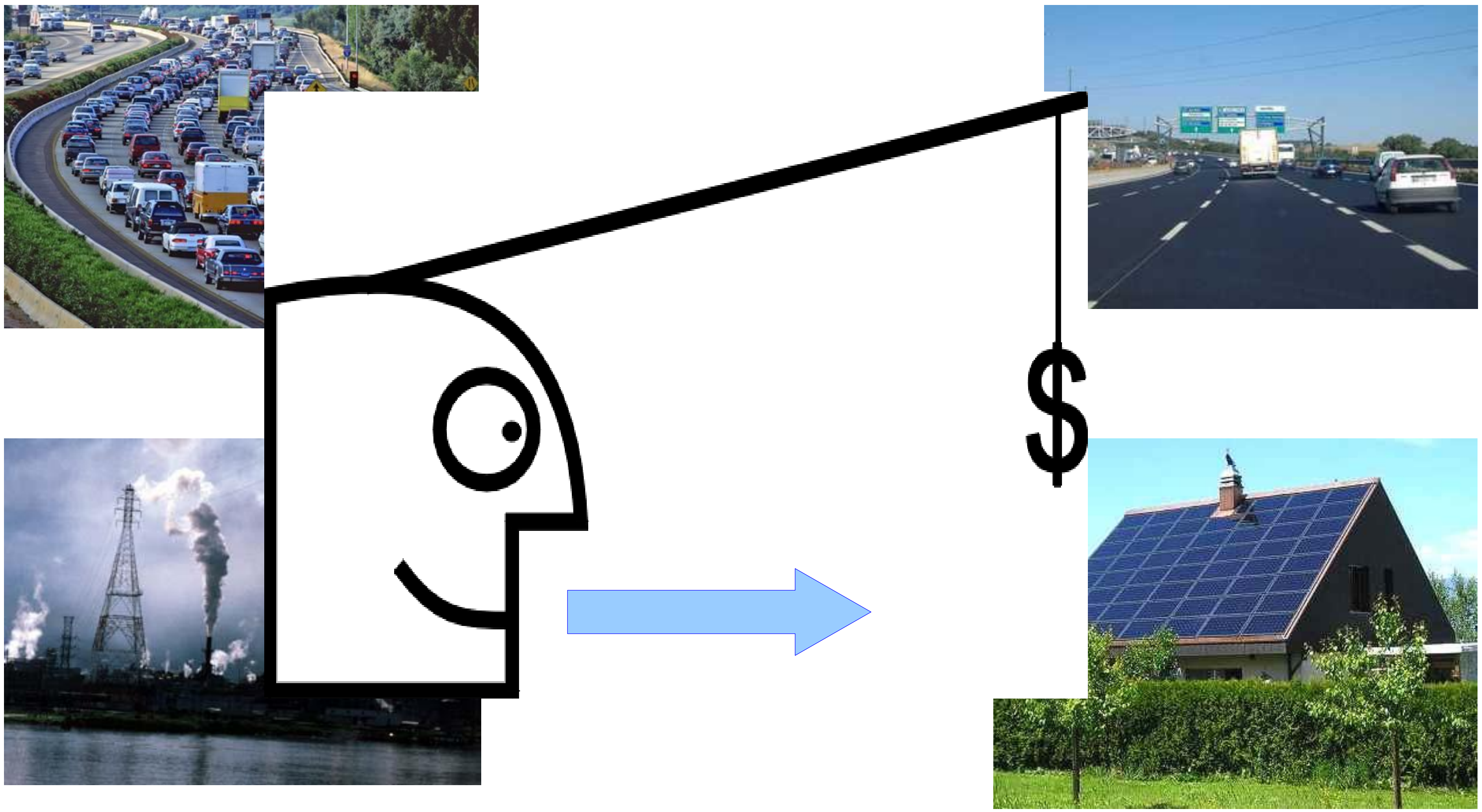
SECOND PART:
Algorithmic Mechanism Design

Mechanism Design



Find correct rules/incentives

The implementation problem

- Imagine you are a planner who develops criteria for social welfare, but you lack information about preferences of individuals. Which social-choice functions (i.e., aggregation of players' preferences w.r.t. to a certain outcome) can be implemented in such a strategic distributed system?
- Why strategic setting?
  - participants act rationally and selfishly
  - Preferences of players (i.e., their opinion about a social status) are private and can be used to manipulate the system

