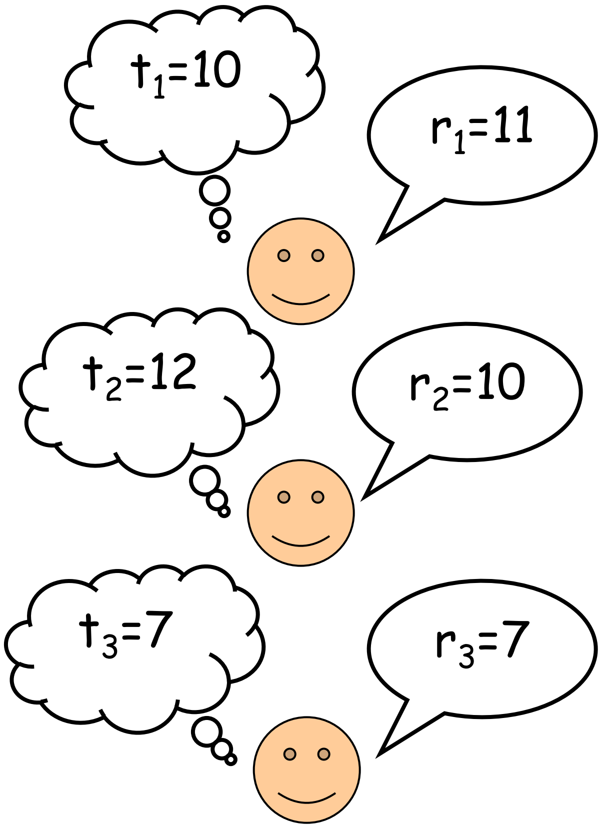
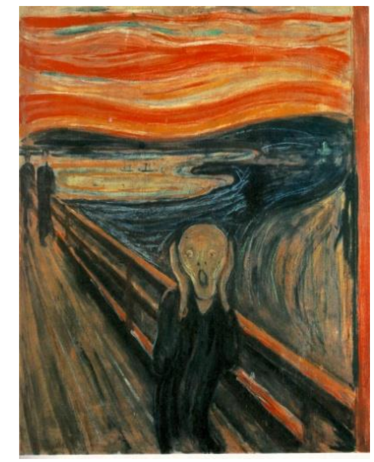
Combinatorial Auction

A single item auction



r_i: is the amount of money player i **bids** (in a sealed envelope) for the painting

Social-choice function: the winner should be the guy **having in mind** the highest value for the painting

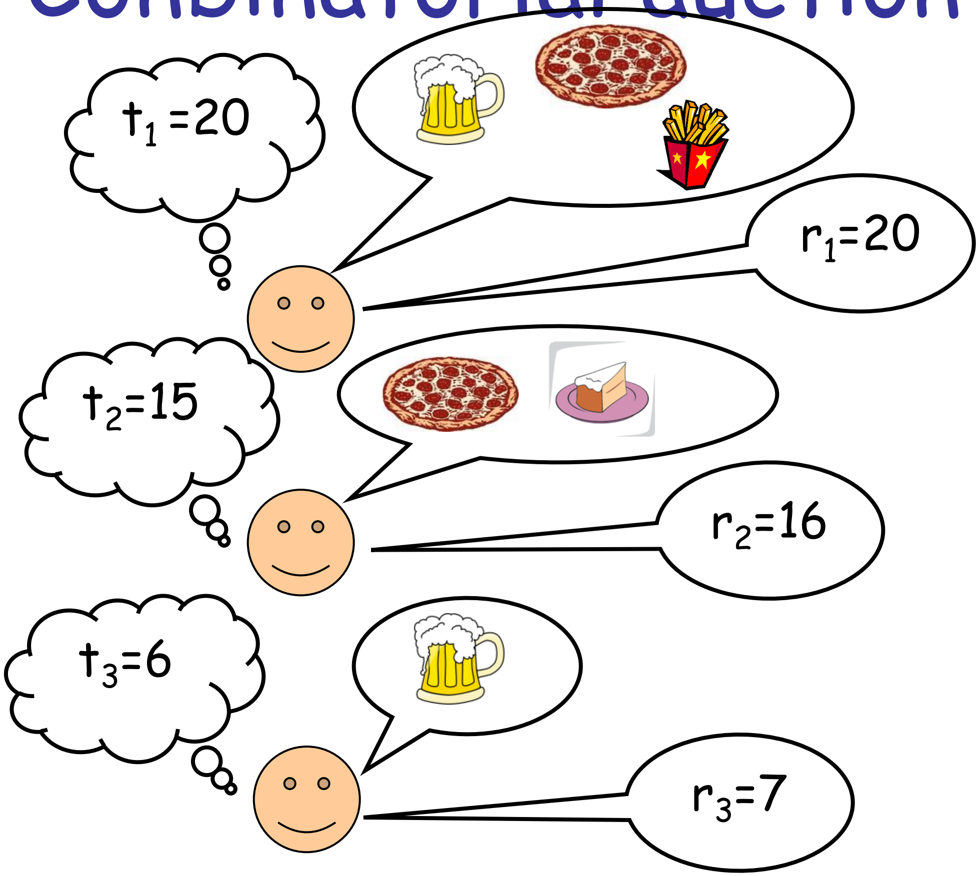


The mechanism tells to players:
(1) How the item will be allocated (i.e., who will be the winner), depending on the received bids
(2) The payment the winner has to return, as a function of the received bids

