How to Fairly Allocate Easy and Difficult Chores

CS656 : Algorithmic Game Theory

Instructor: Sunil Easaw Simon

Introduction

- n agents, m items
- Items are indivisible *goods* or *chores*
- Allocation of items between the agents
  - $\boldsymbol{x} = (x_1, x_2, \dots, x_n)$

- Agent i values item j at $v_i(j)$
  - Goods: $v_i(j) \geq 0 \ \ \forall i$
  - Chores: $v_i(j) \leq 0 \ \ \forall i$

- Agent i values bundle S at $v_i(S)$
- Additive Utility: $v_i(S) = \sum_{s \in S} v_i(s)$

Fairness and Efficiency:

Our goal to find an allocation which is *Fair* and *Efficient*

- Fairness : Envy-Freeness (EF)
- Efficiency : Pareto-Optimality (PO)

Envy-Freeness (EF)

- **EF** : No agent prefers another one's bundle to their allocated bundle.

  $v_i(x_i) > v_i(x_j)\ \ \forall\,(i, j)$ for allocation $\boldsymbol{x} = (x_1, x_2, \dots, x_n)$

- Envy-Free allocation may not exist for indivisible chores.
- Relaxation of EF – Envy-freeness up to one item (**EF1**).

Envy-Free up to one item (EF1)

- **EF1** : No agent prefers another one's bundle to their allocated bundle, after ignoring at most one item.

- Goods: $\exists c \in x_j \ : \ v_i(x_i) \geq v_i(x_j \setminus \{c\}), \forall (i,j).$
- Chores: $\exists c \in x_i \ : \ v_i(x_i \setminus \{c\}) \geq v_i(x_j), \forall (i,j).$

- EF1 allocation always exist.

Pareto-Optimal (PO)

- Allocation $\boldsymbol{x}$ is Pareto optimal, if there is no allocation $\boldsymbol{y}$ such that $\boldsymbol{y}$ pareto dominates $\boldsymbol{x}$ i.e.,

- $\boldsymbol{y}$ pareto dominates $\boldsymbol{x}$

  $\forall i :\ v_i(y_i) \geq\ v_i(x_i)$ and $\exists j : v_j(y_j) > v_j(x_j)$

- Does a *EF1 + PO* allocation exists ?
  - **Yes**, for *goods* under additive utilities *[Caragiannis et al., 2016]*
  - Still an open question for *chores* under additive utilities
  - **Yes,** for *chores* under bivalued utilities *[This paper]*

Bivalued Utilities

- Bivalued utilities
  - $\forall i, j: \quad v_i(j) \in \{a, b\}$
  - For chores: $a \leq b \leq 0$

- Theorem 1
  - *Given a chore division problem with bivalued utilities, an EF1 + PO allocation always exists and computed in polynomial time*

Fisher Markets

- Instance I looks like:
  - n agents N,
  - m items M,
  - $\boldsymbol{p}$ price vector for items
  - Valuations $v_i(j)$, valuation of agent i for item j

- Pain per Buck: $\mathrm{PB_{ij}} = \frac{v_i(j)}{p_j}$ Maximum Pain per Buck: $MPB_i = \max_j PB_{ij}$

- Market Equilibrium $(\boldsymbol{x}, \boldsymbol{p})$
  - All items are allocated
  - Agents only receive MPB items

Fisher Markets

- First Welfare Theorem
  - *Every equilibrium allocation in Fisher market is Pareto Optimal*

- Price envy-freeness up to one item (**pEF1**)
  - A allocation is pEF1 if, for all $i, j \in N$, $\exists c \in x_j : p\left(x_j \setminus \{c\}\right) \leq p(x_i)$

