Lecture 13: *Algorithmic Game Theory & Mechanism Design*

Advanced Algorithms

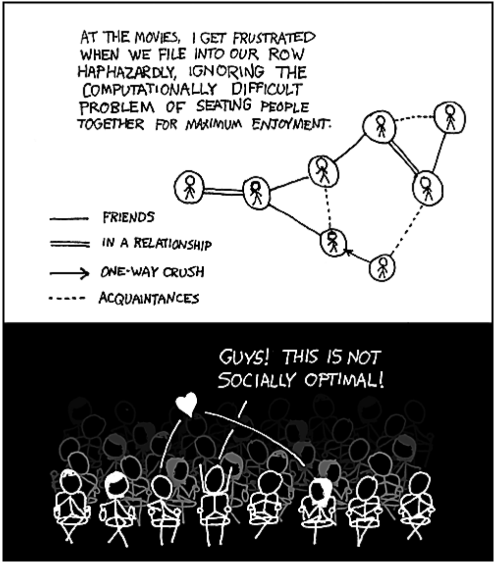
Sid Chi-Kin Chau

Australian National University

sid.chau@anu.edu.au

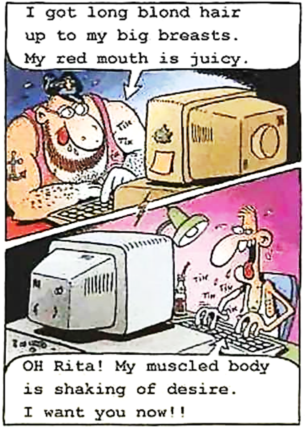
October 25, 2022

Making Socially Optimal is Hard



Algorithmic Mechanism Design

- Our society is a *computer* that decides who should receive what resource, welfare, duties, rewards and penalties
  - *Input*: Individual preferences, rules, constraints of inter-dependence, limitations and capacities
  - *Output*: Decisions of allocations, contributions and liability
- Individuals can manipulate the input to distort the output decisions for their benefits
- Possible manipulations:
  - Lying: falsifying personal preferences
  - Collusion: coordinating joint manipulations
- Consequence of manipulations:
  - Inefficiency, conflicts, failure of social mechanisms
- *Algorithmic Mechanism Design*: How to design desirable social mechanisms from an algorithmic perspective



Example: Auction

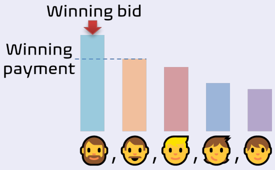
Example (Auction)

- Simplest setting of resource allocation
- Consider a single item being auctioned for sale among n buyers
- Buyer i has a valuation v_i (“willingness to pay” for the item)
  - If i wins, but has to pay price p, then i's utility is $u_i = v_i - p$
  - If someone else wins, then i's utility is $u_i = 0$
- A natural choice is to select the buyer of the highest declared valuation
  - Choose winner $i = \arg\max_j v_j$
- Considering lying (no collusion):
  - If p is fixed, then each i whose $v_j > p$ has incentive to report more than true value $v'_i > v_i$
  - If p is proportional to the highest valuation $(\max_j v_j)$, then each i has incentive to report less than true value $v'_i < v_i$
- **Incentive compatibility**: Lying is not better, and thus reveal the true v_j for each i

Vickrey's Second Price Auction

Definition (Second Price Auction)

- Winner (say i) is the buyer with the highest declared valuation (bid) of v_i
- But i pays the second highest declared bid $p^* = \max_{j \neq i} v_j$
- Assume no collusion among bidders



Lemma (Incentive Compatibility)

Vickrey's second price auction is incentive compatible – each bidder i declares his true valuation, and achieves the highest utility than other declared valuations

Proof:

- Declare $v'_i > v_i$: if i wins, i pays the same p^*; if i loses, utility is same as 0
- Declare $v'_i < v_i$: the same argument applies

Incentive Compatible Mechanism

Definition (General Social Mechanism)

