

Algorithmic Game Theory

Truthful Mechanisms for Welfare Maximization

Vangelis Markakis
markakis@gmail.com

Designing welfare maximizing truthful auctions for single parameter environments

Single parameter auctions

- For the single-item case, we saw that the **Vickrey auction is ideal**
- We would like to achieve the same properties for any other type of auction
 - truthfulness and individual rationality [incentive guarantees]
 - welfare maximization [economic performance guarantees]
 - implementation in polynomial time [computational performance guarantees]
- Can we achieve all 3 properties for any single-parameter environment?

Examples of single-parameter environments

k-item unit-demand auctions

- k identical items for sale
- each bidder submits his value per unit and can win at most one unit

• Knapsack auctions

- k identical items, each bidder has a value for obtaining a certain number of units

• Single-minded auctions

- a set of (non-identical) items for sale
- each bidder is interested in acquiring a specific subset of items (known to the mechanism)
- Each bidder submits his value for the set she desires

Knapsack auctions

- We will see an illustration for knapsack auctions
- k identical items for sale
- Each bidder i has a **publicly known** demand for w_i items
 - Inelastic demand
 - The mechanism should either give w_i items to the bidder or should not give him anything
- Each bidder i submits a bid b_i for his value
- Real value per unit = v_i
- Assume the **demands** (w_1, w_2, \dots, w_n) are **known** to the mechanism
 - Say bidders have no incentive to lie about them
- Only **private information** to bidder i is v_i

Knapsack auctions

Alternative view of knapsack auctions

- The auctioneer has a resource of total capacity k (a knapsack)
- Each bidder requires **size w_i** , if he is served
- Each bidder has a **value v_i** , if he is served
- The auctioneer needs to select a **subset of bidders to serve** so as not to exceed the capacity k

Feasible allocations:

- (x_1, x_2, \dots, x_n) with $x_i \in \{0, 1\}$, and $\sum_i w_i x_i \leq k$
- Just like the feasible solutions of a knapsack problem

Knapsack auctions

Example

- Resource = the half-time break in the Champions League final
- Capacity k = total length of the break
- Each bidder corresponds to a company who wants to be advertised during the break
- The size w_i is the duration of the ad of bidder i
- The auctioneer needs to select a subset of bidders as winners and present their ads without exceeding the time capacity k

Knapsack auctions

- Let $\mathbf{b} = (b_1, b_2, \dots, b_n)$ be the bidding vector
- Need to decide the allocation and payment rule
- For the allocation rule:
 - Think of maximizing the social welfare
 - Then we have precisely the 0-1 Knapsack problem!

$$\max \sum_i b_i x_i$$

s.t.

$$\sum_i w_i x_i \leq k$$

$$x_i \in \{0, 1\}, \text{ for } i = 1, \dots, n$$

Knapsack auctions

Claim: The allocation rule that maximizes the social welfare is monotone

- Consider a winner and see what can happen if he increases his bid

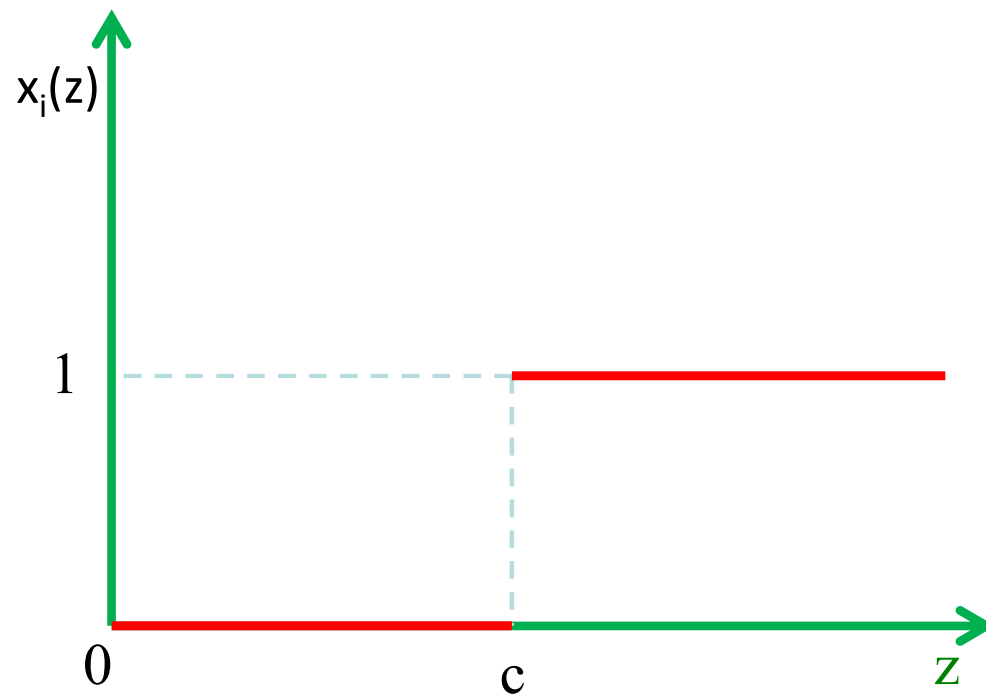
Hence, we can apply Myerson's lemma

How many jumps can we have for the allocation of a single player?

- At most one, a player can jump from being a loser ($x_i = 0$) to being a winner ($x_i = 1$)

Myerson's lemma and knapsack auctions

- The jump for a winner i happens at i 's **critical bid**: the minimum he could bid and still be a winner, also known as **threshold bid**
- Generalization of the payment in Vickrey auction



Final mechanism:

- Solve the knapsack problem and find an optimal solution
- Give to each winner i , the requested number of items w_i
- Charge the winners their **critical bid**

Myerson's lemma and knapsack auctions

Does this mechanism achieve the desirable properties we wanted?

