CS243: Introduction to Algorithmic Game Theory

Distributed Mechanism Design (Dengji ZHAO)

SIST, ShanghaiTech University, China

Recap: centralized mechanism

Famous and traditional mechanisms are all centralized: VCG, Myerson, IDM (auction), TTC (matching).....

- Usually assume the existence of a trusted center.
- All information are reported to the center.
- The center performs all computation required by the mechanism, decides and executes allocations.

Especially, the center will know the global structure of the social network.

Question

What if there is no central authority?

Why we need distributed mechanisms?

This approach requires transmitting all the relevant information to a single, trusted entity, which is feasible if:

- such a trusted center exists, and
- the communication required to transmit the information, and
- the resulting computational burden on the trusted center are both manageable.

However, if either of these assumptions fails, then a more decentralized approach must be considered.

What is a distributed mechanism?

We do not want a central node or a third party, this leads to Distributed Algorithmic Mechanism Design (DAMD):

- Participants do not need to submit their types to a center or a third party.
- The computational tasks are assigned to participants.
- Allocations are decided and executed by these participants.
- Every agent can perform her actions freely according to her own benefits.

Difference between DAMD and AMD

DAMD and AMD differ in two important respects:
The first difference involves the nature of complexity.
Any measure of the complexity of a distributed algorithm executed over an interconnection network T must consider at least five quantities:

1. the total number of messages sent over T,
2. the maximum number of messages sent over any one link in T,
3. the maximum size of a message, the local computational burden at each node,
4. the local computational burden at each node,
5. the storage required at each node.

Difference between DAMD and AMD

The second difference is the strategic nature of the computation itself.
In DAMD, they have more opportunities to manipulate the outcome, e.g,

1. by misrepresenting the results of a local computation to a neighboring agent, or,
2. by simply not communicating with that neighboring agent at all, in order to exclude him from the game.

The assumption of selfishness requires to consider all forms of **manipulative behavior** when designing the mechanism.
Must provide incentives that ensure selfish agents find it in their best interest to perform the distributed computation correctly.

Recap: VCG

Definition 9.16 A mechanism $(f, p_1, \ldots, p_n)$ is called a Vickrey–Clarke–Groves (VCG) mechanism if

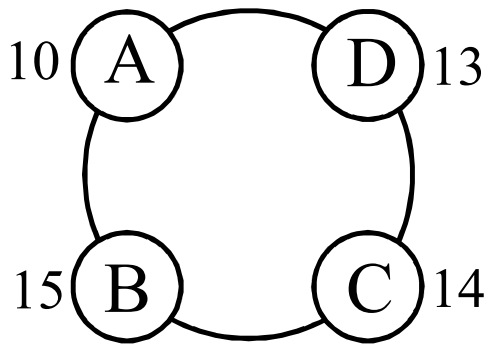
- $f(v_1, \ldots, v_n) \in \text{argmax}_{a \in A} \sum_i v_i(a)$; that is, f maximizes the social welfare, and
- for some functions $h_1, \ldots, h_n$, where $h_i : V_{-i} \to \Re$ (i.e., h_i does not depend on v_i), we have that for all $v_1 \in V_1, \ldots, v_n \in V_n$: $p_i(v_1, \ldots, v_n) = h_i(v_{-i}) - \sum_{j \neq i} v_j(f(v_1, \ldots, v_n))$.

  - Definition of $h_{-i} : V_{-i} \to \mathbb{R}$
    - $h_{-i}(.) = 0$
    - $h_{-i}(v_{-i}) = \sum_{j \in N \setminus \{i\}} v_j(f(v_{-i}))$, the maximum social welfare without i's participation.
    - ...

