Implementation in Advised Strategies:
Welfare Guarantees from Posted-Price Mechanisms when Demand Queries are NP-hard

Linda Cai, Clayton Thomas, Matt Weinberg
Princeton University
July 12, 2023
Model: $n$ bidders, $m$ items.

- Each bidder $i$ has valuation function $v_i: 2^m \rightarrow \mathbb{R}^+$.
- Bidders participate in some (possibly interactive) protocol.
- Auctioneer awards items $S_i$ to bidder $i$, charges price $p_i$.

Goal: Maximizes welfare $= \sum_i v_i(S_i)$.

$\alpha$-approximation: guarantees $\sum_i v_i(S_i) \geq \alpha \cdot \text{OPT}$.

Question: What welfare can a mechanism guarantee when agents are self-interested and strategic?

A mechanism is truthful if for all $v_1(\cdot), ..., v_n(\cdot)$, it is in a bidder's interest to be truthful regardless of what others do.
Combinatorial Auctions

**Model:** $n$ bidders, $m$ items.
- Each bidder $i$ has valuation function $v_i : 2^m \rightarrow R^+$.
- Bidders participate in some (possibly interactive) protocol.
- Auctioneer awards items $S_i$ to bidder $i$, charges price $p_i$.

**Goal:** Maximizes welfare $= \sum_i v_i(S_i)$.
- $\alpha$-approximation: guarantees $\sum_i v_i(S_i) \geq \alpha OPT$.

Question: What welfare can a mechanism guarantee when agents are self-interested and strategic?

- A mechanism is **truthful** if for all $v_1(\cdot) \ldots v_n(\cdot)$, it is in a bidder's interest to be truthful regardless of what others do.
Combinatorial Auctions

Model: $n$ bidders, $m$ items.

- Each bidder $i$ has valuation function $v_i : 2^m \rightarrow R^+$.
- Bidders participate in some (possibly interactive) protocol.
- Auctioneer awards items $S_i$ to bidder $i$, charges price $p_i$.

Goal: Maximizes welfare $= \sum_i v_i(S_i)$.

- $\alpha$-approximation: guarantees $\sum_i v_i(S_i) \geq \alpha \cdot OPT$.

Question: What welfare can a mechanism guarantee when agents are self-interested and strategic?

- A mechanism is truthful if for all $v_1(\cdot) \ldots v_n(\cdot)$, it is in a bidder’s interest to be truthful regardless of what others do.
Combinatorial Auctions

Constraints on Mechanisms:

- Computationally-efficient: auctioneer and bidders can only compute functions in $\mathbb{P}$
- Communication-efficient: auctioneer and bidders can only communicate $\text{poly}(m, n)$ bits
Combinatorial Auctions

Constraints on Mechanisms:

- Computationally-efficient: auctioneer and bidders can only compute functions in $\mathbb{P}$
- Communication-efficient: auctioneer and bidders can only communicate $\text{poly}(m, n)$ bits

Constraints on valuation functions: submodular $\subset$ XOS

- submodular: for all sets $X, Y$, $\nu(X \cup Y) + \nu(X \cap Y) \leq \nu(X) + \nu(Y)$
- XOS (fractionally subadditive): let $L$ be a set of additive functions. Then $\forall S \subset [m], \nu(S) = \max_{\nu_l \in L} \nu_l(S)$.

<table>
<thead>
<tr>
<th></th>
<th>submodular</th>
<th>XOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computation</td>
<td>$\Omega(m^{1/2-\epsilon})$ [DV16]</td>
<td>$\Omega(m^{1/2-\epsilon})$ [DV16]</td>
</tr>
<tr>
<td>Communication</td>
<td>$O((\log \log m)^3)$ [AS19]</td>
<td>$O((\log \log m)^3)$ [AS19]</td>
</tr>
</tbody>
</table>
Combinatorial Auctions

Constraints on Mechanisms:

• Computationally-efficient: auctioneer and bidders can only compute functions in \( P \)
• Communication-efficient: auctioneer and bidders can only communicate \( \text{poly}(m, n) \) bits

Constraints on valuation functions: submodular \( \subset \) XOS

• submodular: for all sets \( X, Y \), \( v(X \cup Y) + v(X \cap Y) \leq v(X) + v(Y) \)
• XOS (fractionally subadditive): let \( L \) be a set of additive functions. Then \( \forall S \subset [m], v(S) = \max_{v_l \in L} v_l(S) \).