- truthfulness [YES]
- welfare maximization [YES]
- implementation in polynomial time [?]
- Knapsack is an **NP-complete** problem
- The properties can be enforced only for special cases where Knapsack is easy
 - If highest bid or highest demand is polynomial in n (by dynamic programming)
 - If weights form a super-increasing sequence

Algorithmic Mechanism Design

- The requirement for low complexity usually comes in conflict with the other criteria
- Goal of algorithmic mechanism design: explore the trade-offs between the 3 main properties (or any other properties that we may require in a given setting)
 - Truthfulness
 - welfare maximization
 - implementation in polynomial time
- **Approach:** relax one of the criteria and see if we can achieve the others
- For Knapsack and in general whenever welfare maximization is NP-complete: resort to approximation algorithms

Knapsack auctions

Goal for Knapsack:

- Find an approximation algorithm for the social welfare
- Prove that it is **monotone**

Recall:

Definition: An algorithm A , for a maximization problem, achieves an approximation factor of γ ($\gamma \leq 1$), if for every instance I of the problem, the solution returned by A satisfies:

$$\text{SOL}(I) \geq \gamma \text{OPT}(I)$$

Where $\text{OPT}(I)$ is the value of the optimal solution for instance I

Knapsack auctions

- There are several heuristics and approximation algorithms for Knapsack, but not all of them are monotone
- A greedy $\frac{1}{2}$ -approximation:
 - For each bidder i , we care to evaluate the quantity b_i/w_i
 - Intuitively, we prefer bidders with small size/demand and large value

- **Step 1:** Sort and re-index the bidders so that

$$b_1/w_1 \geq b_2/w_2 \geq \dots \geq b_n/w_n$$

- **Step 2:** Pick bidders in that order until the first time that adding someone exceeds the knapsack capacity
- **Step 3:** Return either the previous solution, or just the highest bidder if he achieves higher social welfare on his own

Knapsack auctions

- Why do we need the last step?
- Maybe there is a bidder with a very high value, but with a large demand as well
- The algorithm may not select this bidder in the first steps
- Step 3 ensures we do not miss out such highly-valued bidders
- **Claim:** This algorithm is monotone
- **Theorem:** Using Myerson's lemma, we can have a truthful polynomial time mechanism, that produces at least 50% of the optimal social welfare

Knapsack auctions

Going further

- Knapsack also admits an FPTAS (Fully Polynomial Time Approximation Scheme)
 - We can have a $(1 - \varepsilon)$ -approximation for any constant $\varepsilon > 0$
[Ibarra, Kim '75]
 - But this is not a monotone algorithm
- [Briest, Krysta, Voecking '05]: A truthful FPTAS for Knapsack
- **Conclusion:** For a knapsack auction and any $\varepsilon > 0$, we have a truthful mechanism that produces at least $(1 - \varepsilon)$ -fraction of the optimal social welfare and runs in time polynomial in n and $1/\varepsilon$

General Approach

Suppose we have a single-parameter auction where the social welfare maximization problem is NP-hard

- Check if any of the known approximation algorithms for the problem is monotone (usually not)
- If not, then try to tweak it so as to make it monotone (sometimes feasible)
- Or design a new approximation algorithm that is monotone (hopefully without worsening the approximation guarantee)

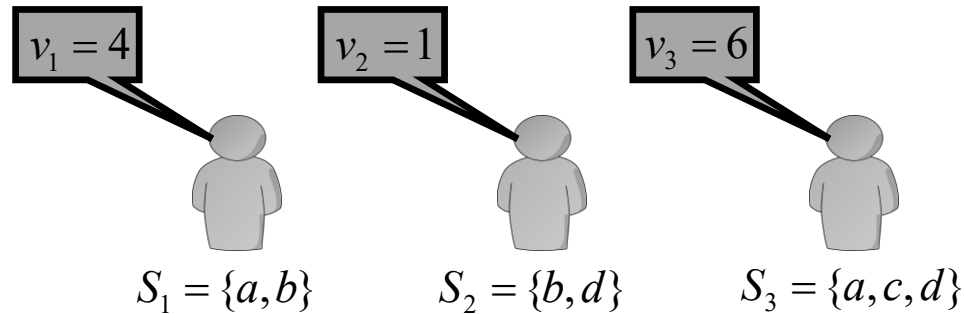
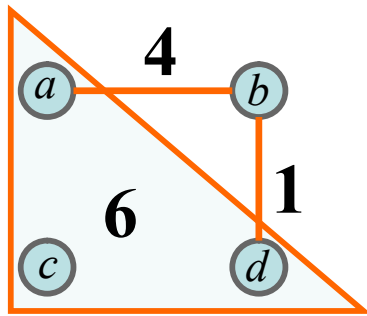
Single-minded bidders

A single-parameter auction with non-identical items

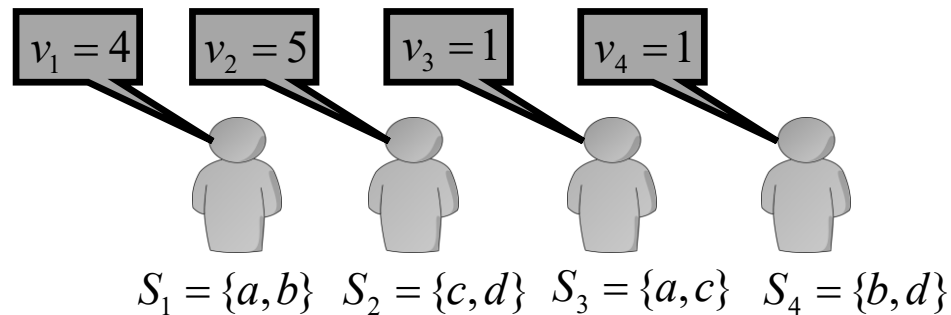
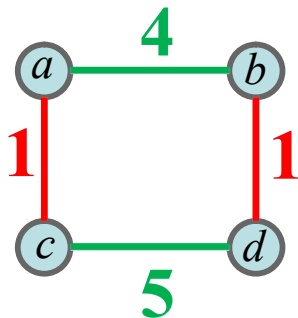
- The auctioneer has a set M of items for sale
- Each bidder i is interested in acquiring a **specific subset** of items, $S_i \subseteq M$ (**known to the mechanism**)
 - If the bidder does not obtain S_i (or a superset of it), his value is 0
- Each bidder submits a bid b_i for his value if he obtains the set
- Motivated by certain spectrum auctions
- Feasible allocations: the auctioneer needs to select winners who do not have overlapping sets

Single-minded bidders

Examples



- In the example above, the auctioneer can accept only 1 bidder as a winner
- In the example below, the auctioneer can accept up to 2 bidders as winners



Single-minded bidders

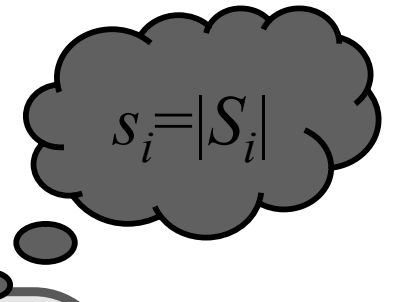
Social welfare maximization:

- Given the bids of the players, select a set of bidders with non-overlapping subsets, so as to maximize the sum of their bids
- It contains the **SET PACKING** problem, hence **NP-hard**
- Actually it gets even worse w.r.t. approximation

Theorem [Sandholm '99]: Under certain complexity theory assumptions, we cannot have an algorithm with approximation factor **better than $1/\sqrt{m}$**

Q: Can we have a $1/\sqrt{m}$ -approximation?