t_i: is the **maximum** amount of money player i is willing to pay for the painting

If player i wins and has to pay p its utility is $u_i=t_i-p$

Conbinatorial auction



Each player wants a bundle of objects

t_i: value player i is willing to pay for its bundle

if player i gets the bundle at price p his utility is $u_i=t_i-p$

the mechanism decides the set of winners and the corresponding payments

F={ $W \subseteq \{1,\ldots,N\}$: winners in W are compatible}

Combinatorial Auction (CA) problem – single-minded case

- Input:
  - n buyers, m indivisible objects
  - each buyer i:
    - Wants a subset S_i of the objects
    - has a value t_i for S_i
- Solution:
  - $W \subseteq \{1,\ldots,n\}$, such that for every $i,j \in W$, with $i \neq j$, $S_i \cap S_j = \emptyset$
- Measure (to maximize):
  - Total value of W: $\sum_{i \in W} t_i$

CA game

- each buyer i is selfish
- Only buyer i knows t_i (while S_i is public)
- We want to compute a “good” solution w.r.t. the true values
- We do it by designing a mechanism
- Our mechanism:
  - Asks each buyer to report its value v_i
  - Computes a solution using an output algorithm $g(\cdot)$
  - takes payments p_i from buyer i using some payment function p

More formally

- Type of agent buyer i:
  - t_i: value of S_i
  - Intuition: t_i is the maximum value buyer i is willing to pay for S_i
- Buyer i's valuation of $W \in F$:
  - $v_i(t_i, W) = t_i$ if $i \in W$, 0 otherwise
- SCF: a good allocation of the objects w.r.t. the true values

How to design a truthful mechanism for the problem?

Notice that:

the (true) total value of a feasible W is:

$$\sum_{i \in W} t_i = \sum_i v_i(t_i, W)$$

the problem is utilitarian!

…VCG mechanisms apply

VCG mechanism

- $M = \langle g(r), p(x) \rangle$:
  - g(r): $x^* = \arg\max_{x \in F} \sum_j v_j(r_j, x)$
  - $p_i(r)$: for each i:

$$p_i(r) = \sum_{j \neq i} v_j(r_j, g(r_{-i})) - \sum_{j \neq i} v_j(r_j, x^*)$$

g(r) has to compute an optimal solution…

…can we do that?

Theorem

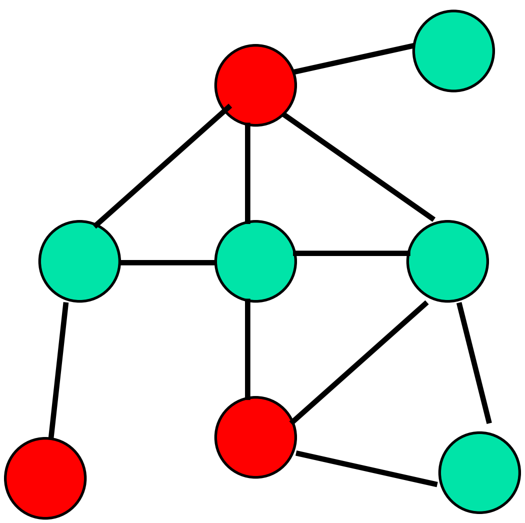
Approximating CA problem within a factor better than $m^{1/2-\varepsilon}$ is NP-hard, for any fixed $\varepsilon>0$.

proof

Reduction from maximum independent set problem

Maximum Independent Set (IS) problem

- Input:
  - a graph $G=(V,E)$
- Solution:
  - $U \subseteq V$, such that no two vertices in U are jointed by an edge
- Measure:
  - Cardinality of U

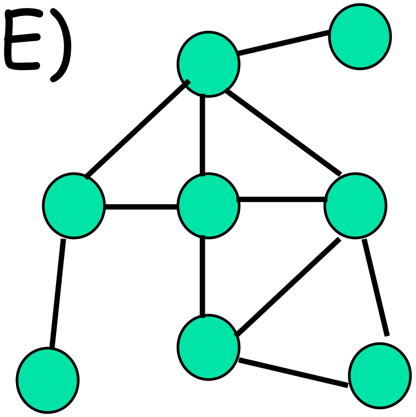


Theorem (J. Håstad, 2002)

