Algorithmic Mechanism Design

Knapsack Auctions

Joseph Chuang-Chieh Lin

Department of Computer Science & Engineering,
National Taiwan Ocean University

Fall 2024

Outline

Outline

Whenever there is a shared resource with *limited* capacity, you have a knapsack problem.

Definition

- We study about another example of single-parameter environments.

Knapsack Auctions

- Each bidder i has a publicly known size w_i and a private valuation.
- The seller has a capacity W.
- The feasible set X is defined as the 0-1 vectors $(x_1, \ldots, x_n)$ such that $\sum_{i=1}^{n} w_i x_i \leq W$.
  - $x_i = 1$: i is a winning bidder.

Explanations

- Each bidder's size could represent
  - the duration of a company's television ad;
  - the valuation its willingness-to-pay for its ad being shown;
  - the seller capacity the length of a commercial break.
- The situation that bidders who want
  - files stored on a *shared server*,
  - data streams sent through a shared communication channel
  - processes to be executed on a *shared* supercomputer.

  ⋮

Assumptions

- We receive truthful bids and decide on our allocation rule.
- **Goal:** Devise a payment rule that extends the allocation rule to a DSIC mechanism.

Outline

1. Knapsack Auctions
   - Welfare-Maximizing DSIC Knapsack Auctions
   - Critical Bids
   - Intractability of Welfare Maximization
2. Algorithmic Mechanism Design
   - The Best-Case Scenario: DSIC for Free
   - Knapsack Auctions Revisited
3. The Revelation Principle
   - Justifying Direct Revelation
   - Beyond Dominant-Strategy Equilibria

To maximize the welfare:

$$\mathbf{x}(\mathbf{b}) = \arg\max_{X} \sum_{i=1}^{n} b_i x_i .$$

The goal is to compute the subset of items of maximum total value that has total size bounded by W.

- It's maximum by the assumption that bidders bid truthfully.

To maximize the welfare:

$$\mathbf{x}(\mathbf{b}) = \arg\max_{X} \sum_{i=1}^{n} b_i x_i .$$

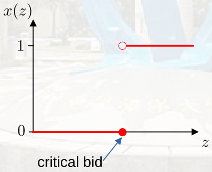
The goal is to compute the subset of items of maximum total value that has total size bounded by W.

- It's maximum by the assumption that bidders bid truthfully.
- ⋆ Check that the allocation rule $\mathbf{x}(\cdot)$ is monotone.
  - Bidding higher can only get her more stuff.

Outline

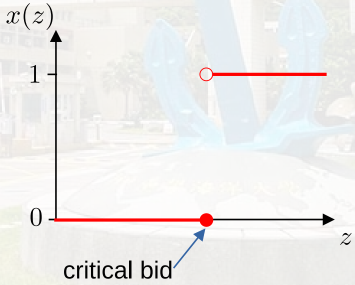
The Guarantee from Myerson's Lemma

- Myerson's lemma guarantees the existence of a payment rule $\mathbf{p}$ such that the mechanism $(\mathbf{x}, \mathbf{p})$ is DSIC.
- Since the allocation rule is monotone and assigns 0 or 1 to each bidder, the allocation curve $x_i(\cdot, \mathbf{b}_{-i})$ is 0 until some "breakpoint" z.
    - At z, the allocation jumps to 1.



The Guarantee from Myerson's Lemma (contd.)

- If i bids less than z, she loses and pays 0.
- If i bids more than z, she pays $\geq z \cdot (1 - 0) = z$.
  - z is the infimum bid that she could make and continue to win (holding $\mathbf{b}_{-i}$ fixed).



Outline

1. Knapsack Auctions
   - Welfare-Maximizing DSIC Knapsack Auctions
   - Critical Bids
   - Intractability of Welfare Maximization
2. Algorithmic Mechanism Design
   - The Best-Case Scenario: DSIC for Free
   - Knapsack Auctions Revisited
3. The Revelation Principle
   - Justifying Direct Revelation
   - Beyond Dominant-Strategy Equilibria

(Recall) An ideal mechanism

Properties of an Ideal Mechanism

- DSIC
- welfare maximizing (assuming truthful bids).
- runs in polynomial time in the input size (e.g., bids, sizes, the capacity).