Single-minded bidders



[Lehmann, O' Callaghan, Shoham '01]:

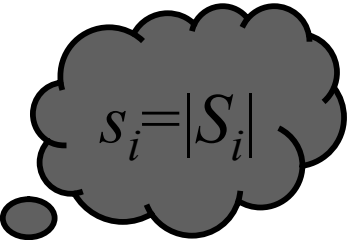
- Order the bidders in decreasing order of $b_i/\sqrt{s_i}$
- Accept each bidder in this order unless overlapping with previously accepted bidders
- Payment i : largest bid b_j for set S_j with nonempty intersection with S_i .

• This algorithm achieves

- $1/\sqrt{m}$ -approximation, where $m = |M|$
- $1/d$ -approximation, where $d = \max_i s_i$
- Monotonicity and truthfulness.

Final conclusion: truthful polynomial time mechanism with the best possible approximation to the social welfare

Single-minded bidders


$$s_i = |S_i|$$

- Order the bidders in decreasing order of $b_i/\sqrt{s_i}$
 - Accept each bidder in this order unless overlapping with previously accepted bidders
-
- A algorithm's solution (set of indices accepted by Greedy)
 - O optimal solution (set of indices accepted by OPT)

Wlog. assume that $O \cap A = \emptyset$.

Partition O into O_i , $i \in A$, s.t. $j \in O_i$ if $j \in O$ and $S_i \cap S_j \neq \emptyset$.

$$\sum_{j \in O_i} v_j \leq \frac{v_i}{\sqrt{s_i}} \sum_{j \in O_i} \sqrt{s_j} \quad \text{Greedy property}$$

$$\leq \frac{v_i}{\sqrt{s_i}} \sqrt{\sum_{j \in O_i} s_j} \sqrt{|O_i|} \quad \text{Cauchy-Schwarz ineq.}$$

$$\leq \frac{v_i}{\sqrt{s_i}} \sqrt{m} \sqrt{s_i} \quad |O_i| \leq s_i \text{ and } \sum_{j \in O_i} s_j \leq m$$

$$\leq v_i \sqrt{m}$$

Wrapping Up

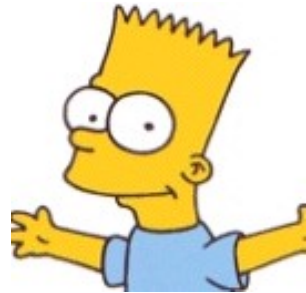
- **Single parameter** bidders: private information of bidder i is **single value** v_i , expressed by **bid** b_i
- **Myerson's Lemma**: truthful mechanism iff monotone allocation, payments are uniquely determined (and virtually always easy to compute).
 - 2nd price / Vickrey auction is the **only** truthful **single-item** auction.
 - Optimal is always monotone: if allocation problem is easy, we also get **computational efficiency**.
 - If allocation problem is hard, we seek **monotone poly-time approximation** algorithms.
 - **(1-1/k)-approximation** in **time** $O(n^{k+1})$ and **FPTAS** for Knapsack (with demand known).
 - Single-minded bidders / set packing: Greedy wrt $b_i/\sqrt{s_i}$ is monotone and $O(\sqrt{m})$ -approximation (best possible approximation in polynomial time).

Multi-dimensional Bidders / Combinatorial Auctions

The model

Set of players

$$N = \{1, 2, \dots, n\}$$



Set of **indivisible goods**

$$M = \{1, 2, \dots, m\}$$



Combinatorial Auctions

- Any auction with **multiple items** for sale
- The players may be allowed to express interest / bids on **various combinations** of goods
- In practice very active field within the last 10-15 years
 - Spectrum licenses
 - The FCC incentive auction:
 - <https://www.fcc.gov/about-fcc/fcc-initiatives/incentive-auctions>
 - Transportation routes
 - Logistics

Combinatorial auctions

- In practice, it seems economically more efficient and profitable to **sell the items together** than have a separate auction for each good
- Main challenges:
 - **Algorithmic:** How shall we design the **allocation rule** (especially if we have many overlaps in what the players want the most)?
 - **Game-theoretic:** Can we **generalize Myerson's lemma** to get truthful mechanisms?

Valuation functions

- So far we studied settings where a single parameter v_i determined all the information we needed for a player
- Most general scenario: consider that each player has a **valuation function** defined for **every subset of the items**
- $v_i : P(M) \rightarrow R$
 - where $P(M)$ = powerset of M (all subsets of M)
 - For every $S \subseteq M$,
 - $v_i(S)$ = value of player i if he acquires set S
 - $b_i(S)$ = maximum amount willing to pay for acquiring S
- We always assume **monotonicity** (“free-disposal”):
for all $T \subseteq S$, $v_i(T) \leq v_i(S)$.

Examples of valuation functions

Additive valuation functions

- For every $S \subseteq M$, $v_i(S) = \sum_{j \in S} v_{ij}$
 - where v_{ij} = utility of acquiring item j
- Hence, the function can be completely determined by specifying the vector $(v_{i1}, v_{i2}, \dots, v_{im})$
- m parameters for each bidder
- In such cases, the goods can be **auctioned independently**:
 - The value of an item is not affected by other items that a bidder may have already obtained

Examples of valuation functions

- In practice, the items may be interrelated with each other and additive valuations are not appropriate
- The value they add to a player may depend on the other items that the player has
- The items may exhibit
 - **Complementarity:** some items may be valuable only when they are sold together with other items (e.g. left and right shoe)
 - **Substitutability:** some items may be of similar type and should not be sold together to the same player (e.g. 2 cars with the same features)