Distributed Implementation of VCG

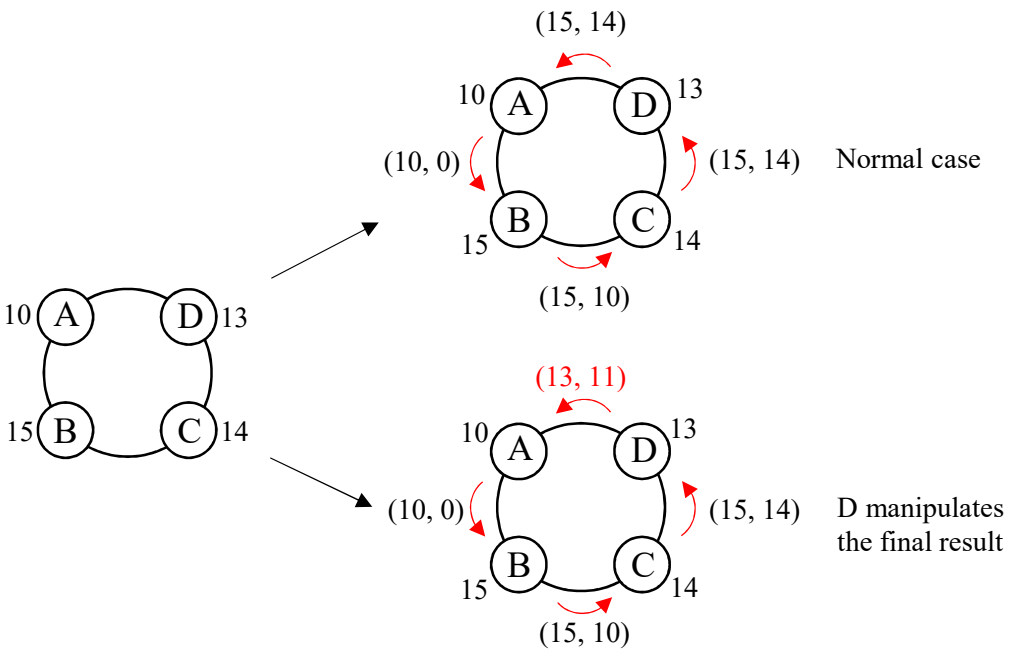
Now assume there is no trusted center, i.e., the computation of the VCG must be done by the agents themselves.
An example to see how a distributed computation can be manipulated:

- Consider the agents are connected in a ring.



Distributed Implementation of VCG

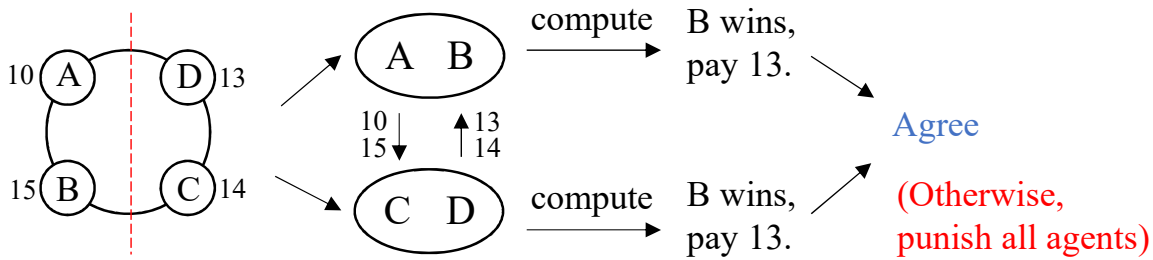
- Assume the agents are computing a second-price auction by passing around a message containing the top two bids.
- An agent can put her bid on top and put in a very low bid for the second bid, then she can get the item more cheaply. (as long as these bids are not overwritten by later agents.)



Distributed Implementation of VCG

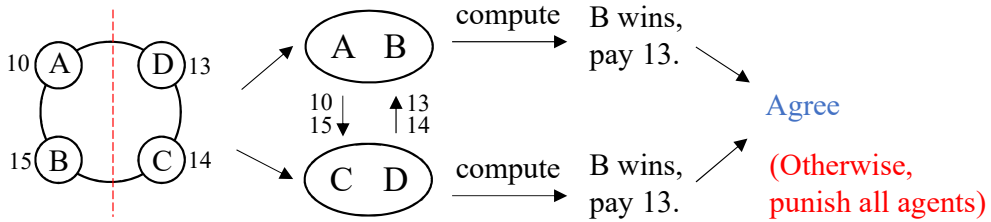
One way to avoid this is *replication*:

- Break the agents into two groups, have them exchange all their valuations, and then have each group compute its own version of f and the p_i.



Distributed Implementation of VCG

- If the two groups agree on the outcomes and payments, then those outcomes and payments are adopted;
- if not, all agents suffer a severe penalty.



Note: the method still presume the existence of some central *enforcer* whose responsibility is to implement the outcome and collect the payments, and conduct the punishment.

Distributed Implementation of VCG

An agent plays different roles in the two versions of the computation:

- In the first, his role is to help compute the outcome and payments;
- an agent could engage in arbitrary computational manipulation to obtain a more favorable p_i or choose an outcome he prefers to the socially optimal one.
- In the other, his role is to provide his valuation so that others may perform this computation.
- all he could do is lie about v_i.

Distributed Implementation of VCG

Notice that faithful computation is not a dominant strategy now. For example:

- If all the other agents decide to choose a suboptimal outcome, then agent *i* is better off going along with that choice rather than causing a disagreement.
- If all the other agents faithfully execute the prescribed algorithm, then agent *i* is best off doing so as well.