- Let A be a set of feasible allocations (e.g., possible winning bidders), then the valuation of player i is modeled by a function $v_i(\cdot) : A \mapsto \Re$ (where $\Re$ is the set of real numbers)
- Let $V_i \subseteq \Re^{|A|}$ be a set of feasible valuation functions for player i
- Let $v \triangleq (v_1, ..., v_n)$, $v_{-i} \triangleq (v_1, ..., v_{i-1}, v_{i+1}, ..., v_n)$, $(v_i, v_{-i}) \triangleq (v_1, ..., v_n)$
- A mechanism is a social choice function $f : V_1 \times ... \times V_n \mapsto A$ and payment functions $p_1, ..., p_n$, where $p_i : V_1 \times ... \times V_n \mapsto \Re$ is the amount of player i needs to pay
- A mechanism $(f, p_1, ..., p_n)$ is called *incentive compatible* if for every player i, every $v_i(\cdot) \in V_i$ and every $v'_i(\cdot) \in V_i$, then

$$v_i(a) - p_i(v_i, v_{-i}) \geq v_i(a') - p_i(v'_i, v_{-i})$$

where $a \triangleq f(v_i, v_{-i})$ and $a' \triangleq f(v'_i, v_{-i})$

VCG Mechanism

- Intuitively, incentive compatibility makes player i prefer reporting his true valuation (v_i), rather than any "lie" (v'_i)
- How do we achieve incentive compatibility?

Definition (VCG Mechanism)

A mechanism $(f, p_1, ..., p_n)$ is called a Vickrey-Clarke-Groves (VCG) mechanism, if

- $f(v_1, ..., v_n) \in \arg\max_{a \in A} \sum_i v_i(a)$; f maximizes the social welfare,
- and for all $v_1 \in V_1, ..., v_n \in V_n$,

$$p_i(v_1, ..., v_n) = h_i(v_{-i}) - \sum_{j \neq i} v_j(f(v_1, ..., v_n))$$

where $h_1, ..., h_n$ are some functions, such that $h_i : V_{-i} \mapsto \Re$ (i.e., h_i does not depend on player i's valuation v_i)

VCG Mechanism

Theorem (Incentive Compatibility)

VCG mechanism satisfies incentive compatibility

Proof:

- Given i, v_{-i}, v_i and v_i', we show that for player i with valuation v_i, the utility when declaring v_i is not less than the utility when declaring v_i'
- Denote $a \triangleq f(v_i, v_{-i})$ and $a' \triangleq f(v_i', v_{-i})$
- The utility of i, when declaring v_i, is $v_i(a) + \sum_{j \neq i} v_j(a) - h_i(v_{-i})$, but when declaring v_i' is $v_i(a') + \sum_{j \neq i} v_j(a') - h_i(v_{-i})$
- But since a maximizes social welfare over all alternatives, we have

$$v_i(a) + \sum_{j \neq i} v_j(a) - h_i(v_{-i}) \geq v_i(a') + \sum_{j \neq i} v_j(a') - h_i(v_{-i})$$

VCG Mechanism

Definition (Clarke Pivot Rule)

- A mechanism is (ex-post) *individually rational*, if players always get nonnegative utility; if for every $v_1, \ldots, v_n$ we have that $v_i(f(v_1, \ldots, v_n)) - p_i(v_1, \ldots, v_n) \geq 0$
- A mechanism has *no positive transfers*, if no player is ever paid money: if for every $v_1, \ldots, v_n$ and every i, $p_i(v_1, \ldots, v_n) \geq 0$
- **Clarke Pivot Rule**:

$$h_i(v_{-i}) \triangleq \max_{b \in A} \sum_{j \neq i} v_j(b)$$

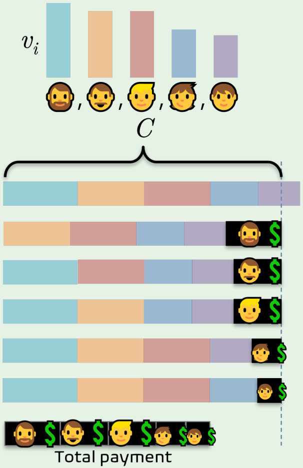
So the payment is $p_i(v_1, \ldots, v_n) = \max_{b \in A} \sum_{j \neq i} v_j(b) - \sum_{j \neq i} v_j(a)$, where $a \triangleq f(v_1, \ldots, v_n)$