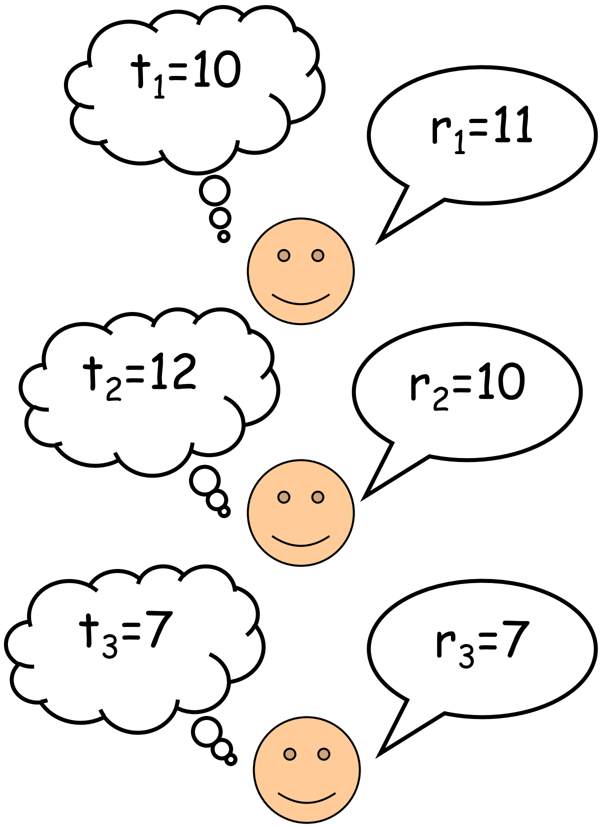
Designing a Mechanism

- Informally, designing a mechanism means to define a game in which a desired outcome must be reached (in equilibrium)
- However, games induced by mechanisms are different from games in standard form:
  - Players hold independent private values
  - The payoff matrix is a function of these types

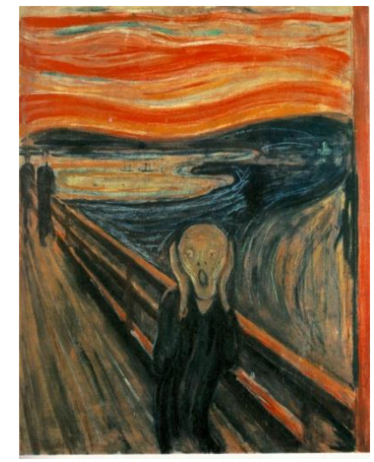
$\Rightarrow$ Games with incomplete information

An example: auctions

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



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



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 $u_i=t_i-p$

The mechanism tells to players:

(1) How the item will be allocated (i.e., who will be the winner), depending on the received bids

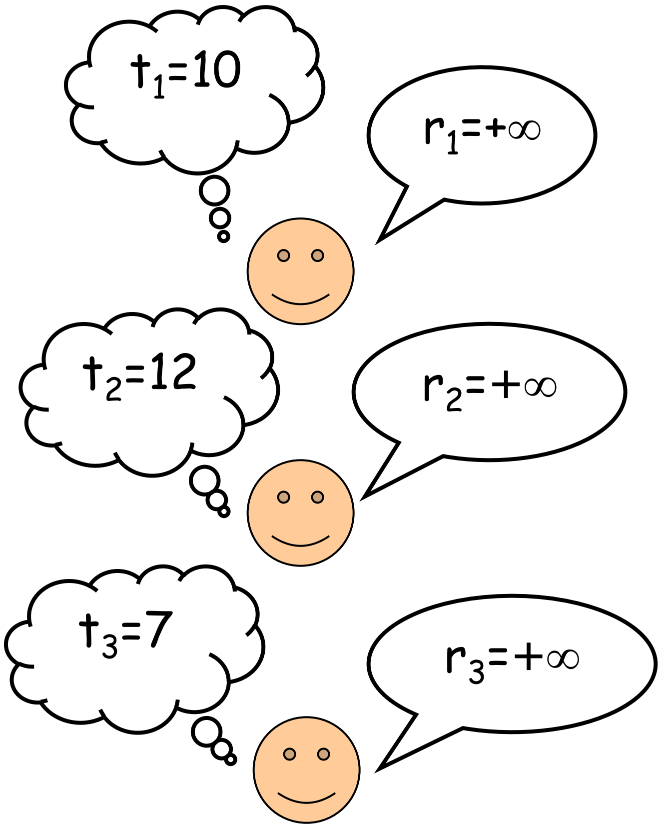
(2) The payment the winner has to return, as a function of the received bids