Thus, the most natural solution concept when considering computational manipulation is not dominant strategies but instead **ex-post Nash equilibrium**.

Recap: Mechanism Design in Social Networks (IDM)

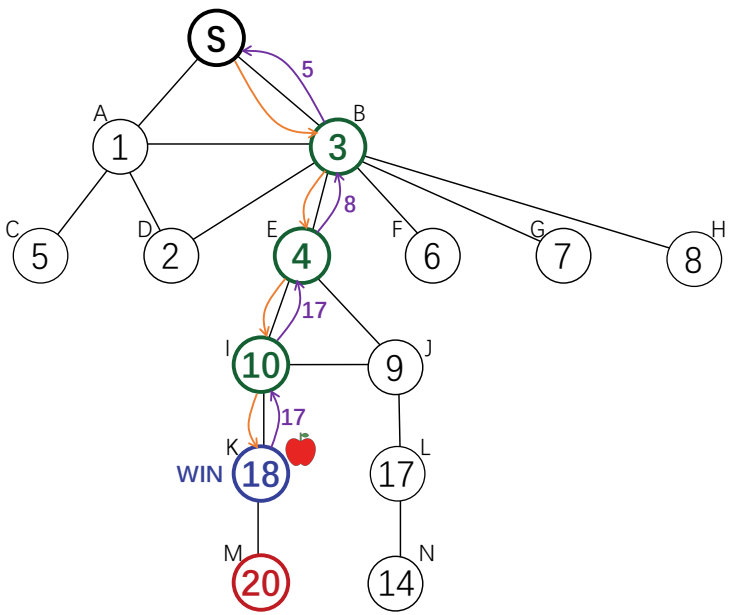
The allocation definition:

- Identify the node i with the highest bid and the node's diffusion critical node path $P_{c_i} = (c_i^1, c_i^2, ..., i)$.
- Give the item to the first node of P_{c_i}, the node pays to the seller and then decides to whether keep the item or pass it to the next node in P_{c_i}:
  - If the payment of the next node is greater than the bid of the current node, passes it to the next node and receives the payment from the next node; the next node makes a similar decision;
  - otherwise, keep the item.

Information Diffusion Mechanism

The outcome of the Information Diffusion Mechanism:

- the item is allocated to node **K**.
- node **K** pays 17 to **I**, **I** pays 17 to **E**, **E** pays 8 to **B**, **B** pays 5 to the **seller**.
- the utilities of **K**, **I**, **E**, **B**, **seller** are $1, 0, 9, 3, 5$.



From centralized mechanism to distributed mechanism

- In centralized settings, agents need to report her type to the center (the only action space for manipulation). In distributed settings, the execution is assigned to agents.
- Thus, agents have a larger action space and more possibilities to manipulate, so we need to guarantee the participants execute the mechanism correctly.
- Use a new concept, **strategy** s_i, to capture how agent i behaves in all states of the mechanism. And let $\Sigma = (\Sigma_1, ..., \Sigma_n)$ be the strategy space.

Distributed Mechanism Design in Social Networks

Distributed Mechanism

A distributed mechanism d^M is a tuple $d^M = (\Sigma, (\pi, p), s^M)$, where $s^M = (s_1^M, ..., s_n^M) \in \Sigma$ is the intended strategy of the mechanism.

The intended strategy can be considered as a series of algorithms or actions that the mechanism intends i to perform.

- Note: In the centralized scenario, the strategy space can be reduced to type space, i.e., $\Sigma_i = \Theta_i$.

Distributed Mechanism Design in Social Networks

An agent's strategy can be decomposed into three parts:

- Information-revelation action: t_i
- Message-passing action: q_i
- Computational action: f_i

For distributed mechanisms, we also design an agent's intended strategy:

$$s_i^M = (t_i^M, q_i^M, f_i^M)$$

Another property: Ex-post IC

Ex-post incentive compatible:

$$u_i(\theta_i, (s_i^M, s_{-i}^M), (\pi, p)) \geq u_i(\theta_i, (s_i, s_{-i}^M), (\pi, p))$$

- It means no one can obtain a higher utility by deviating from the equilibrium that everyone executes the intended strategy.
- Ex-post IC $\leftrightarrow$ s^M is an ex-post Nash equilibrium.

Question

Why we only want ex-post IC in distributed mechanisms?