Lemma

VCG mechanism with Clarke pivot payment satisfies no positive transfers & individual rationality

VCG Mechanism: Example

Example (Cost Sharing)



- Sharing cost (C) of a public project (e.g., a bridge), if it is built
- Valuation of player i: $v_i \geq 0$
- Social choice: $f(v_1, ..., v_n)$ = build the public project, if $\sum_i v_i \geq C$; otherwise, $f(v_1, ..., v_n)$ = don't build
- VCG mechanism with $h_i(v_{-i}) = C$:
  - $p_i(v_1, ..., v_n) = 0$, if $\sum_{j \neq i} v_j \geq C$ or $\sum_i v_i < C$
  - $p_i(v_1, ..., v_n) = C - \sum_{j \neq i} v_j$, if $\sum_{j \neq i} v_j < C$ and $\sum_i v_i \geq C$
- But it is possible that $\sum_j p_j < C$ (i.e. cannot recover C)
  - E.g., if $v_i = \frac{C}{n-1}$, then $p_i = 0$, hence, external subsidy is needed

VCG Mechanism: Example

Example (Reserving a Path in Network)

- Given graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where each link $e \in \mathcal{E}$ is owned by a owner e, and has a cost $c_e \geq 0$ if his link is reserved
- Each owner declares c_e
- We want to reserve a s-t path in $\mathcal{G}$ from source s to destination t
- We find P^*, the shortest path s-t path in $\mathcal{G}$ (in terms of $\sum_{e \in p} c_e$)
- VCG mechanism with Clarke pivot rule means that each $e \in P^*$ will earn a payment as

$$p_e = \sum_{e' \in \hat{P}(e)} c_{e'} - \sum_{e' \in P^* \setminus \{e\}} c_{e'}$$

  where $\hat{P}(e)$ is the shortest s-t path in $\mathcal{G}$ that does not contain the edge e
- Note that it is possible that $c_e > p_e$ (owner may not be able to recover cost)

Randomized Incentive Compatible Mechanism

Definition (Randomized Incentive Compatible Mechanism)

- A randomized mechanism is a distribution over deterministic mechanisms (all with the same players, types spaces V_i and outcome space A)
- A randomized mechanism is incentive compatible in expectation, if for all i, all v_i, v_{-i}, and v'_i, we have

$$\mathbb{E}[v_i(a) - p_i] \geq \mathbb{E}[v_i(a') - p'_i]$$

where (a, p_i), and (a', p'_i) are random variables denoting the outcome and payment when i bids, respectively, v_i and v'_i, and $\mathbb{E}[\cdot]$ denotes expectation over the randomization of the mechanism

Profit Maximizing Mechanism

- We consider mechanisms that maximize profit of a mechanism controller
- Each player i has a private valuation v_i
- Decide allocation $x = (x_1, ..., x_n)$ where $x_i \in \{0, 1\}$, and payment $p = (p_1, ..., p_n)$
- Player i receives utility: $u_i = v_i x_i - p_i$
- Profit of mechanism controller is $\sum_i p_i - c(x)$, where $c(\cdot)$ is a cost function

Example

- Single item auction: $c(x) = 0$ if $\sum_i x_i \leq 1$, otherwise $c(x) = \infty$
- Unlimited digital goods: Selling software, games, movie streaming: $c(x) = 0$

Single-Priced Auctions for Digital Goods

- Consider zero cost $c(x) = 0$
- Charge all winners the same price: set $p_i = p^*$ when $x_i = 1$, otherwise set $p_i = 0$
- Without loss of generality, we assume descending order of private valuations: $v_1 \geq ... \geq v_n$
- Optimal single-priced profit function: $\mathcal{F}(v) \triangleq \max_{i=1,\dots,n} i \cdot v_i$
- Optimal profit function: $\mathcal{T}(v) \triangleq \sum_{i=1}^{n} v_i$

Lemma

$\ln(n) \cdot \mathcal{F}(v) \geq \mathcal{T}(v)$ *for all* v *(that is,* $\mathcal{F}(v)$ *is* $(\ln(n))$*-competitive to* $\mathcal{T}(v)$*)*