<table>
<thead>
<tr>
<th></th>
<th>submodular</th>
<th>XOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computation</td>
<td>( \Omega(m^{1/2-\epsilon}) ) [DV16]</td>
<td>( \Omega(m^{1/2-\epsilon}) ) [DV16]</td>
</tr>
<tr>
<td>Communication</td>
<td>( O((\log \log m)^3) ) [AS19]</td>
<td>( O((\log \log m)^3) ) [AS19]</td>
</tr>
</tbody>
</table>

Why is there a separation between computationally-efficient and communication-efficient truthful mechanisms?
Motivation

XOS Bidder Combinatorial Auctions

- $n$ buyer, $m$ items, bidder valuation functions are XOS
- Goal: maximize welfare
Motivation

XOS Bidder Combinatorial Auctions

- \( n \) buyer, \( m \) items, bidder valuation functions are XOS
- Goal: maximize welfare

Communication model:

- State of the art truthful mechanism “Price Learning Mechanism”[AS19] is at its core a posted price mechanism:
  - visits bidders one at a time, posts a price \( p_j \) on each remaining item \( j \)
  - offers the option to purchase any set \( S \) of items, here bidders pick set that maximize utility (called demand query, NP-hard to compute)
Motivation

**XOS Bidder Combinatorial Auctions**

- $n$ buyer, $m$ items, bidder valuation functions are XOS
- Goal: maximize welfare

**Communication model:**
- State of the art truthful mechanism “Price Learning Mechanism” [AS19] is at its core a posted price mechanism:
  - visits bidders one at a time, posts a price $p_j$ on each remaining item $j$
  - offers the option to purchase any set $S$ of items, here bidders pick set that maximize utility (called demand query, NP-hard to compute)

**Computation model:**
- NP-hard for truthful mechanisms to achieve a $m^{1/2-\epsilon}$-approximation for any $\epsilon > 0$ [DV16]
- $\sqrt{m}$-approximation algorithm is tight [DNS10]
Motivation

Submodular Bidder Combinatorial Auctions

- $n$ buyer, $m$ items, bidder valuation functions are submodular
- Goal: maximize welfare

Communication model:

- State of the art truthful mechanism “Price Learning Mechanism” [AS19] is at its core a posted price mechanism:
  - visits bidders one at a time, posts a price $p_j$ on each remaining item $j$
  - offers the option to purchase any set $S$ of items, here bidders pick set that maximize utility (called demand query, NP-hard to compute)

Computation model:

- NP-hard for truthful mechanisms to achieve a $m^{1/2-\epsilon}$-approximation for any $\epsilon > 0$ [DV16]
- exists $e/(e - 1)$-approximation algorithm [Von08]
Motivation

Simpler example: one buyer combinatorial public project

- 1 buyer, $m$ items
- the buyer can only receive $k$ out of the $m$ items
- Goal: maximize welfare
Simpler example: one buyer combinatorial public project

- 1 buyer, \( m \) items
- the buyer can only receive \( k \) out of the \( m \) items
- Goal: maximize welfare

Communication model:

- Truthful mechanism “Set-For-Free”: let bidder pick any \( k \)-set they like achieves optimal welfare
Motivation

Simpler example: one buyer combinatorial public project

• 1 buyer, \( m \) items
• the buyer can only receive \( k \) out of the \( m \) items
• Goal: maximize welfare

Communication model:

• Truthful mechanism “Set-For-Free”: let bidder pick any \( k \)-set they like achieves optimal welfare

Computation model:

• NP-hard for truthful mechanisms to achieve a \( m^{1/2-\epsilon} \)-approximation for any \( \epsilon > 0 \) [SS08]
• Exists poly-time \( e/(e-1) \)-approximation algorithm [NWF78]
A Different Solution Concept

Advice

- Takes input valuation $v_i(\cdot)$ of $i$ and tentative strategy $s(\cdot)$, outputs advised strategy $A^{v_i,s}(\cdot)$ which is either $s(\cdot)$ or one that dominates it.
- Advice is idempotent (applying advice twice is the same as applying advice once).

We say that $s(\cdot)$ is advised for $v_i(\cdot)$ under $A$ if $A^{v_i,s}(\cdot) = s(\cdot)$. A bidder with valuation $v_i(\cdot)$ follows advice $A$ if they use a strategy which is advised under $A$. 
Advice

• Takes input valuation $v_i(\cdot)$ of $i$ and tentative strategy $s(\cdot)$, outputs advised strategy $A^{v_i,s}(\cdot)$ which is either $s(\cdot)$ or one that dominates it

• Advice is idempotent (applying advice twice is the same as applying advice once)

We say that $s(\cdot)$ is advised for $v_i(\cdot)$ under $A$ if $A^{v_i,s}(\cdot) = s(\cdot)$. A bidder with valuation $v_i(\cdot)$ follows advice $A$ if they use a strategy which is advised under $A$.

Implementation in Advised Strategy

A poly time mechanism guarantees an $\alpha$-approximation in implementation in advised strategies if there exists poly-time advice for each player such that an $\alpha$-approximation is achieved whenever all players follow advice.