Is our mechanism for the knapsack auction ideal?

$$\mathbf{x}(\mathbf{b}) = \arg\max_{X} \sum_{i=1}^{n} b_i x_i .$$

Is our mechanism for the knapsack auction ideal?

$$\mathbf{x}(\mathbf{b}) = \arg\max_{X} \sum_{i=1}^{n} b_i x_i .$$

The answer: NO.

Is our mechanism for the knapsack auction ideal?

$$\mathbf{x}(\mathbf{b}) = \arg\max_{X} \sum_{i=1}^{n} b_i x_i .$$

The answer: NO.

- The knapsack problem is a notorious **NP**-hard problem.

Is our mechanism for the knapsack auction ideal?

$$\mathbf{x}(\mathbf{b}) = \arg\max_{X} \sum_{i=1}^{n} b_i x_i.$$

The answer: NO.

- The knapsack problem is a notorious **NP**-hard problem.
  - No polynomial time implementation of the allocation rule unless $\mathbf{NP} = \mathbf{P}$.

Is our mechanism for the knapsack auction ideal?

$$\mathbf{x}(\mathbf{b}) = \arg\max_{X} \sum_{i=1}^{n} b_i x_i.$$

The answer: NO.

- The knapsack problem is a notorious **NP**-hard problem.
  - No polynomial time implementation of the allocation rule unless $\mathbf{NP} = \mathbf{P}$.
- Hence, we would like to consider relaxing at least one of the three goals.

An ideal mechanism

Properties of an Ideal Mechanism

- DSIC
- welfare maximizing (assuming truthful bids).
- runs in polynomial time in the input size (e.g., bids, sizes, the capacity).

- Relax the second requirement as little as possible.
- Design a polynomial time and monotone allocation rule that comes as close as possible to the maximum possible social welfare.

Outline

Outline

Approximation algorithms come to rescue?

- The primary goal in approximation algorithms is to design polynomial-time algorithms for **NP**-hard optimization problems that are as close to the optimal solution as possible.

Approximation algorithms come to rescue?

- The primary goal in approximation algorithms is to design polynomial-time algorithms for **NP**-hard optimization problems that are as close to the optimal solution as possible.
- Algorithmic mechanism design has exactly the same goal, except that the algorithms must additionally obey a monotonicity constraint.

Approximation algorithms come to rescue?

- The primary goal in approximation algorithms is to design polynomial-time algorithms for **NP**-hard optimization problems that are as close to the optimal solution as possible.
- Algorithmic mechanism design has exactly the same goal, except that the algorithms must additionally obey a monotonicity constraint.
- The incentive constraints of the mechanism design goal are neatly compiled into a relatively intuitive extra constraint on the allocation rule.

Approximation Algorithms come to rescue? (contd.)

- The design space of polynomial-time DSIC mechanisms is only smaller than that of polynomial-time approximation algorithms.

Approximation Algorithms come to rescue? (contd.)

- The design space of polynomial-time DSIC mechanisms is only smaller than that of polynomial-time approximation algorithms.
- (Imagine) The best-case scenario: DSIC constraint causes no additional welfare loss (beyond the loss from the polynomial-time requirement).

Approximation Algorithms come to rescue? (contd.)

- The design space of polynomial-time DSIC mechanisms is only smaller than that of polynomial-time approximation algorithms.
- (Imagine) The best-case scenario: DSIC constraint causes no additional welfare loss (beyond the loss from the polynomial-time requirement).
- Exact welfare maximization automatically yields a monotone allocation rule.

Approximation Algorithms come to rescue? (contd.)