Proof:

- Consider $v_i = \frac{n}{i}$ which induces $\ln(n)$ as competitive ratio
- Suppose $\mathcal{F}(v) = k \cdot v_k$, namely $k \cdot v_k \geq i \cdot v_i$ for all i
- Therefore, $\mathcal{T}(v) = \sum_i v_i \leq \sum_i \frac{k \cdot v_i}{i} \leq \mathcal{F}(v) \sum_i \frac{1}{i}$

Single-Priced Auctions for Digital Goods

- How do we design an incentive compatible mechanism $f : v \mapsto (x, p)$ that maximizes competitive ratio compared to $\mathcal{F}(v)$ over all instances?

Bid Independent $\mathcal{A}_{\mathsf{bi}}(f)$

- For ech player i
  - Find $p_i^* \leftarrow f(v_{-i})$
  - If $v_i \geq p_i^*$ then
    - Set $x_i \leftarrow 1$ and $p_i \leftarrow p_i^*$
  - Else set $x_i \leftarrow 0$ and $p_i \leftarrow 0$

- $f(v_{-i})$ sets bid for each i without relying on v_i
- E.g., set $f(v_{-i}) = v_{\hat{j}(i)}$ where $\hat{j}(i) = \arg\max_{i \neq j} j \cdot v_j$

Single-Priced Auctions for Digital Goods

Lemma

There exists no incentive compatible mechanism with a constant competitive ratio to $\mathcal{F}(v)$

Idea:

- We can show that all incentive compatible mechanisms are equivalent to bid independent mechanism $\mathcal{A}_{\mathsf{bi}}(f)$
- Consider two players $(v_1 = 1, v_2 = V)$; we take $V \to \infty$
- Hence, we need to consider an alternative: profit function with at least two winners

$$\mathcal{F}_2(v) \triangleq \max_{i=2,\ldots,n} i \cdot v_i$$

- But can we design an incentive compatible mechanism with a constant competitive ratio to $\mathcal{F}_2(v)$?

Single-Priced Auctions for Digital Goods

Random Sampling $\mathcal{A}_{\text{rand2}}$

- Randomly put each i into subsets A_1 or A_2 with probability $\frac{1}{2}$
- Find $\hat{j}_1 \leftarrow \arg\max_{j \in A_1} j \cdot v_j$
- Find $\hat{j}_2 \leftarrow \arg\max_{j \in A_2} j \cdot v_j$
- If $i \in A_2$ and $v_i \geq v_{\hat{j}_1}$ then
  - Set $x_i \leftarrow 1$ and $p_i \leftarrow v_{\hat{j}_1}$
- If $i \in A_1$ and $v_i \geq v_{\hat{j}_2}$ then
  - Set $x_i \leftarrow 1$ and $p_i \leftarrow v_{\hat{j}_2}$
- Else set $x_i \leftarrow 0$ and $p_i \leftarrow 0$

Single-Priced Auctions for Digital Goods

Lemma

The competitive ratio of $\mathcal{A}_{\text{rand2}}$ is 4 for $\mathcal{F}_2(v)$

Proof:

- Suppose $\mathcal{F}_2(v) = \tilde{k} \cdot v_{\tilde{k}}$, where buyers $i = 1, .., \tilde{k}$ are selected by $\mathcal{F}_2(v)$
- We study worst case: $\min\{\text{Profit}(A_1), \text{Profit}(A_2)\}$
- Buyers $i = 1, .., \tilde{k}$ are randomly selected in A_1 or A_2 with probability $\frac{1}{2}$
- Let $\tilde{K}_1$ be the random number of buyers in $\{1, .., \tilde{k}\}$ selected in A_1 (similarly, $\tilde{K}_2$ for A_2)

$$\frac{\mathbb{E}[\text{Profit}(\mathcal{A}_{\text{rand2}})]}{\mathcal{F}_2(v)} \geq \frac{\mathbb{E}[\min\{\tilde{K}_1, \tilde{K}_2\}]}{\tilde{k}} = \frac{1}{\tilde{k}} \sum_{i=1}^{\tilde{k}-1} \min\{i, \tilde{k} - i\} \binom{\tilde{k}}{i} \frac{1}{2^{\tilde{k}}} = \frac{1}{2} - \binom{\tilde{k} - 1}{\lfloor \frac{\tilde{k}}{2} \rfloor} \frac{1}{2^{\tilde{k}}}$$

where the minimum is attained when $\tilde{k} = 2$ and $\frac{\mathbb{E}[\min\{\tilde{K}_1, \tilde{K}_2\}]}{\tilde{k}} = \frac{1}{4}$