Examples of valuation functions

Subadditive functions

- For any 2 disjoint subsets $S \subseteq M$, $T \subseteq M$,

$$v_i(S \cup T) \leq v_i(S) + v_i(T)$$

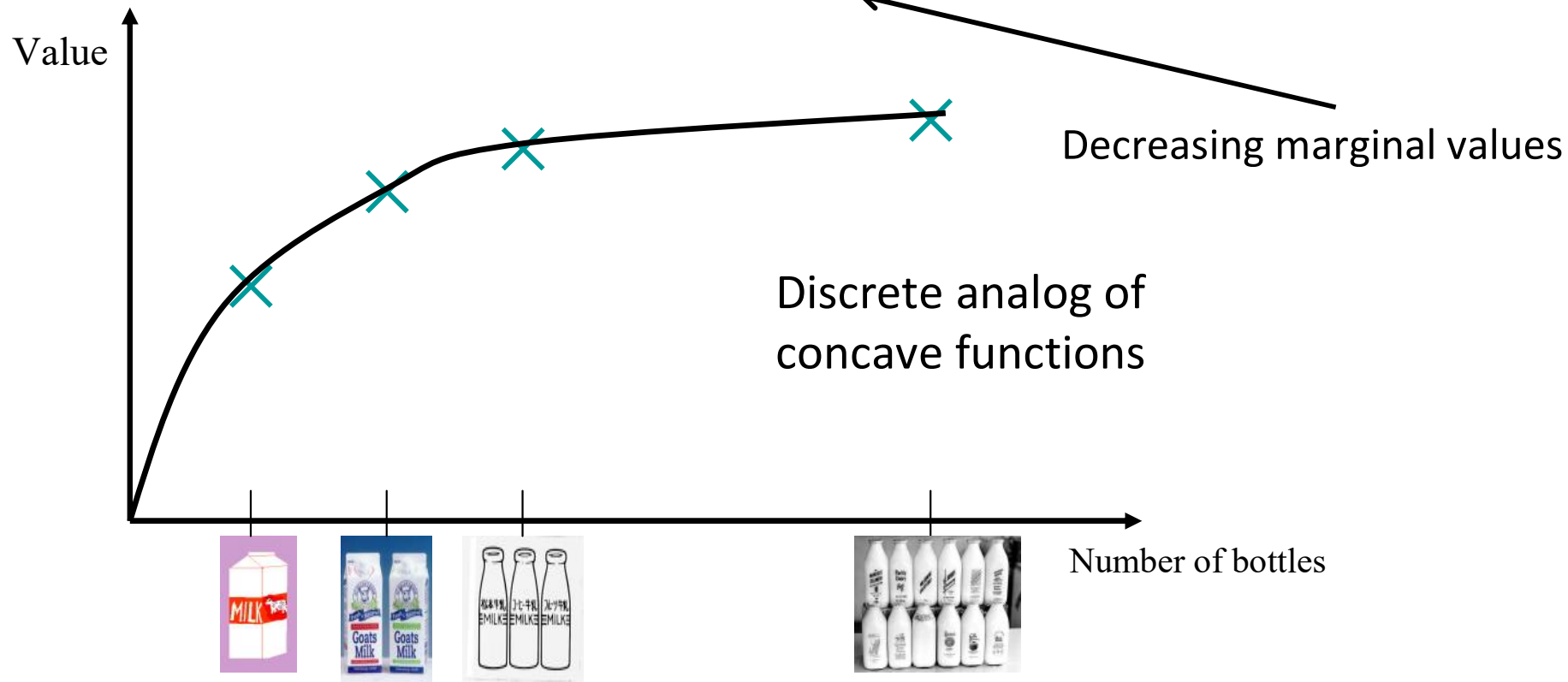
- In this case, we have **substitutability** among the goods
- They are also called **complement-free** functions (since we do not have complementarity)

Examples of valuation functions

Submodular functions

For any 2 subsets S, T , with $S \subseteq T \subseteq M$, and for every $j \notin T$

$$v_i(T \cup \{j\}) - v_i(T) \leq v_i(S \cup \{j\}) - v_i(S)$$



Examples of valuation functions

- Submodular functions form a **special class** of subadditive valuations
- Hence, they also do not exhibit complementarity
- They play a key role in micro-economic theory
- Expressing the fact that **utility gets “saturated”** as we keep allocating substitutes to the same player

Examples of valuation functions

Symmetric submodular

- Special case of submodular functions, where all **goods are identical**
 - Hence, the final utility depends only on **how many items** the player receives
- Applicable for multi-unit auctions
 - E.g., auctions for government bonds fall under this framework
- For k identical items, such functions can be represented by **a vector of k marginal values**
 - $(m_i(1), m_i(2), \dots, m_i(k))$ with $m_i(j) \geq m_i(j+1)$
 - Where $m_i(j)$ = additional utility to the player for obtaining the j -th unit, if the player already has $j-1$ units

