting special case: Santa Claus problem.
The Santa Claus Problem
Santa Claus: Known Results

- Natural LP has $\Omega(m)$ integrality gap.
- [Bansal, Sviridenko ‘06]:
  - Introduced a new configuration LP
  - $O(\log \log m/\log\log\log m)$-approximation algorithm
- Non-constructive constant upper bounds on integrality gap of the LP [Feige ‘08], [Asadpour, Feige, Saberi ‘08].
- Constant approximation [Haeupler, Saha, Srinivasan ‘10]

Bad news: Configuration LP has $\Omega(\sqrt{m})$ integrality gap for Max-Min Allocation [Bansal, Sviridenko ‘06].
Known Results for Max Min Allocation

• (n-m+1)-approximation [Bezakova, Dani ‘05].
• $\tilde{O}(\sqrt{m})$-approximation via the configuration LP [Asadpour, Saberi ‘07].
• Configuration LP has $\Omega(\sqrt{m})$ integrality gap [Bansal, Sviridenko ‘06].
• Best current hardness of approximation factor: 2 [Bezakova, Dani ’05]
Our Results

• $\tilde{O}(n^\varepsilon)$-approximation algorithm in time $n^{O(1/\varepsilon)}$
  – Poly-logarithmic approximation in quasi-polynomial time.
  – $n^\varepsilon$-approximation in poly-time for any constant $\varepsilon$.

• We use an LP with $\Omega(\sqrt{m})$ integrality gap as a building block.
Independent Work

[Bateni, Charikar, Guruswami ‘09] obtained similar results for special cases of the problem:

• All utilities are in \{0, 1, M\}, where M=OPT.
  – All items have degree at most 2
  – Graph contains no cycles

• An \( \tilde{O}\left(n^\epsilon\right) \)-approximation in time \( n^{O(1/\epsilon)} \)
The $\tilde{O}(n^\epsilon)$-Approximation Algorithm

For simplicity, assume all utilities are in \{0,1,M\} where M=OPT.
Each agent A is assigned:
• One utility-M item or
• M utility-1 items
Each agent $A$ is assigned:

- One utility-$M$ item or
- $M$ utility-$1$ items

$\alpha$-approximate solution

$OPT=M$

utility 1

utility M

$M/\alpha$
All agents are either **heavy** or **light**.

Can assume w.l.o.g. we are given a canonical instance.
Step 1: Turn the Problem into a Flow Problem!
Main Idea

- Temporarily assign private items to agents

OPT = M

utility 1
utility M
Main Idea

- Temporarily assign private items to agents
  - Item can be private for at most one agent
  - If i is private for A then $u_{A,i} = M$
  - Every light agent gets a private item

![Diagram showing items and agents with utility assignments](image-url)
Main Idea

- **Temporarily assign private items to agents**
  - Item can be private for at most one agent
  - If i is private for A then $u_{A,i}=M$
  - Every light agent gets a private item
Re-Assignment of Items

• Use flow from source vertices towards terminals.
• An agent releases its private item iff it is satisfied by other items.
• Goal: find flow satisfying the terminals.
The Flow Network
The Flow Network

Heavy agent w. private item

Private item
The Flow Network

Heavy agent w. private item

Terminal

Light Agent

Source s and items in S

Sends 1 flow unit iff receives 1 flow unit

Must receive 1 flow unit

Sends 1 flow unit iff receives M flow units

Private item
The Flow Network

- **Light Agent**
  - Sends 1 flow unit iff receives M flow units

- **Private item**
  - Sends 1 flow unit iff receives 1 flow unit

- **Conservation of flow on items**

- **Source S and items in S**
  - Sends 1 flow unit
  - Must receive 1 flow unit

- **Terminal**
  - At most 1 flow unit leaves any vertex

Want to find **integral flow** satisfying these constraints...
Interpretation of Flow

Edge $e$ carries 1 flow unit

Lies in the symmetric difference of OPT and our assignment of private items

No flow sent through agent $A$

A is assigned its private item

Flow from item $i$ to agent $A$

Item $i$ is assigned to $A$
Interpretation of Flow

Edge $e$ carries 1 flow unit

Lies in the symmetric difference of OPT and our assignment of private items

• If OPT=M then such flow always exists!
The Flow Network

Heavy agent w. private item

- Private item
- Sends 1 flow unit iff receives 1 flow unit

Light Agent

- $\alpha$-relaxed flow
- Sends 1 flow unit iff receives $M/\alpha$ flow units

Terminal

- Must receive 1 flow unit

Source $s$ and items in $S$
Interpretation of Flow

- Edge e carries 1 flow unit
- Lies in the symmetric difference of OPT and our assignment of private items

- If OPT=M then such flow always exists!
- An $\alpha$-relaxed flow gives an $\alpha$-approximation!
What Does a Feasible Flow Look Like?

A collection of structures like this:
What Does a Feasible Flow Look Like?

A collection of trees like this:
Equivalent Problem Statement

Find a collection of such disjoint trees!