Single-Priced Auctions for Digital Goods

Example

- The competitive ratio of $\mathcal{A}_{\text{rand2}}$ to $\mathcal{F}_2(v)$ as 4 is tight
- Consider a set of bidders consisting of two very high bids h and $h + \epsilon$, and all other bids are negligibly small
- Then we have $\mathcal{F}_2(v) = 2h$
- Therefore,

$$\begin{aligned}\mathbb{E}[\text{Profit}(\mathcal{A}_{\text{rand2}})] &= h \cdot \mathbb{P}(\text{two high bidders are split in } A_1 \text{ and } A_2) \\ &= \frac{h}{2}\end{aligned}$$

Combinatorial Auction

- Consider a multiple-in-one auction: Multiple items for sale, bidders can express preferences on bundles of items
- Examples: Wireless spectrum, bus routes, holiday itinerary

Definition

- $\mathcal{U}$ is a set of m items
- $\mathcal{N}$ is a set of n bidders
- For each $j \in \mathcal{N}, v_j : 2^{\mathcal{U}} \mapsto \Re^+$ is bidder valuation
- Decide allocation $(S_1, ..., S_n)$ and payment $(p_1, ..., p_n)$
- Maximizing social welfare $W = \sum_j v_j(S_j)$
- Assume each bidder has quasi-linear utility: $u_j = v_j(S_j) - p_j(S_j)$

Multi-unit Auction

Definition (Multi-unit Auction)

- There are m identical copies of items are for sale
- Each bidder j is willing to pay v_j for quantity q_j items
- This is equivalent to a knapsack problem

Definition (Multi-unit Combinatorial Auction)

- There are m types of items are for sale
- Each type has m_i identical items for sale
- Each bidder j is willing to pay v_j for quantity q_{ij} items for all m types
- This is m-dimensional knapsack problem

Combinatorial Auction

- Bidder j is *single-minded* if there exists some $S_j^* \subseteq \mathcal{U}$

$$v_j(S) = \begin{cases} v_j^*, & \text{if } S_j^* \subseteq S \\ 0, & \text{otherwise} \end{cases}$$

- We assume every bidder j is single-minded, and S_j^* is known to the auctioneer

Definition (Single-minded Combinatorial Auction)

- Input: bids $v = (v_1, ..., v_n)$
- Output: winning bidders $\mathcal{A}(v) \subseteq \mathcal{N}$
  - Subject to winning bids are pairwise disjoint
  - $S_j \cap S_k = \varnothing$ for $j, k \in \mathcal{A}(v)$
- Decide payment rule $p_j(S_j)$
- Maximize social welfare: $W = \sum_{j \in \mathcal{A}(v)} v_j$
- Rationality: losers should pay nothing

Incentive Compatible Mechanism

Definition (Incentive Compatible Mechanism)

- A mechanism is called **incentive compatible** (or truthful), if for all bidders j, v_{-j}, v_j, and any other v'_j
$$u_j(v_j, v_{-j}) \geq u_j(v'_j, v_{-j})$$
- An allocation algorithm $\mathcal{A}$ is called *monotone*, when for all bidders j and v_{-j}, if v_j is a winning bid, then $v'_j \geq v_j$ is a winning bid

Theorem

A mechanism is incentive compatible, if and only if

- *Allocation algorithm $\mathcal{A}(v)$ is monotone*
- *Payment $p_j(S_j)$ is set as a critical value that does not depend on j*

Monotone Allocation Algorithm

Definition

A mechanism $(\mathcal{A}, p)$ is normalized, if losers pay zero. Alternatively, we can set normalized payment by $p'_j(S) = p_j(S) - p_j(\varnothing)$