- Lemma:
  - *If an allocation is pEF1 then it is EF1*

pEF1 + equilibrium → EF1 + PO

Algorithm for EF1 + PO

Phase 1 Initialization

Let x be an allocation maximizing social welfare $\sum_{i \in \mathcal{N}} v_i(x_i)$.
For each $c \in \mathcal{M}$, let p_c = p · $|\max_{i \in \mathcal{N}} v_i(c)|$
k ← 1, the number of the current iteration

- We start with an allocation and prices in equilibrium $(\boldsymbol{x}, \boldsymbol{p})$
- If $c \in x_i$ then $v_i(c) = \max_j v_j(c)$

$$PB_i(c) = |v_i(c)|/(p \cdot |v_i(c)|) = 1/p$$

Algorithm for EF1 + PO

Phase 2a Reallocate chores

```
for ℓ ∈ (k − 2, k − 3, . . . , 2, 1) do
  while true do
    i ← an agent from arg max_i∈H_ℓ p_up to 1(x_i)
    j ← an agent from arg min_j∈H_{ℓ+1}∪···∪H_{k−1} p(x_j)
    if p_up to 1(x_i) > p(x_j) then
      c ← any item from x_i \ entitled(i)
      Transfer c from i to j
    else
      break
```

Phase 2b Reallocate chores

```
while true do
  ls ← an agent from arg min_i∈𝒩 p(x_i)
  if there is an MPB alternating path i →c1 j →c2 ··· →cl ls with
    p_up to 1(x_i1) > p(x_ls) then
    Choose such a path of minimum length ℓ
    Transfer c_ℓ from i_ℓ to i_{ℓ−1}
  else
    break
if x satisfies pEF1 then
  return x
```

- Keeping the prices constant
- Reallocate chores to decrease envy

- MPB-path $i \overset{c_1}{\rightarrow} j \overset{c_2}{\rightarrow} \cdots \overset{c_l}{\rightarrow} k$

- Local Changes
- $i \overset{c}{\rightarrow} j$ exists and $p(x_j) < p(x_i) - p_c$

  then transferring c to j reduces envy

  and remain in equilibrium

Algorithm for EF1 + PO

Phase 3 Price reduction

$H\_k \leftarrow \{i \in \mathcal{N} : \text{there is an agent ls} \in \arg\min_i \in \mathcal{N}\ p(x_i) \text{ with ls} \rightsquigarrow i\}$

▶ Timestamp: t_{k,b}

$\alpha \leftarrow \min\{PB_i(c)/MPB_i : i \in H\_k,\ c \in \cup_j \in \mathcal{N} \setminus H\_k\ x_j\}$
for $i \in H\_k$ do
  entitled(i) $\leftarrow x_i$
  for $c \in x_i$ do
    $p\_c \leftarrow 1/\alpha \cdot p\_c$

▶ Timestamp: t_{k,a}

$k \leftarrow k + 1$
Start Phase 2a (i.e. go to line 5)

- Keeps the allocation $\boldsymbol{x}$ fixed
- Identify set of agents and reduce prices of chores allocated to them by multiplicative factor

- Each time we identify different set of agents
- Algorithm terminates in poly. steps

Algorithm EF1 + PO

- Algorithmic Frame work:
  - Start with an allocation and price in equilibrium
  - Make local changes reducing envy (equilibrium maintained)
  - Reach pEF1 (+ equilibrium) → terminate

- Correctness is proved by induction of some properties on iterations

Another Fairness Notion: Maximin Share

- Maximin Share (MMS) allocation *[Budish, 2011]*
- P: Set of all partitions of chores/goods M into n bundles
- Agent i's maximin share (aka **MMS value**) is

$$\mathrm{MMS}_i = \max_{p \in P} \min_{k \in [n]} v_i(p_k)$$

- $\boldsymbol{x}$ is an MMS allocation if $v_i(x_i) \geq \mathrm{MMS_i} \quad \forall i$

- Finding MMS values is **NP-hard**
- MMS allocations may not exist in general. *[Kurokowa et al '16]*