• *Equivalent to ”Algorithmic Implementation” in [BLP09].
A Different Solution Concept

**example: one buyer combinatorial public project**

- 1 buyer, \( m \) items
- the buyer can only receive \( k \) out of the \( m \) items
- Goal: maximize welfare
A Different Solution Concept

example: one buyer combinatorial public project

- 1 buyer, $m$ items
- the buyer can only receive $k$ out of the $m$ items
- Goal: maximize welfare

Consider “Set-For-Free” (player pick any $k$-set) with advice $A$ that
- takes input $v(\cdot)$ and set $S$
- Runs $e/(e-1)$-approximation algorithm to get set $T$. Returns $\text{argmax}\{v(S), v(T)\}$
A Different Solution Concept

example: one buyer combinatorial public project

- 1 buyer, $m$ items
- the buyer can only receive $k$ out of the $m$ items
- Goal: maximize welfare

Consider “Set-For-Free” (player pick any $k$-set) with advice $A$ that

- takes input $v(\cdot)$ and set $S$
- Runs $e/(e-1)$-approximation algorithm to get set $T$. Returns $\text{argmax}\{v(S), v(T)\}$

“Set-For-Free” guarantees an $e/(e-1)$-approximation in implementation in advised strategy with advice $A$. 
Main Result

Can “Price Learning Mechanism” be modified into a poly-time mechanism in implementation in advised strategy?

Theorem 1

There exists a poly-time mechanism for submodular welfare maximization guaranteeing $O((\log \log m)^3)$-approximation in implementation in advised strategies with polynomial time computable advice.
Main Result

Can “Price Learning Mechanism” be modified into a poly-time mechanism in implementation in advised strategy?

Theorem 1

There exists a poly-time mechanism for submodular welfare maximization guaranteeing $O((\log \log m)^3)$-approximation in implementation in advised strategies with polynomial time computable advice.

Mechanism Construction Outline:

- Find some notion of approximate demand query for submodular bidders
- Use “Price Learning Algorithm” with approximate demand query as advice
For any $c, d \leq 1$, a $c$-approximate demand oracle takes as input a valuation function $v(\cdot)$ and a price vector $p$ and outputs a set of items $S$ such that

$$v(S) - p(S) \geq c \cdot \max_T \{v(T) - p(T)\}.$$ 

[FJ14] It is NP-hard to design a $m^{1-\epsilon}$-approximate demand oracle when $v(\cdot)$ is submodular.
For any $c, d \leq 1$, a $(c, d)$-approximate demand oracle takes as input a valuation function $v(\cdot)$ and a price vector $p$ and outputs a set of items $S$ such that

$$v(S) - p(S) \geq c \cdot \max_T \{v(T) - p(T)/d\}.$$
Advice: Approximate Demand Oracle

(c, d)-Approximate Demand Oracle

For any \( c, d \leq 1 \), a \((c, d)\)-approximate demand oracle takes as input a valuation function \( v(\cdot) \) and a price vector \( p \) and outputs a set of items \( S \) such that

\[
v(S) - p(S) \geq c \cdot \max_T \{v(T) - p(T)/d\}.
\]

Theorem 2

Let \( \mathcal{V} \) be a subclass of XOS valuations and let \( D \) be a poly-time \((c, d)\)-approximate demand oracle for valuation class \( \mathcal{V} \). Then there exists a poly-time mechanism for welfare maximization when all valuations are in \( \mathcal{V} \) with approximation guarantee

\[
O \left( \max \left\{ \frac{1}{c}, \frac{1}{d} \right\} \cdot (\log \log m)^3 \right)
\]

in implementation in advised strategies with polynomial time computable advice.
Theorem 2

Let \( \mathcal{V} \) be a subclass of XOS valuations and let \( D \) be a poly-time \((c, d)\)-approximate demand oracle for valuation class \( \mathcal{V} \). Then there exists a poly-time mechanism for welfare maximization when all valuations are in \( \mathcal{V} \) with approximation guarantee \( O \left( \max \left\{ \frac{1}{c}, \frac{1}{d} \right\} \cdot (\log \log m)^3 \right) \) in implementation in advised strategies with polynomial time computable advice.

- When \( \mathcal{V} \) is submodular, exists \((\frac{1}{2}, \frac{1}{2})\)-approximate demand oracle
Advice: Approximate Demand Oracle

Theorem 2
Let $\mathcal{V}$ be a subclass of XOS valuations and let $D$ be a poly-time $(c, d)$-approximate demand oracle for valuation class $\mathcal{V}$. Then there exists a poly-time mechanism for welfare maximization when all valuations are in $\mathcal{V}$ with approximation guarantee $O\left(\max\left\{ \frac{1}{c}, \frac{1}{d} \right\} \cdot (\log \log m)^3 \right)$ in implementation in advised strategies with polynomial time computable advice.

- When $\mathcal{V}$ is submodular, exists $(\frac{1}{2}, \frac{1}{2})$-approximate demand oracle

Algorithm 2 SimpleGreedy$(v, p, M)$

$$S \leftarrow \emptyset$$
for $j = 1, \ldots, m$ :
    if $v(S \cup \{j\}) - v(S) \geq 2p(j)$ :
        $S \leftarrow S \cup \{j\}$
return $S$
We use the solution concept implementation in advised strategies to show that “Price Learning Mechanism” for submodular welfare maximization maintains its approximation guarantee when buyers follow advice recommended by a (1/2, 1/2)-approximate demand oracle.

- “Implementation in advised strategies” equivalent to “algorithmic implementation” [BLP09], first application since introduction.
- more application out there?
Thank you for listening!