Theorem

A normalized mechanism $(\mathcal{A}, p)$ is incentive compatible, if and only if $\mathcal{A}$ is monotone and its payment $p_j(S_j)$ is set as a critical value that does not depend on j

- Monotone allocation algorithms can capture almost all incentive compatible mechanisms
- Sufficient and necessary to consider monotone allocation algorithms

Monotone Allocation Algorithm

Theorem

A normalized mechanism $(\mathcal{A}, p)$ is incentive compatible, if and only if $\mathcal{A}$ is monotone and its payment $p_j(S)$ is set as the critical value that does not depend on j

Idea:

Lemma

If $\mathcal{A}$ is monotone, then exists a unique critical value $\theta(v_{-j})$ such that

- *For all $v_j < \theta(v_{-j})$, v_j is a losing bid*
- *For all $v_j > \theta(v_{-j})$, v_j is a winning bid*

- The payment scheme $p(\cdot)$ based on the criticial value is

$$p_j(S) = \begin{cases} \theta(v_{-j}), & \text{if } S_j^* \subseteq S \text{ (i.e., } j \text{ wins)} \\ 0, & \text{otherwise} \end{cases}$$

Monotone Allocation Algorithm

- Monotone is a generalization of concept of VCG mechanisms
- VCG mechanisms can be computationally inefficient
  - Finding social optimal is hard in NP-Hard problems
- Approximation algorithms are not necessary VCG mechanisms
- But approximation algorithms can induce incentive compatible mechanisms, if they are monotone
- Note that not every approximation algorithm is monotone
- Goal: Monotonize known approximation algorithms
  - Hot research topic, a lot of smart ideas

Greedy Algorithm

Greedy Algorithm $\mathcal{A}_{\text{gre}}$

- Reorder bids by decreasing $r_j \triangleq \frac{v_j}{|S_j^*|}$
- $\text{WinningBids} \leftarrow 0, \text{NonAllocItems} \leftarrow \mathcal{U}$
- For each j in new order, if $S \subseteq \text{NonAllocItems}$
  - $\text{WinningBids} \leftarrow \text{WinningBids} \cup \{j\}$
  - $\text{NonAllocItems} \leftarrow \text{NonAllocItems} \backslash S_j^*$
- Return WinningBids

Lemma

If ranking $\{r_j\}$ is monotone, then $\mathcal{A}_{\text{gre}}$ is monotone

Proof:

- Since ranking $\{r_j\}$ is monotone in the bid v_j, increasing v_j can only move it closer to the beginning of the ranking

Bitonic Allocation Algorithm

Definition

- An allocation algorithm $\mathcal{A}(v)$ is *bitonic*, if for any v_{-j} either
  - Welfare $W = \sum_{j \in \mathcal{A}(v)} v_j$ is non-decreasing in $v_j < \theta(v_{-j})$ & non-increasing in $v_j \geq \theta(v_{-j})$, or
  - Welfare $W = \sum_{j \in \mathcal{A}(v)} v_j$ is non-increasing in $v_j \leq \theta(v_{-j})$ & non-decreasing in $v_j > \theta(v_{-j})$

Intuition:

- If j is winning (i.e. $v_j > \theta(v_{-j})$), then welfare W should be increasing in v_j
- If j is losing (i.e. $v_j < \theta(v_{-j})$), then welfare W should be independent of v_j (i.e. non-increasing)

Bitonic Allocation Algorithm

Example

- Allocation function can be monotone, but not bitonic
- Consider three bidders i, j, k
- $\mathcal{A}_{\mathsf{XOR}}(y, i, j, k) = \begin{cases} j \text{ wins }, & \text{if } v_i < y \\ k \text{ wins }, & \text{if } y \le v_i < 2y \\ i \text{ wins }, & \text{otherwise} \end{cases}$
- $\mathcal{A}_{\mathsf{XOR}}(y, i, j, k)$ is monotone for i, j, k
- $\mathcal{A}_{\mathsf{XOR}}(y, i, j, k)$ is not bitonic, if $v_j < v_k$