Approximating IS problem within a factor better than $n^{1-\varepsilon}$ is NP-hard, for any fixed $\varepsilon>0$.

the reduction

G=(V,E)



each edge is an object

each node i is a buyer with:

S_i: set of edges incident to i

t_i=1

CA instance has a solution of total value ≥ k if and only if there is an IS of size ≥ k

A solution of value k for the instance of CA with $Opt_{CA}/k \leq m^{\frac{1}{2}-\varepsilon}$ for some $\varepsilon>0$

would imply

A solution of value k for the instance of IS and hence:

$$Opt_{IS}/k = Opt_{CA}/k \leq m^{\frac{1}{2}-\varepsilon} \leq n^{1-2\varepsilon}$$

since $m \leq n^2$

How to design a truthful mechanism for the problem?

Notice that:

the (true) total value of a feasible W is:

$$\sum_i v_i(t_i,W)$$

the problem is utilitarian!

...but a VCG mechanism is not computable in polynomial time!

what can we do?

...fortunately, our problem is one parameter!

A problem is binary demand (BD) if

1. a_i's type is a single parameter $t_i \in \Re$
2. a_i's valuation is of the form:

$$v_i(t_i,o)= t_i\, w_i(o),$$

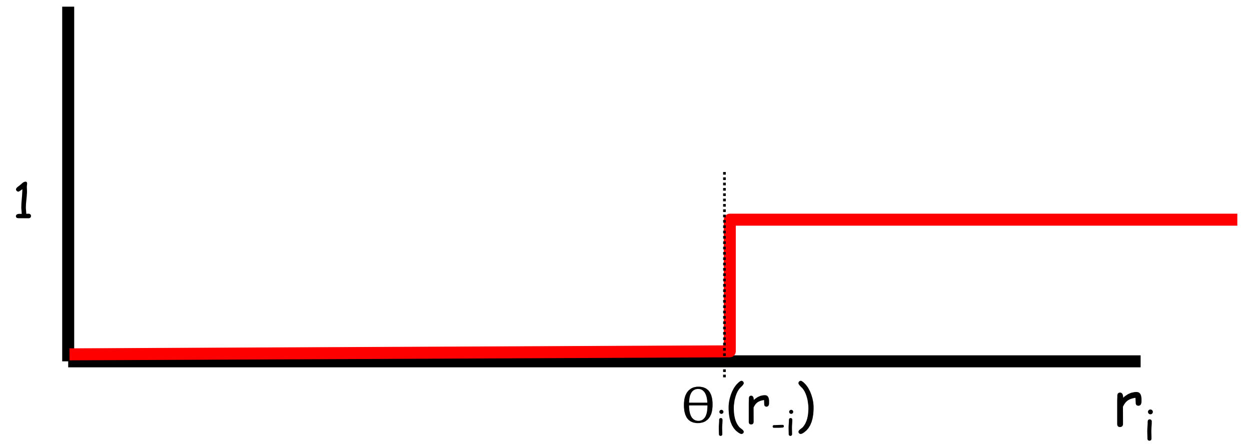
$w_i(o) \in \{0,1\}$ work load for a_i in o

when $w_i(o)=1$ we'll say that a_i is selected in o

Definition

An algorithm $g()$ for a maximization BD problem is monotone if

$\forall$ agent a_i, and for every $r_{-i}=(r_1,\ldots,r_{i-1},r_{i+1},\ldots,r_N)$, $w_i(g(r_{-i},r_i))$ is of the form:



$\theta_i(r_{-i}) \in \Re \cup \{+\infty\}$: threshold

payment from a_i is:

$p_i(r) = \theta_i(r_{-i})$

- Our goal: to design a mechanism satisfying:
  1. $g(\cdot)$ is monotone
  2. Solution returned by $g(\cdot)$ is a "good" solution, i.e. an approximated solution
  3. $g(\cdot)$ and $p(\cdot)$ computable in polynomial time

A greedy $\sqrt{m}$-approximation algorithm

1. reorder (and rename) the bids such that

   $$v_1/\sqrt{|S_1|} \geq v_2/\sqrt{|S_2|} \geq \ldots \geq v_n/\sqrt{|S_n|}$$

2. $W \leftarrow \emptyset$; $X \leftarrow \emptyset$
3. for $i=1$ to n do
   1. if $S_i \cap X = \emptyset$ then $W \leftarrow W \cup \{i\}$; $X \leftarrow X \cup S_i$
4. return W

Lemma

The algorithm $g(\,)$ is monotone

proof

It suffices to prove that, for any selected agent i, we have that i is still selected when it raises its bid

$$v_1/\sqrt{|S_1|} \geq \ldots \geq v_i/\sqrt{|S_i|} \geq \quad \ldots \quad \geq v_n/\sqrt{|S_n|}$$

Increasing v_i can only move bidder i up in the greedy order, making it easier to win

■

Computing the payments

…we have to compute for each selected bidder i its threshold value

How much can bidder i decrease its bid before being non-selected?