Factored Utilities

- Factored Utilities:

  $$v_{ij} \in \{1, p_1, p_2, \cdots, p_k\} \subset \mathbb{Z} \text{ such that } p_{i-1} \mid p_i \quad \forall i$$

  - Eg: {1, 2, 6, 12, 36}

- Theorem
  - *For factored utilities $\boldsymbol{v}$ over a set of items M (all goods or all chores), an MMS values can be found in polynomial time*

Algorithm for MMS values

$x \leftarrow (x_i = \emptyset)_{i \in \mathcal{N}}$
for $r \in \mathcal{M}$ in a nonincreasing order of $|v(r)|$ do
    $k^* \leftarrow \arg\min_{k \in \mathcal{N}} |v(x_k)|$
    $x_{k^*} \leftarrow x_{k^*} \cup \{r\}$
return x

- Items are sorted in nonincreasing order
- We model the problem as packing items into bundles such worst bundle is as good as possible

MMS allocation

- MMS allocation is not guaranteed to exist for all instances of additive utilities

- So we will discuss MMS allocation for two special subclasses of Factored Utilities
    - Personalised Factored Bi-valued Utilities
    - Weakly lexicographic Utilities

Personalized Factored Bivalued

- $v_{ij} \in \{a_i, b_i\}$ and $\exists k \in \mathbb{N}$ such that $a_i = b_i \cdot k$
  - Note that $a_i, b_i \in \mathbb{Z}$

- Theorem 2(a)
  - *For personalised factored bivalued chores or goods*
    - *MMS allocation always exists*
    - *MMS allocation can be found in PTIME*

Weakly Lexicographic Utilities

- Divides items into sets/levels
  - $\{a, b\} \succ \{c, d, e\} \succ \{f\} \succ \{g, h\}$

  - $\forall i \quad v_i(c) = v_i(d) = v_i(e)$
  - $\forall i \quad |v_i(a)| > \sum_{m \prec a} |v_i(m)|$

- Theorem 2(b)
  - *For weakly lexicographic chores or goods*
    - *MMS allocation always exists*
    - *MMS allocation can be found in PTIME*

MMS allocation

- We make some valid reductions to reduce the problem

- $I = (N, M, \boldsymbol{v}) \rightsquigarrow I' = (N-1, M \setminus S, \boldsymbol{v})$ is valid if
  - $v_i(S) \geq \text{MMS}_i^n(M)$ and
  - $\text{MMS}_j^{n-1}(M \setminus S) \geq \text{MMS}_j^n(M) \quad \forall j \in N \setminus \{i\}$

- If $\boldsymbol{x}$ is an MMS allocation, and $\boldsymbol{x}'$ is a Pareto improvement over $\boldsymbol{x}$ then $\boldsymbol{x}'$ is also MMS
- For weakly lexicographic and factored bivalued utilities, given an MMS allocation, an MMS + PO allocation can be computed in poly time

MMS + PO

- If $\boldsymbol{x}$ is an MMS allocation, and $\boldsymbol{x}'$ is a Pareto improvement over $\boldsymbol{x}$ then $\boldsymbol{x}'$ is also MMS

- For weakly lexicographic and factored bivalued utilities, given an MMS allocation, an MMS + PO allocation can be computed in poly time

- We give a polynomial time algorithm find these pareto improvements

Summary

- EF1 + PO for bivalued chores:
  - Can be found in polynomial time

- MMS values for factored utilities can be found in polynomial time

- MMS+PO exists for two subclasses of factored utilities
  - Factored bivalued and weakly lexicographic and can be found in poly. Time

- For personalized bivalued utilities, finding MMS+PO allocations in polynomial time remains an open question

Thank You!

MMS + PO

- We start with the MMS allocation and create a directed graph where edges represent an agent preferring another agent's item.
- We then repeatedly look for cycles, which can be done efficiently in poly. time
- By reallocating items along these cycles, we make Pareto improvements while preserving MMS fairness