Combination of Basic Algorithm: Max Operator

$\max(\mathcal{A}_1, \mathcal{A}_2)$

- $W_1 \leftarrow \mathcal{A}_1(v)$
- $W_2 \leftarrow \mathcal{A}_2(v)$
- If $W_1 \geq W_2$, return $\mathcal{A}_1(v)$
- Else return $\mathcal{A}_2(v)$

Lemma

If $\mathcal{A}_1, \mathcal{A}_2$ are monotone bitonic allocation algorithms, then $\max(\mathcal{A}_1, \mathcal{A}_2)$ is monotone and bitonic

Combination of Basic Algorithm: Max Operator

Example

- $\mathcal{A}_1, \mathcal{A}_2$ are monotone, but $\max(\mathcal{A}_1, \mathcal{A}_2)$ is not monotone
- Consider $v_j < v_i < v_k$
- Consider $\mathcal{A}_{\textsf{XOR}}(y, i, j, k)$ and $\mathcal{A}_{\textsf{XOR}}(4y, i, j, k)$, both are monotone
- $\max\Big(\mathcal{A}_{\textsf{XOR}}(y, i, j, k), \mathcal{A}_{\textsf{XOR}}(4y, i, j, k)\Big)$ is not monotone
- If $2y < v_i' < 4y$, then i wins; but If $4y < v_i' < 8y$, then k wins

Combination of Basic Algorithm: Max Operator

Lemma

If $\mathcal{A}_1, \mathcal{A}_2$ are monotone bitonic allocation algorithms, then $\max(\mathcal{A}_1, \mathcal{A}_2)$ is monotone and bitonic

Proof:

- Let critical values for $\mathcal{A}_1, \mathcal{A}_2$ be $\theta_1(v_{-j})$ and $\theta_2(v_{-j})$
- Suppose $\theta_1(v_{-j}) < \theta_2(v_{-j})$
- There always exists $\theta_1(v_{-j}) < \theta_{\max}(v_{-j}) < \theta_2(v_{-j})$
  - If j is winning in $\max(\mathcal{A}_1, \mathcal{A}_2)$ (i.e. $v_j > \theta_{\max}(v_{-j})$), then welfare W of $\max(\mathcal{A}_1, \mathcal{A}_2)$ should be increasing in v_j
  - If j is losing $\max(\mathcal{A}_1, \mathcal{A}_2)$ (i.e. $v_j < \theta_{\max}(v_{-j})$), then welfare W of $\max(\mathcal{A}_1, \mathcal{A}_2)$ should be should be non-increasing in v_j

Multi-unit Auction

Definition (Multi-unit Auction)

- There are m identical copies of items are for sale
- Each bidder j is willing to pay v_j for quantity q_j items

Approx-MUA

- Let $\mathcal{A}_{\mathsf{greV}}$ be Greedy based on value ranking $r_i = v_i$
- Let $\mathcal{A}_{\mathsf{greD}}$ be Greedy based on density ranking $r_i = \frac{v_i}{q_j}$
- Return $\max(\mathcal{A}_{\mathsf{greV}}, \mathcal{A}_{\mathsf{greD}})$

Algorithm for Combinatorial Auction: Approx-MUA

Lemma

Approx-MUA *is monotone*

Proof:

- $\mathcal{A}_{\text{greV}}$ and $\mathcal{A}_{\text{greD}}$ are monotone and bitonic
- $\max(\mathcal{A}_{\text{greV}}, \mathcal{A}_{\text{greD}})$ is monotone

Lemma

Approx-MUA *is a 2-approximation for multi-unit auction problem*

Proof:

- $\mathcal{A}_{\text{greD}}$ produces round-off solution to knapsack problem
- The utility for boundary round-off item is upper bounded by $\mathcal{A}_{\text{greV}}$

Partial Exhaustive Search

Partial Exhaustive Search $\mathcal{A}_{\text{exh}}(k)$

- WinningBids $\leftarrow 0, \max \leftarrow 0$
- For each $J \subseteq \{1, ..., n\}$ subject to $|J| \leq k$
  - If the S_j's are pairwise disjoint and $(\sum_j v_j > \max)$
    - $\max \leftarrow \sum_j v_j$
    - WinningBids $\leftarrow J$
- Return WinningBids