Examples of valuation functions

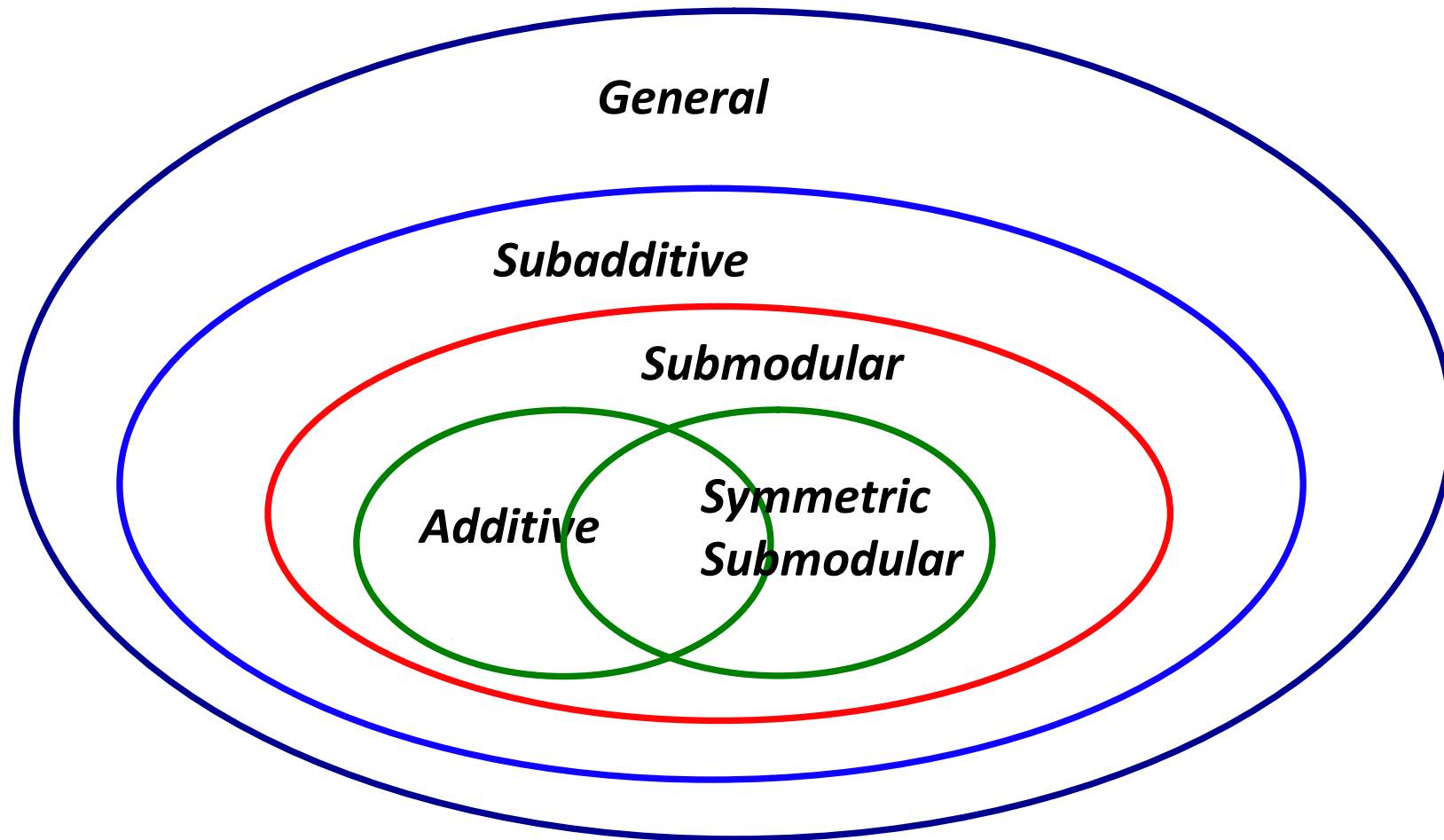
Superadditive functions

- For any 2 disjoint subsets $S \subseteq M$, $T \subseteq M$,

$$v_i(S \cup T) \geq v_i(S) + v_i(T)$$

- In this case, we have **complementarity**
- For example, the items may not have any value if they are sold on their own, but only when sold in bundles with other goods
 - Single-minded bidders fall under this class

Relations between different classes of valuation functions



Social Welfare Maximization

- Need to define social welfare in this more general setting
- **Definition:** Let $\mathbf{S} = (S_1, S_2, \dots, S_n)$ be an allocation of the items to the players, where S_i = subset assigned to player i . Then the social welfare derived from \mathbf{S} is

$$SW(\mathbf{S}) = \sum_i v_i(S_i)$$

The SWM problem (Social Welfare Maximization):

Input: The valuation functions of the players (**how?**)

Output: Find an allocation $\mathbf{S}^* = (S_1, S_2, \dots, S_n)$ that produces the highest possible social welfare:

$$SW(\mathbf{S}^*) \geq SW(\mathbf{S}) \text{ for any other allocation } \mathbf{S}$$

Social welfare maximization

Example with additive valuations

- 3 players, 4 items
- The input can be determined by a 3 x 4 array

48	41	11	0
35	10	50	5
45	20	10	25

- Optimal allocation: $S^* = (S_1, S_2, S_3) = (\{1, 2\}, \{3\}, \{4\})$
- Optimal social welfare: $48 + 41 + 50 + 25 = 164$

Integer Programming Formulation

$$\max \sum_{i,S} x_{i,S} v_i(S)$$

$$\sum_S x_{i,S} \leq 1 \quad \forall i \in [n]$$

$$\sum_{i,S:j \in S} x_{i,S} \leq 1 \quad \forall j \in [m]$$

$$x_{i,S} \geq 0$$

$$\min \sum_{j \in [m]} p_j + \sum_{i \in [n]} u_i$$

$$u_i \geq v_i(S) - \sum_{j \in S} p_j \quad \forall i, S$$

$$p_j \geq 0 \quad \forall j \in [m]$$

$$u_i \geq 0 \quad \forall i \in [n]$$

$$u_i = \max_S \{v_i(S) - p(S)\}$$

- p_j is the **price** of item j and u_i is the **utility** of bidder i
- Complementary slackness: in optimal solution (assuming integrality), **each bidder gets a utility maximizing set and each item with positive price is allocated.**
- Optimal solutions, if integral, correspond to equilibrium!

Walrasian (Competitive) Equilibrium

- **Competitive (Walrasian) equilibrium** is price vector $\mathbf{p} = (p_1, \dots, p_m)$ and allocation $\mathbf{S}^* = (S_1, \dots, S_m)$ such that
 - $v_i(S_i) - p(S_i) \geq v_i(S) - p(S)$, for any subset S of items.
 - Every item j with $p_j > 0$ is allocated.
- **Example:** two bidders Alice and Bob, two items x and y .
 - Alice has value 2 for x , y and $x+y$, 0 for empty set.
 - Bob has value 4 for $x+y$ and 0 for anything else.
 - $p_x = p_y = 2$, Alice nothing, Bob $x+y$ is equilibrium.
 - If Bob had **value 3 for $x+y$** and 0 for anything else, Walrasian equilibrium does **not** exist!

Walrasian (Competitive) Equilibrium

- **Competitive (Walrasian) equilibrium** is price vector $\mathbf{p} = (p_1, \dots, p_m)$ and allocation $\mathbf{S}^* = (S_1, \dots, S_m)$ such that
 - $v_i(S_i) - p(S_i) \geq v_i(S) - p(S)$, for any subset S of items.
 - Every item j with $p_j > 0$ is allocated.
- **First Welfare Theorem:** (If exists,) Walrasian equilibrium maximizes social welfare, even among fractional solutions.

For any feasible (fractional) solution $x_{i,S}$, for any bidder i ,

$$v_i(S_i) - \sum_{j \in S_i} p_j \geq \sum_S x_{i,S} \left(v_i(S) - \sum_{j \in S} p_j \right) \quad (1)$$

by first condition and because $\sum_S x_{i,S} \leq 1$.

- We sum up (1) and observe that **sums of prices cancel out**, because allocations must be **disjoint**.