Mechanism degree of freedom

- The mechanism has to decide:
  - The allocation of the item
  - The payment by the winner

- …in a way that cannot be manipulated
  - the mechanism designer wants to obtain/compute a specific outcome (defined in terms of the real and private values held by the players)

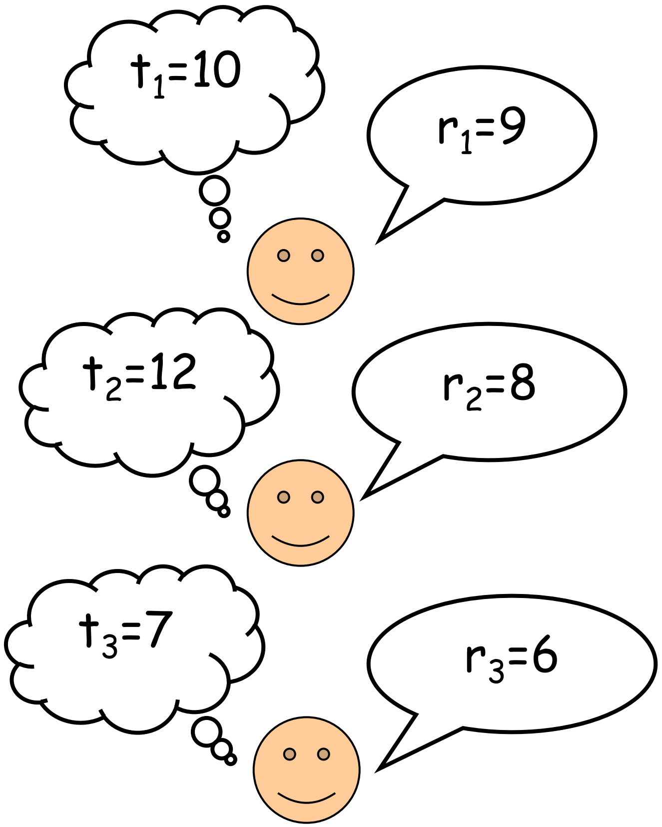
A simple mechanism: no payment



The highest bid wins
and the price of the item
is 0

...it doesn't work...

Another simple mechanism: pay your bid



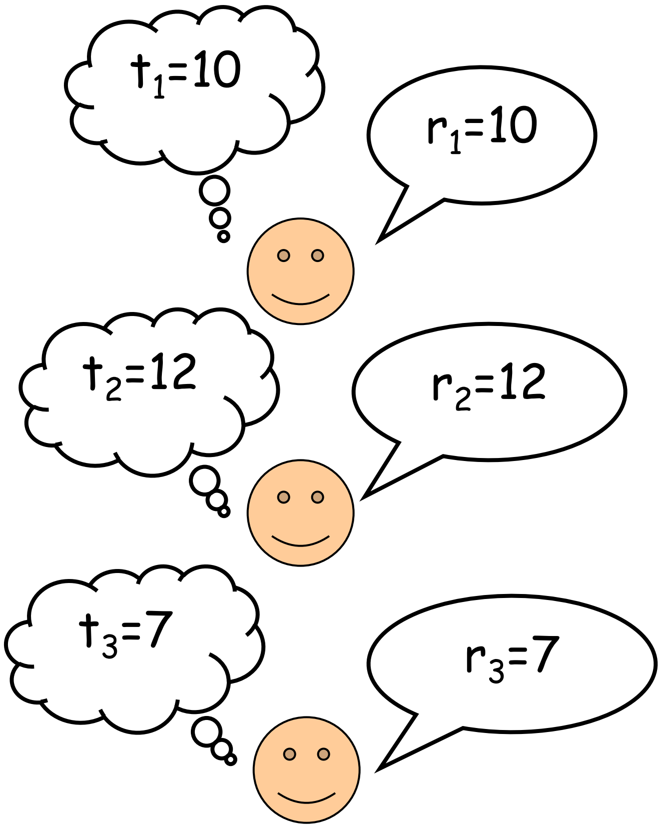
Mechanism: The highest bid wins and the winner will pay his bid