- The design space of polynomial-time DSIC mechanisms is only smaller than that of polynomial-time approximation algorithms.
- (Imagine) The best-case scenario: DSIC constraint causes no additional welfare loss (beyond the loss from the polynomial-time requirement).
- Exact welfare maximization automatically yields a monotone allocation rule.
- Is that true for *approximate* welfare maximization?

Outline

Greedy approach

- Say S be a set of winners with total size $\sum_{i \in S} w_i \leq W$.
- We choose such a set S via a simple greedy algorithm.

⋆ We can assume that $w_i \leq W$ for all i (why?)

A Greedy Knapsack Heuristic

A Greedy Algorithm

1. Sort and re-index the bidders so that
$$\frac{b_1}{w_1} \geq \frac{b_2}{w_2} \geq \cdots \geq \frac{b_n}{w_n}.$$
2. Pick winners in this order until one doesn't fit, and then halt.
3. Return either the solution from Step 2 or the highest bidder: $\arg\max_i b_i$, whichever has larger social welfare.

Theorem (Knapsack Approximation Guarantee)

Assuming truthful bids, the social welfare achieved by the greedy allocation is at least half of the maximum social welfare.

Sketch of proving the theorem

- To have an upper bound on the maximum social welfare, allow bidders to be chosen fractionally, with the value prorated accordingly.
  - E.g., if 70% of a bidder with value 10 is chosen, then it contributes 7 to the welfare.
- Sort the bidders according to the step above, and pick winners in this order until the the capacity W is fully exhausted.
  - You can verify that this maximizes the welfare over all feasible solutions.

Sketch of proving the theorem (contd.)

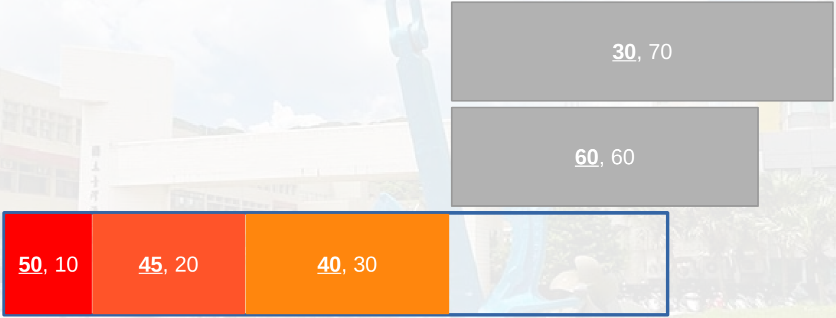
- In the optimal *fractional* solution, suppose that the first k bidders in the sorted order win and that the $(k+1)$th bidder *fractionally* wins.

⋆ The welfare achieved by steps ① and ② in the greedy allocation rule $=$ the total value of the first k bidders.

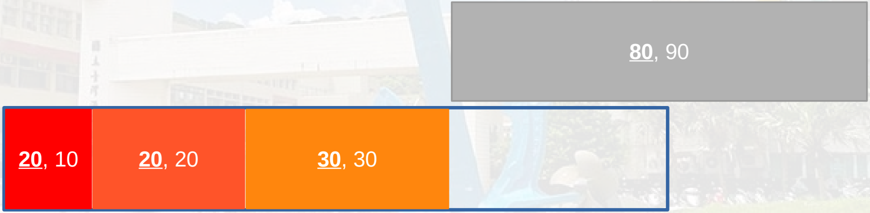
⋆ The welfare consisting only the highest bidder $\geq$ the fractional value of the $(k+1)$th bidder.

- The better of these two solutions $\geq \frac{1}{2}\times$ the welfare of the optimal fractional solution.

  $\Rightarrow$ Exercise!


30, 70
60, 60
50, 10
45, 20
40, 30


20, 50
30, 45
60, 60


80, 90
20, 10
20, 20
30, 30

Sum up

- The greedy allocation rule is monotone (check by yourself).
- Using Myerson's lemma, we can extend it to a DSIC mechanism that runs in polynomial time and, assuming truthful bids, achieves social welfare at least 50% of the maximum possible.