Lemma

For every k, $\mathcal{A}_{\text{exh}}(k)$ is monotone and bitonic

Proof:

- If j is winning, increasing v_j still wins and increases social welfare
- If j is losing, decreasing v_j still loses and cannot change social welfare

LP Based Algorithm

Definition (Multi-unit Combinatorial Auction)

- There are m types of items are for sale
- Each type has m_i identical items for sale
- Each bidder j is willing to pay v_j for quantity q_{ij} items for all m types

Definition

- Define LP(v) problem:
  - Find (x_j)

$$\max \sum_j v_j x_j$$

subject to $\sum_j x_j q_{i,j} \leq m_i$ for all $i = 1, ..., m$, and $0 \leq x_j \leq 1$

- Compute an optimal solution x for LP(v)
- Satisfy all bids j for which $x_j = 1$

LP Based Algorithm

Theorem

LP-Based algorithm is monotone

Lemma

For any v_{-j}, x_j is a non-decreasing function of v_j

Proof:

- $\sum_{l=1}^{n} x'_l v_l \leq \sum_{l=1}^{n} x_l v_l$ because x' is not an optimal solution by misreporting
- Also, $\sum_{l=1}^{n} x_l v'_l \leq \sum_{l=1}^{n} x_l v'_l$
- Hence, $x_j(v'_j - v_j) + \sum_{l=1}^{n} x_l v_l \leq x'_j(v'_j - v_j) + \sum_{l=1}^{n} x'_l v_l$
- $0 \leq \sum_{l=1}^{n}(x_l - x'_l)v_l \leq (x'_j - x_j)(v'_j - v_j)$
- Therefore, $x'_j \geq x_j$ if $v'_j \geq v_j$
- Finally, if x_j is 1 for some v_j, then it is also for $v'{}_j > v_j$

Combination of Basic Algorithm: If-Then-Else

Lemma

If $\mathcal{A}_1, \mathcal{A}_2$ are monotone allocation algorithms and $\mathrm{Cond}(\cdot)$ *is aligned with $\mathcal{A}_\infty$, then If-Then-Else*$(\mathrm{Cond}, \mathcal{A}_1, \mathcal{A}_2)$ *is monotone*

Proof:

- Suppose $v_j \leq v'_j$
- If $\mathrm{Cond}(v_{-j}, v_j)$ is true and $j \in \mathcal{A}_1$, then $\mathrm{Cond}(v_{-j}, v'_j)$ and If-Then-Else$(\mathrm{Cond}(v_{-j}, v'_j), \mathcal{A}_1, \mathcal{A}_2)$ will output $\mathcal{A}_1(v_{-j}, v'_j)$
- If $\mathrm{Cond}(v_{-j}, v_j)$ is false and $\mathrm{Cond}(v_{-j}, v'_j)$ is false, then $\mathrm{Cond}(v_{-j}, v'_j)$ and If-Then-Else$(\mathrm{Cond}(v_{-j}, v'_j), \mathcal{A}_1, \mathcal{A}_2)$ will output $\mathcal{A}_2(v_{-j}, v'_j)$
- If $\mathrm{Cond}(v_{-j}, v_j)$ is false and $\mathrm{Cond}(v_{-j}, v'_j)$ is true, this is a contradiction

Approx Algorithm for Multi-unit Combinatorial Auction

Approx-MUCA

- Compute an optimal vertex solution x to LP(v)
- Let $v_h = \max_j v_j$
- If $\sum_j x_j v_j < (m+1)v_h$
  - Return Largest(v)
- Else return LP(v) based rounding solution

Lemma

The IF condition is aligned with Largest, thus Approx-MUCA *is monotone*

Lemma

Approx-MUCA *is* $(m+1)$ *approximation algorithm*

References

Reference Materials

- Algorithmic Game Theory (Edited by Nisan, Roughgarden, Tardos, Vazirani)
  - Chapters 9-11

Recommended Materials

- "*Competitive Generalized Auctions*", (Goldberg, Hartline, Karlin, Saks, Wright), STOC, 2002
- "*Truthful Approximation Mechanisms for Restricted Combinatorial Auctions*", (Mu'alem, Nissan), AAAI, 2002