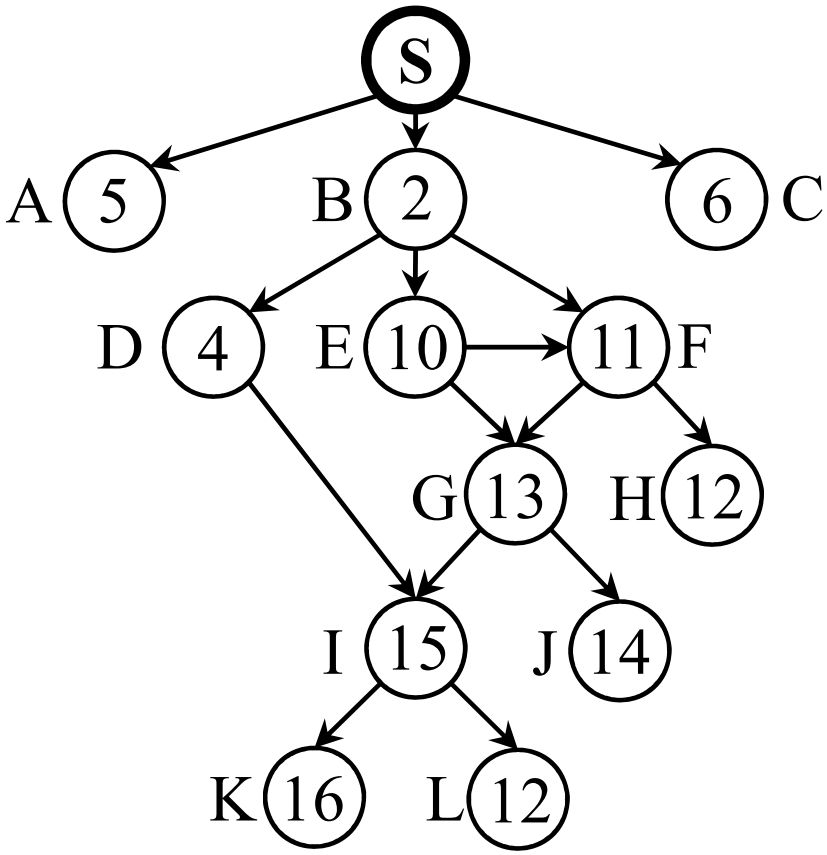
Sequential Resale Auction

Model the distributed auction as a three-stage process:

1. In the first stage, the buyers diffuse the sale information to their neighbors.
2. In the second stage, the buyers collect their invited neighbors' bids and represent them to join the sale.
3. In the third stage, do sequential resales from the seller to the final winner.

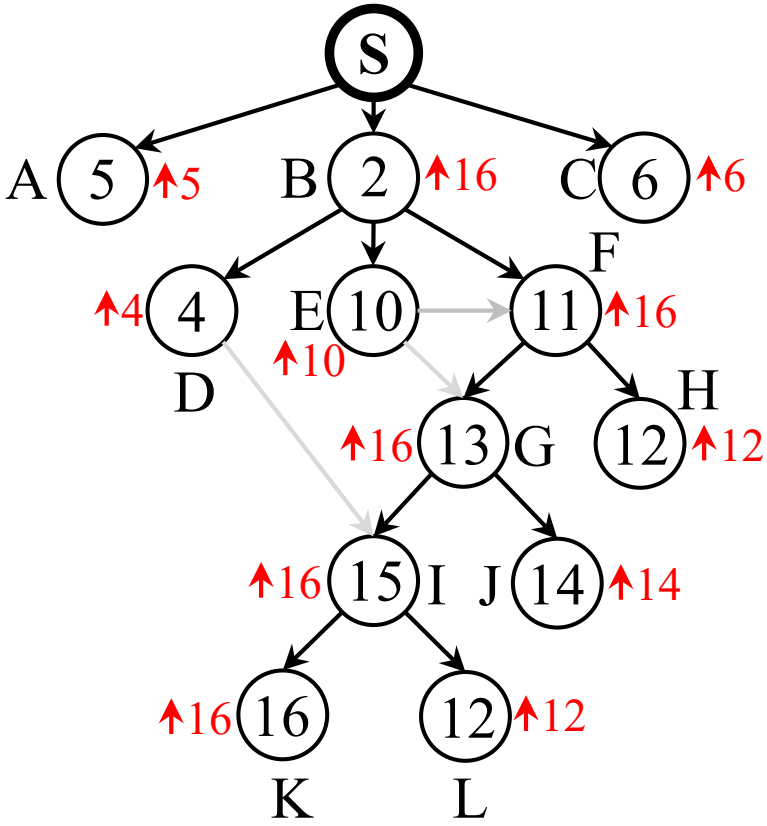
Example of Sequential Resale Auction

Assume the graph after the first stage is shown as below:



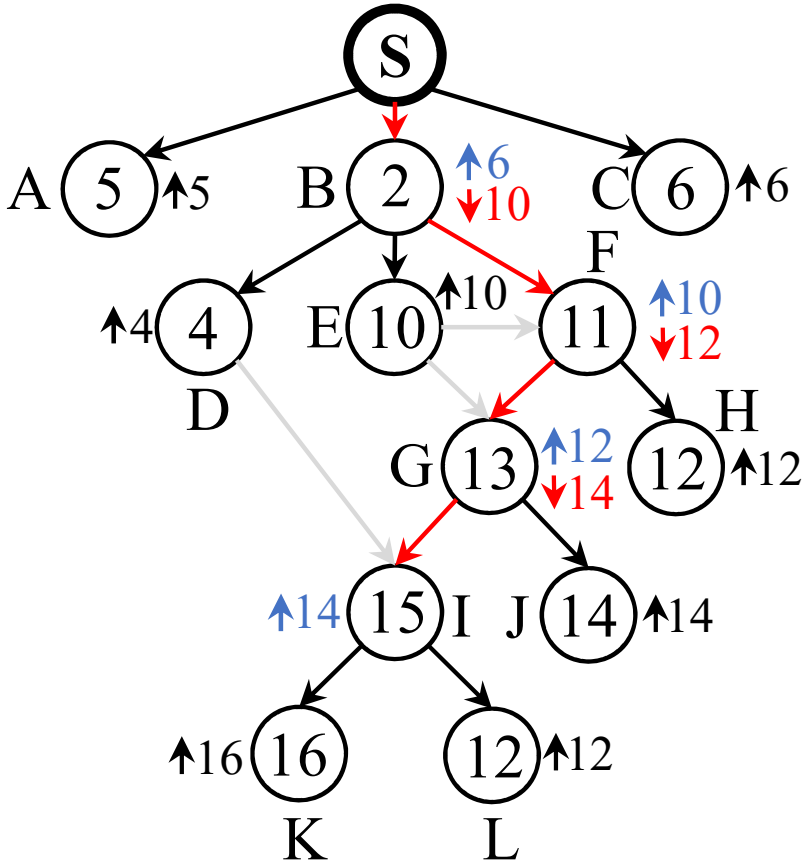
Example of Sequential Resale Auction

The process of bottom-up aggregation is shown as below:



Example of Sequential Resale Auction

The sequential resale in the third stage is shown as below:



Sequential Resale Auction

Stage 1 - Top-down diffusion stage
The sale information is spread in the social network starting from the original seller S.

Information-revelation Action

A buyer's information-revelation action t_i is to decide her bid v_i' in the sale and choose neighbors $r_i' \subseteq r_i$ to invite, i.e., $t_i = (v_i', r_i')$. The intended information-revelation action is to truthfully reveal her type, $t_i^M = (v_i, r_i)$

Note that the graph is unknown to any agent, and everyone only knows who invites her and who she invites.

Sequential Resale Auction

Stage 2 - Bottom-up aggregation stage
Each buyer aggregates and passes the information through her.

Computational Action

A buyer's computational action f_i is to aggregate all bids she collected into a new bid. The intended computational computational action f_i^M is to output the largest bid from all bids she collects, i.e., $f_i^M(B_i, v_i') = \max(B_i \cup \{v_i'\})$.

Message-passing Action

A buyer's message-passing action q_i is to select one or more inviters to report her aggregated bid. The intended message-passing action q_i^M is to randomly select an inviter.

Sequential Resale Auction

Stage 3 - Top-down allocation stage

- The entire auction is modeled as a series of resales, and each resale is a local one-layer auction.
- A local auction can be a simple second-price auction with reserve price.
- The reserve price of current local seller is her purchasing price in the previous local auction.

Centralized Reduction of SRA

The centralized setting can be viewed as everyone follows the intended message-passing action and intended compuational action.
Everyone's strategy $s_i(\theta_i) = (t_i, q_i^M, f_i^M)$ is reduced to the reported type θ_i'.
Centralized reduction of SRA:

- Generate spanning trees randomly.
- Determine the path from S to the highest bidder in each spanning tree and make resales on the path.
- The expectation result on all spanning trees is the final result.

(Some specific details are omitted, refer to [3])

Advanced Reading

1. Distributed Algorithmic Mechanism Design [AGT Chapter 14]
2. Jeffrey Shneidman and David C Parkes. 2004. Specification faithfulness in networks with rational nodes. In Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing. 88-97.
3. Haoxin Liu, Yao Zhang, Dengji Zhao. AAMAS 2023. Distributed Mechanism Design in Social Networks.