Computing payment p_i

Consider the greedy order without i

$$v_1/\sqrt{|S_1|} \geq \ldots \geq \cancel{v_i/\sqrt{|S_i|}} \geq \quad \ldots \quad \geq v_n/\sqrt{|S_n|}$$

index j

Use the greedy algorithm to find the smallest index j (if any) such that:

1. j is selected
2. $S_j \cap S_i \neq \emptyset$

$p_i = v_j \sqrt{|S_i|}/\sqrt{|S_j|}$

$p_i = 0$ if j doesn't exist

Lemma

Let OPT be an optimal solution for CA problem, and let W be the solution computed by the algorithm, then

$$\Sigma_{i \in OPT} v_i \leq \sqrt{m}\, \Sigma_{i \in W} v_i$$

proof

$\forall i \in W \qquad OPT_i = \{j \in OPT : j \geq i \text{ and } S_j \cap S_i \neq \emptyset\}$

since

$\bigcup_{i \in W} OPT_i = OPT$ it suffices to prove: $\displaystyle\sum_{j \in OPT_i} v_j \leq \sqrt{m}\, v_i \qquad \forall i \in W$

$$\sum_{j \in OPT} v_j \leq \sum_{i \in W} \sum_{j \in OPT_i} v_j \leq \sum_{i \in W} \sqrt{m}\, v_i \leq \sqrt{m} \sum_{i \in W} v_i$$

Lemma

Let OPT be an optimal solution for CA problem, and let W be the solution computed by the algorithm, then

$$\sum_{i \in OPT} v_i \leq \sqrt{m} \sum_{i \in W} v_i$$

proof

$\forall i \in W$ $\quad OPT_i = \{j \in OPT : j \geq i \text{ and } S_j \cap S_i \neq \emptyset\}$

since $\bigcup_{i \in W} OPT_i = OPT$ it suffices to prove: $\displaystyle\sum_{j \in OPT_i} v_j \leq \sqrt{m}\, v_i \quad \forall i \in W$

crucial observation
for greedy order we have

$$v_j \leq \frac{v_i \sqrt{|S_j|}}{\sqrt{|S_i|}} \quad \forall j \in OPT_i$$

proof

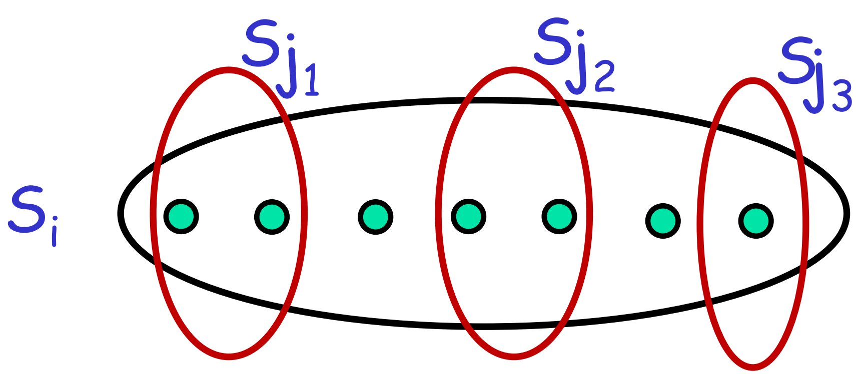
$\forall i \in W$

$$\sum_{j \in OPT_i} v_j \le \frac{v_i}{\sqrt{|S_i|}} \sum_{j \in OPT_i} \sqrt{|S_j|} \le \sqrt{m}\, v_i$$

we can bound

Cauchy-Schwarz inequality

$$\sum_{j \in OPT_i} \sqrt{|S_j|} \le \underbrace{\sqrt{|OPT_i|}}_{\le |S_i|} \sqrt{\underbrace{\sum_{j \in OPT_i} |S_j|}_{\le m}} \le \sqrt{|S_i|}\sqrt{m}$$



$OPT_i = \{j_1\ j_2\ j_3\}$

Cauchy-Schwarz inequality

$$\left(\sum_{i=1}^{n} x_i y_i\right) \leq \left(\sum_{i=1}^{n} x_i^2\right)^{1/2} \left(\sum_{i=1}^{n} y_i^2\right)^{1/2}.$$

…in our case…

$n = |OPT_i|$

$x_j = 1$

$y_j = \sqrt{|S_j|}$ for $j = 1, \dots, |OPT_i|$