Outline

Reiteration

- There are good reasons to strive for a DSIC guarantee.
  - Easy for a participant to figure out what to do in a DSIC mechanism.
  - The designer can predict the mechanism's outcome.

Outline

The DSIC Condition

The DSIC Condition

(a) For every valuation profile, the mechanism has a dominant-strategy equilibrium.

⋆ An outcome that results from every participant playing a *dominant strategy*.

(b) In this dominant-strategy equilibrium, every participant truthfully reports her private information to the mechanism.

- The revelation principle asserts that:

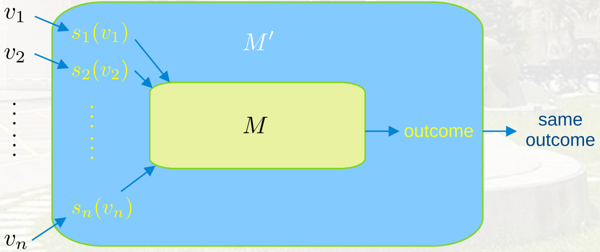
given (a), then (b) comes for free!

The Revelation Principle

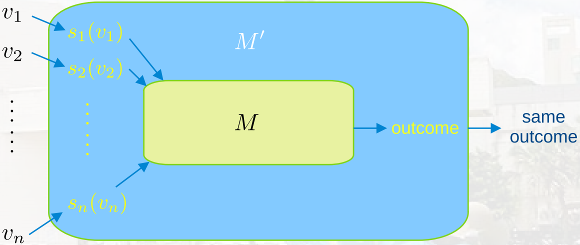
Theorem (Revelation Principle for DSIC Mechanisms)

For every mechanism M where every participant always has a dominant strategy, there is an equivalent direct-revelation DSIC mechanism M'.

- We use a simulation argument to construct M' as follows.

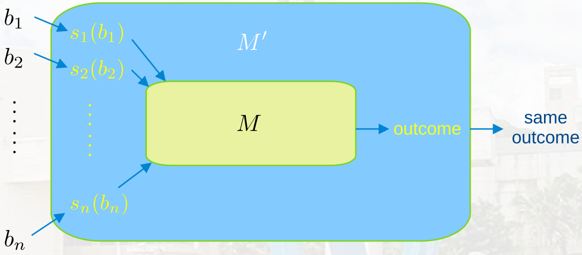


Proof



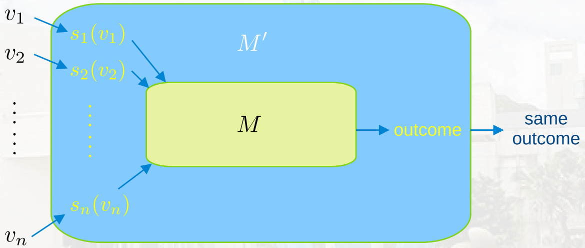
- For every participant i and its private information v_i, she has a dominant strategy $s_i(v_i)$ in mechanism M (by assumption).

Proof



- Construct M', such that participants delegate the responsibility of playing the appropriate dominant strategy to M'.
  - M' accepts bids $b_1, \ldots, b_n$.
  - Then M', which is of direct-revelation, submits the bids $s_1(b_1), \ldots, s_n(b_n)$ to the mechanism M and choose the same outcome that M does.

Proof



- Mechanism M' is DSIC:
  - If a participant i has private information v_i, then submitting a bid other than v_i can only result in M' playing a strategy other than $s_i(v_i)$ in M, which can only decrease i's utility.

What we have learned from the theorem?

- Truthfulness per se is not important.
- The difficult part is the requirement to have a dominant-strategy equilibrium.

Outline

Heads up

- DSIC and non-DSIC mechanisms are *incomparable*.
  - The former enjoys stronger incentive guarantees
  - The latter may enjoy better performance guarantees.