Walrasian (Competitive) Equilibrium

- **Competitive (Walrasian) equilibrium** is price vector $\mathbf{p} = (p_1, \dots, p_m)$ and allocation $\mathbf{S}^* = (S_1, \dots, S_m)$ such that
 - $v_i(S_i) - p(S_i) \geq v_i(S) - p(S)$, for any subset S of items.
 - Every item j with $p_j > 0$ is allocated.
- **Second Welfare Theorem:** If LP admits integral optimal solution, then Walrasian equilibrium exists.
 - Follows from complementary slackness conditions.
- LP admits integral optimal solution for **gross substitutes**.
 - When price for an item increases and prices for other items remain constant, the demand for other items does not decrease.
 - Walrasian equilibrium computed by natural tatonnement process.
 - [Kelso-Crawford, '82] Special case of discrete convexity!!!
 - <http://www.inbalalgam.com/slides/GS%20Tutorial%20Part%20I.pdf> and <http://www.inbalalgam.com/slides/GS%20Tutorial%20Part%20II.pdf>

Walrasian Tatonnement

- Demand correspondence:

$$D(v, p) = \{S \subseteq U : v(S) - p(S) \geq v(T) - p(T), \forall T \subseteq U\}$$
$$D_i(p) = \{S \subseteq U : v_i(S) - p(S) \geq v_i(T) - p(T), \forall T \subseteq U\}$$

An item-price ascending auction for substitutes valuations:

Initialization:

For every item $j \in M$, set $p_j \leftarrow 0$.
For every bidder i let $S_i \leftarrow \emptyset$.

Repeat

For each i , let D_i be the demand of i at the following prices:

p_j for $j \in S_i$ and $p_j + \epsilon$ for $j \notin S_i$.

If for all i $S_i = D_i$, exit the loop;

Find a bidder i with $S_i \neq D_i$ and update:

- For every item $j \in D_i \setminus S_i$, set $p_j \leftarrow p_j + \epsilon$
- $S_i \leftarrow D_i$
- For every bidder $k \neq i$, $S_k \leftarrow S_k \setminus D_i$

Finally: Output the allocation S_1, \dots, S_n .

Mechanisms for Combinatorial Auctions

How do the players describe their valuations to auctioneer?

- For a general function, the bidder would need to specify $v_i(S)$, for every $S \subseteq M$ (2^m numbers, prohibitive!)
- Three approaches:
 1. Some functions can be described with a **small number of parameters**
 - E.g. additive or symmetric submodular (**m parameters**)
 2. The auctioneer can ask the bidders during the auction for their **values on certain subsets** of items
 - **Value queries.**
 - No need to know the entire function.
 3. The auctioneer computes prices and let the bidders decide on **their utility maximizing set.**
 - **Demand queries** – NP-hard to compute, in general.
 - No information about valuation is given to auctioneer.

Mechanisms for Combinatorial Auctions

- Truthful mechanisms for combinatorial auctions?
- Can we generalize the 2nd price auction when we have multiple items?
- We need to generalize:
 - **The allocation algorithm:** with 1 item, the winner was the highest bidder
 - multiple winners (with non-overlapping sets of goods), but monotonicity still necessary!
 - **The payment rule:** with 1 item, we offered a «discount» to the winner
 - Adjust the discount to the more general setting (and we also need a separate discount for each winner)

The VCG mechanism

- A generalization of the Vickrey auction
- Named after [Vickrey '61, Clarke '71, Groves '73]
 1. $\mathbf{S}^* = (S_1, S_2, \dots, S_n)$ social welfare maximizing allocation.
 2. **Allocation rule:** For $i=1, \dots, n$, player i receives set S_i
 3. **Payment rule:**
 - Payment of player i : $p_i = SW_{-i}^* - \sum_{j \neq i} v_j(S_j)$
where SW_{-i}^* = optimal social welfare without player i
 - Every player pays the “externality” that his presence causes to the welfare of the others
 - **Utility** (value – payment) of player i : $u_i = SW^* - SW_{-i}^*$
 - Every player has utility equal to the increase in the social welfare due to his presence.

The VCG mechanism

In conclusion:

- Every player receives the items specified by the optimal allocation (w.r.t. the social welfare)
- His payment is determined by the declarations of the other players, just as in the Vickrey auction

Theorem: For any valuation functions, the VCG mechanism is truthful and maximizes the social welfare

Can we implement efficiently the VCG mechanism?

-Only when we can solve the SWM problem efficiently

The VCG Mechanism Truthfulness

Fix i and \mathbf{b}_{-i} . When the chosen outcome $\mathbf{x}(\mathbf{b})$ is ω^* , i 's utility is

$$v_i(\omega^*) - p_i(\mathbf{b}) = \underbrace{\left[v_i(\omega^*) + \sum_{j \neq i} b_j(\omega^*) \right]}_{(A)} - \underbrace{\left[\max_{\omega \in \Omega} \sum_{j \neq i} b_j(\omega) \right]}_{(B)}.$$

- Part (B) is **independent of i ' bid b_i** (truthfulness holds **for any (B)** that does not depend on b_i).
 - Part (B), a.k.a. **Clarke pivot rule**, ensures non-positive transfers (NPT) and individual rationality (IR).
- Bidding **truthfully**, i.e. $b_i = v_i$, allows the mechanism to **maximize part (A)**, which is exactly what player i wants!
 - Players' **incentives fully aligned with objective** of mechanism!

Implementing the VCG mechanism

How to compute the allocation and the payment rule of VCG:

- It suffices to solve $n+1$ instances of the SWM problem
- **1 instance** with all players present to determine the **allocation** of the items
- n more instances with a different player absent each time (SWM with $n-1$ out of the initial n players)
- Final complexity: $O(n) \cdot (\text{complexity of SWM})$

Implementing the VCG mechanism

Additive valuations

- Input: $n \times m$ matrix
- Solving SWM: Easy, greedy algorithm
 - For every item j : give it to the player with the highest value
- Implementing the payment rule of VCG:
 - Easy, solve n more times SWM with 1 player absent each time

Implementing the VCG mechanism

Example with additive valuations

- 3 players, 4 items

48	41	11	0
35	10	50	5
45	20	10	25

- Optimal allocation: $S^* = (S_1, S_2, S_3) = (\{1, 2\}, \{3\}, \{4\})$
- Optimal social welfare: $48 + 41 + 50 + 25 = 164$

Implementing the VCG mechanism

Example with additive valuations

- 3 players, 4 items

48	41	11	0
35	10	50	5
45	20	10	25

Payments:

- $p_1 = SW_{-1}^* - \sum_{j \neq 1} v_j(S_j) = 140 - (50+25) = 65$
- $p_2 = SW_{-2}^* - \sum_{j \neq 2} v_j(S_j) = 125 - (89+25) = 11$
- Similarly, $p_3 = 5$

Implementing the VCG mechanism

Additive valuations

- What if we run m independent Vickrey auctions for every item separately?
- We get the same result!
- It is due to the fact that we have additive valuations (hence, the values of different items for a player are not correlated)

Corollary:

For additive valuations, the VCG mechanism is equivalent to executing an independent Vickrey auction for each item

Implementing the VCG mechanism

Submodular functions?