• A tree for each terminal
• Solution value = min degree of a light agent.
• If we only want $\tilde{O}(n^\epsilon)$-approximation, can assume that $h \leq 1/\epsilon$
Rest of the Algorithm

• LP and its rounding
• Use the LP-rounding as a sub-routine to get final solution.
LP-rounding

• Can write LP relaxation of flow constraints and try LP-rounding.
  – Easy to see that such an LP is too weak.

• We write a stronger LP.
• LP-variable for every \( h \)-tuple of light agents.
• LP-size: \( n^{O(h)} \)
• Integrality gap: \( \Omega(\sqrt{m}) \)
• LP-rounding gives poly(log \( n \))-approximate almost-feasible solutions!
Almost Feasible Solutions

**In-degree M**

- **Flow directly to terminals**
- **Flow to light agents**
Almost Feasible Solutions

An item may appear on one blue and one green path.

Flow directly to terminals
Flow to light agents

$M/\alpha$

In-degree M

An item may appear on one blue and one green path.
Rest of the Algorithm

• LP and its rounding
• Use the LP-rounding as a sub-routine to get final solution.
Rest of the Algorithm

• LP and its rounding

• Use the LP-rounding as a sub-routine to get final solution.
Getting around the Integrality Gap

Integrality gap of the LP is $\Omega(\sqrt{m})$

$\Rightarrow$ For some inputs to LP the gap is large

Can we find a better assignment of private items, to make the gap go down?
Lower the Integrality Gap?

- The integrality gap is $\Omega(\sqrt{m})$.
- But it is no more than the number of terminals.
- If we assign private items so that we have few terminals, the gap will go down!
Lower the Integrality Gap?

- The integrality gap is $\Omega(\sqrt{m})$
- But it is no more than the number of terminals
- If we assign private items so that we have few terminals, the gap will go down!

Maximum matching gives smallest possible number of terminals
Lower the Integrality Gap?

- The integrality gap is $\Omega(\sqrt{m})$
- But it is no more than the number of terminals
- If we assign private items so that we have few terminals, the gap will go down!

Maximum matching gives smallest possible number of terminals
A Revised Plan

• Compute a good assignment of items to some subset $A'$ of agents
• Remove agents of $A'$ from the instance
• Give their private items to other agents!

Number of terminals goes down
⇒ integrality gap improves!
A Revised Plan

• Compute a good assignment of items to some subset $A'$ of agents
• Remove agents of $A'$ from the instance
• Give their private items to other agents!

Problem:
Items that are assigned to agents in $A'$ may be later assigned to other agents.

Nice assignments!
On Nice Partial Assignments

Iteration 1

M

Iteration 2

M

Not nice
On Nice Partial Assignments

Iteration 1

Iteration 2

nice
Our Nice Partial Assignments

Find a collection of completely disjoint trees.
• Will not get a tree for each terminal
• For each such tree, remove A from the instance
• Reassign private items along the blue path
Our Nice Partial Assignments

Find a collection of completely disjoint trees.
• Will not get a tree for each terminal
• For each such tree, remove A from the instance
• Reassign private items along the blue path
Our Nice Partial Assignments

Find a collection of completely disjoint trees.
• Will not get a tree for each terminal
• For each such tree, remove A from the instance
• Reassign private items along the blue path
• t is not a terminal anymore!

If almost every terminal gets a tree, the number of terminals goes down fast!
A Revised Plan

• Compute a *nice* assignment of items to some subset $A'$ of agents
• Remove agents of $A'$ from the instance
• Give their private items to other agents

**Question:**
How do we find this nice assignment?

*By LP-rounding!*
LP-rounding

Problem instance

Assignment of private items

Nice assignment for set $A'$ of agents

New assignment of private items

# of terminals goes down!

$A'$

$A''$

$A'''$

...
LP-rounding

Problem instance

Assignment of private items

Nice assignment for set \( A'' \) of agents

New assignment of private items

\( A' \)
\( A'' \)

...
LP-rounding

Problem instance

Assignment of private items

Nice assignment for set $A'''$ of agents

New assignment of private items

$A'$

$A''$

$A'''$

...
LP-rounding

Problem instance —> Assignment of private items

LP-rounding

New assignment of private items

Nice assignment for set $A'''$ of agents

$1/\varepsilon$ iterations

use almost feasible solutions

New assignment of private items

combine the nice solutions in the end
From Almost Feasible Solutions to Nice Assignments

**Input:** Almost feasible solution
- A tree for every terminal
- Green and blue paths share vertices

**Output:** Nice partial solution
- A tree for almost every terminal
- The trees are completely disjoint
LP-rounding

Problem instance

Assignment of private items

1/ε iterations

use almost feasible solutions

Nice assignment for set $A'''$ of agents

New assignment of private items

combine the nice solutions in the end

$A'$

$A''$

$A'''$

...
Summary

• We have shown $\tilde{O}(n^\epsilon)$-approximation for Max Min Allocation, in $n^{O(1/\epsilon)}$ running time
  – poly-logarithmic approximation in quasi-polynomial time
• Best current hardness of approximation is 2.
• Can we use similar LP-rounding technique for other problems?

Thank you!