Good news

Theorem: The VCG mechanism can be implemented in polynomial time for **symmetric submodular** valuations

- Greedy (wrt. marginal values) allocation is optimal.

Bad news

- For general submodular valuations, SWM is NP-complete
 - Reduction from Knapsack
- The same also holds for subadditive valuations

Implementing the VCG mechanism

Submodular functions?

[Lehmann, Lehmann, Nisan '01]: greedy, 1/2-approximation

- Fix an ordering of the goods, $1, 2, \dots, m$
- For $j = 1, \dots, m$
 - Let (S_1, S_2, \dots, S_n) be the current allocation to the bidder
 - Allocate next good to the bidder with **currently highest marginal value** for this good
 - i.e., calculate $v_i(S_i \cup \{j\}) - v_i(S_i)$ for each player i
 - We measure how much extra welfare is derived by adding the good to the currently assigned bundle of a player

Implementing the VCG mechanism

Submodular functions?

- **Further progress:** $(1 - 1/e \approx 0.632)$ -approximation with value queries [Vondrak '08]
- [Mirrokni, Schapira, Vondrak '08]: Better approximation would require exponentially many value queries.
- Unfortunately these algorithms cannot be combined with the VCG payment formula to obtain a truthful mechanism
- Open problem to derive a **computationally efficient truthful** mechanism for submodular valuations with the best possible approximation to the social welfare

Truthful Mechanisms for Subadditive Valuations

Value Queries [Dobzinski, Nisan, Schapira 05]:

1. Query each bidder for values of all singleton sets and U .
2. Find best “matching” allocation where **each bidder gets at most one good** (maximum bipartite matching).
 - **Complete bipartite** graph with **agents** on the left, **goods** on the right, and weight $v_i(\{j\})$ on each **{ agent i , good j }** edge.
3. Return best of maximum “matching” and $\max\{v_i(U)\}$
 - Algorithm finds **optimal over subset of feasible allocations**, that includes only “matchings” and “winner-takes-all”.
 - **Maximal-in-Range** (MiR) mechanisms: optimize over a **predetermined subset** of feasible solutions (a.k.a. “range”).
 - Allocation is optimal-in-range: **truthfulness with VCG payments!**
 - Range chosen to guarantee **good approximation** and **polynomial-time optimization**.

Truthful Mechanisms for Subadditive Valuations

Value Queries [Dobzinski, Nisan, Schapira 05]:

1. Query each bidder for values of all singleton sets and U .
2. Find best “matching” allocation where **each bidder gets at most one good** (maximum bipartite matching).
 - Complete bipartite graph with **agents** on the left, **goods** on the right, and weight $v_i(\{j\})$ on each **{ agent i , good j }** edge.
3. Return best of maximum “matching” and $\max\{v_i(U)\}$
 - Approximation ratio $O(\sqrt{m})$ for **subadditive** valuations.
 - If **most of OPT SW by “large sets”** (cardinality $\geq \sqrt{m}$, so at most \sqrt{m} of them), $\max\{v_i(U)\}$ is \sqrt{m} -approximation.
 - If **most of OPT SW by “small sets”** (cardinality $< \sqrt{m}$) maximum “matching” is \sqrt{m} -approximation (due to subadditivity and bound on cardinality).

Truthful Mechanisms for Subadditive Valuations

Value Queries [Dobzinski, Nisan, Schapira '05]:

1. Query each bidder for values of all singleton sets and U .
2. Find best “matching” allocation where **each bidder gets at most one good** (maximum bipartite matching).
 - **Complete bipartite** graph with **agents** on the left, **goods** on the right, and weight $v_i(\{j\})$ on each **{ agent i , good j }** edge.
3. Return best of maximum “matching” and $\max\{v_i(U)\}$
 - **Theorem.** MiR algorithm above is truthful with VCG payments and achieves \sqrt{m} -approximation for subadditive valuations.
 - **Maximal-in-Distributional Range** gives \sqrt{m} -approximation for CAs with **general** valuations [Lavi, Swamy '05]

<https://www.cs.princeton.edu/~smattw/Teaching/521fa17lec19.pdf>

<https://www.math.uwaterloo.ca/~cswamy/papers/mechdeslp-journ.pdf>

Linear Programming Relaxation of Social Welfare Maximization

$$\max \sum_{i,S} x_{i,S} v_i(S)$$

$$\sum_S x_{i,S} \leq 1 \quad \forall i \in [n]$$

$$\sum_{i,S:j \in S} x_{i,S} \leq 1 \quad \forall j \in [m]$$

$$x_{i,S} \geq 0$$

$$\min \sum_{j \in [m]} p_j + \sum_{i \in [n]} u_i$$

$$u_i \geq v_i(S) - \sum_{j \in S} p_j \quad \forall i, S$$

$$p_j \geq 0 \quad \forall j \in [m]$$

$$u_i \geq 0 \quad \forall i \in [n]$$

- p_j is the **price** of item j and u_i is the **utility** of bidder i

$$u_i = \max_S \{v_i(S) - p(S)\}$$

$$D_i(U_i, p) = \{S \subseteq U_i : v_i(S) - p(S) \geq v_i(T) - p(T), \forall T \subseteq U_i\}$$

Truthful Mechanisms for Submodular Valuations

Demand Queries [Krysta, Vocking, '12]:

Algorithm 1. Overselling MPU algorithm

- 1 For each good $e \in U$ do $p_e^1 := p_0$.
 - 2 For each bidder $i = 1, 2, \dots, n$ do
 - 3 Set $S_i := D_i(U_i, p^i)$, for a suitable $U_i \subseteq U$.
 - 4 Update for each good $e \in S_i$: $p_e^{i+1} := p_e^i \cdot 2$
-

- **Binary search** in **optimal prices** of goods!
- **Truthful** because prices p_i do not depend on bidder i and demand queries.
- If $p_0 = \max\{v_i(U)\} / (4m)$, Alg1 allocates $\leq \log_2(4m)+1$ copies of each good.
 - After allocating so many copies of good e ,
 $p_e > \max\{v_i(U)\}$ and no player can afford it anymore.

Truthful Mechanisms for Submodular Valuations

Lemma. p_e^* denotes final price of good e . Then,

$$\text{Alg} = \sum_{i=1}^n v_i(S_i) \geq \sum_{e \in U} p_e^* - mp_0$$

$$\sum_{i=1}^n v_i(S_i) \geq \sum_{i=1}^n \sum_{e \in S_i} p_e^i$$

$$= \sum_{i=1}^n \sum_{e \in S_i} 2^{\ell_e^i} p_0$$

ℓ_e^i = copies of i sold before i

$$= p_0 \sum_{e \in U} \sum_{k=1}^{\ell_e^* - 1} 2^k$$

ℓ_e^* = copies i sold in total

$$= p_0 \sum_{e \in U} (2^{\ell_e^*} - 1)$$

$$= \sum_{e \in U} p_e^* - mp_0 = \sum_{e \in U} p_e^* - \frac{\text{OPT}}{4}$$

$$p_0 = \frac{\max\{v_i(U)\}}{4m} \leq \frac{\text{OPT}}{4m}$$

Truthful Mechanisms for Submodular Valuations

- Approximation ratio: compare social welfare of Alg and OPT
 - Demand query ensures that:

$$\forall \text{ player } i, \quad v_i(S_i) \geq v_i(S_i^*) - \sum_{e \in S_i^*} p_e^*$$

- Summing up and using Lemma:

$$\text{Alg} \geq \text{OPT} - \sum_{e \in U} p_e^* \geq \text{OPT} - \text{Alg} - \frac{\text{OPT}}{4}$$

- We get **Alg \geq 3OPT/8** (but with logarithmic “overselling”).

Truthful Mechanisms for Submodular Valuations

Algorithm 1. Overselling MPU algorithm

- 1 For each good $e \in U$ do $p_e^1 := p_0$.
 - 2 For each bidder $i = 1, 2, \dots, n$ do
 - 3 Set $S_i := D_i(U_i, p^i)$, for a suitable $U_i \subseteq U$.
 - 4 Update for each good $e \in S_i$: $p_e^{i+1} := p_e^i \cdot 2$
-

- “Overselling” is fixed with oblivious rounding and sets U_i
 - U_i is the set of **available goods** at step i .
 - **After** the demand query $D_i(U_i, p^i)$ is answered, S_i is allocated with probability $1/\log_2(4m)$
 - Approximation ratio increases by factor $O(\log_2(4m))$ for **submodular** valuations.
- Demand-query truthful approximations extends to budgeted bidders and liquid welfare: $\text{LiquidValuation}_i(S) = \min\{v_i(S), B_i\}$

Negative Cycles, Monotonicity and Truthfulness

- Consider allocation (and truthfulness) from viewpoint of single bidder (as in Myerson's Lemma, but multi-dimensional)
 - Fix allocation rule \mathbf{x} , other bids \mathbf{b}_{-i} and payments \mathbf{p} .
 - Consider allocation $\mathbf{x}(\mathbf{b})$, payments $\mathbf{p}(\mathbf{b})$ and utility $v(\mathbf{x}(\mathbf{b})) - \mathbf{p}(\mathbf{b})$ of bidder i as functions of i 's bid \mathbf{b} and i 's true valuation (a.k.a. type) v .
 - We want to characterize allocation rules \mathbf{x} that allow for truthful payments \mathbf{p} (similar to Myerson's Lemma).
 - Definition of truthfulness:
$$v(\mathbf{x}(v)) - \mathbf{p}(v) \geq v(\mathbf{x}(\mathbf{b})) - \mathbf{p}(\mathbf{b}), \text{ for all types } v, \mathbf{b}$$
 - Focus on discrete domains (finite set of types), but everything generalizes to infinite (and continuous) domains.

Negative Cycles, Monotonicity and Truthfulness

- Let D set of all possible types.
- Correspondence graph $G(D, E, w)$ is an edge-weighted **complete** directed graph on D .
 - Let b and b' be two types / vertices and $o = \mathbf{x}(b)$ and $o' = \mathbf{x}(b')$ corresponding outcomes.
 - $w(b, b') = b(o) - b(o')$ (and $w(b', b) = b'(o') - b'(o)$).
 - **When true type b** , how much bidder prefers o (outcome if he is **truthful**) to o' (outcome if he **misreports** b')
 - Payments p **function of outcomes** (only)!
- Allocation **\mathbf{x} is truthful** (without payments!) iff $w(b, b') \geq 0$, for all edges (b, b') .

Negative Cycles, Monotonicity and Truthfulness

- Correspondence graph $G(D, E, w)$.
 - Let b, b' be types and $o = \mathbf{x}(b), o' = \mathbf{x}(b')$ outcomes.
 - $w(b, b') = b(o) - b(o')$ (and $w(b', b) = b'(o') - b'(o)$).
- Allocation \mathbf{x} admits **truthful** payments $\mathbf{p} : \text{Outcomes} \rightarrow \mathbb{R}_+$, if all edges (b, b') become non-negative after we apply \mathbf{p} :
$$b(o) - p(o) \geq b(o') - p(o')$$
- Allocation \mathbf{x} admits **truthful** payments \mathbf{p} iff $G(D, E, w)$ does **not** have **negative cycles!**
 - Truthful payments computed by **Johnson's algorithm!**
- If domain D is **convex**, allocation \mathbf{x} admits **truthful** payments \mathbf{p} iff $G(D, E, w)$ does **not** have **negative 2-cycles.**
 - Weak monotonicity: $b(o) - b'(o) \geq b(o') - b'(o')$, for all b, b' [Zaks, Yu '05]

Research questions on the implementation of truthful mechanisms

- Find special cases where SWM is solvable in polynomial time
- Design approximation algorithms for SWM for various types of valuation functions
- **General problem with approximation algorithms:** they cannot always be combined with some payment rule and get a truthful mechanism
- At the end, we need to understand how truthful mechanisms look like for multi-parameter environments, esp. when SWM is difficult