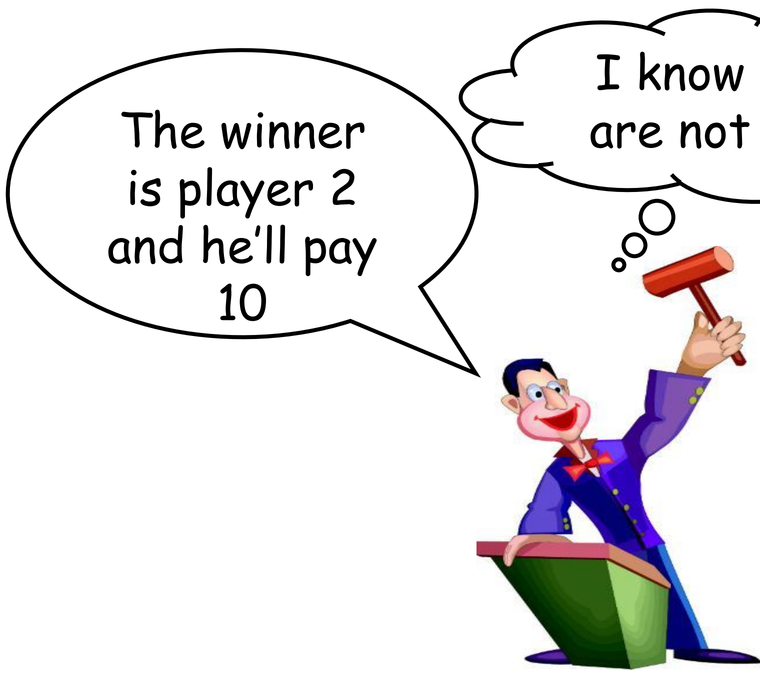
Player i will bid $r_i < t_i$ (in this way he is guaranteed not to incur a negative utility)

…and so the winner could be the wrong one…

…it doesn't work…

An elegant solution: Vickrey's second price auction





The highest bid wins and the winner will pay the second highest bid

every player has convenience to declare the truth!
(we prove it in the next slide)

Theorem

In the Vickrey auction, for every player i, $r_i=t_i$ is a dominant strategy

proof Fix i and t_i, and look at strategies for player i. Let **R**= $\max_{j\neq i} \{r_j\}$

Case $t_i \geq$ **R** (observe that **R** is unknown to player i)

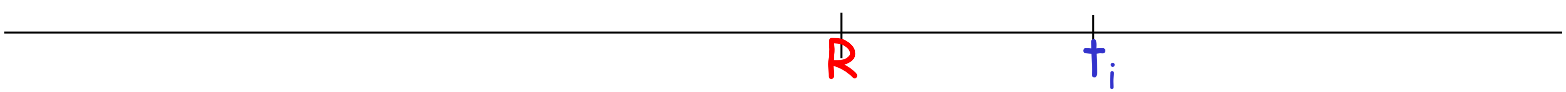
declaring $r_i=t_i$ gives utility $u_i= t_i-$**R** ≥ 0

(player wins if $t_i >$ **R**, while if $t_i =$ **R** then player can either win or lose, depending on the tie-breaking rule, but its utility would be 0)

declaring any $r_i >$ **R**, $r_i\neq t_i$, yields again utility $u_i= t_i-$**R** ≥ 0

(player wins)

declaring any $r_i <$ **R** yields $u_i=0$ (player loses)



Theorem

In the Vickrey auction, for every player i, $r_i=t_i$ is a dominant strategy

proof Fix i and t_i, and look at strategies for player i. Let $R= \max_{j\neq i} \{r_j\}$

Case $t_i \geq R$ (observe that R is unknown to player i)

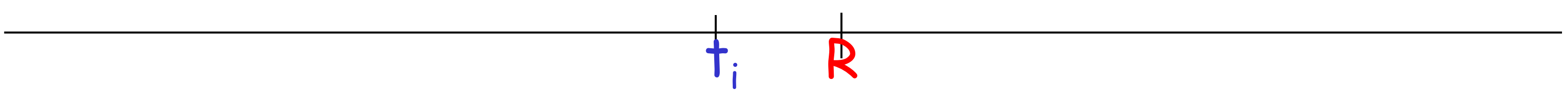
declaring $r_i=t_i$ gives utility $u_i= t_i-R \geq 0$

(player wins if $t_i > R$, while if $t_i = R$ then player can either win or lose, depending on the tie-breaking rule, but its utility would be 0)

declaring any $r_i > R$, $r_i\neq t_i$, yields again utility $u_i= t_i-R \geq 0$

(player wins)

declaring any $r_i < R$ yields $u_i=0$ (player loses)



Case $t_i < R$

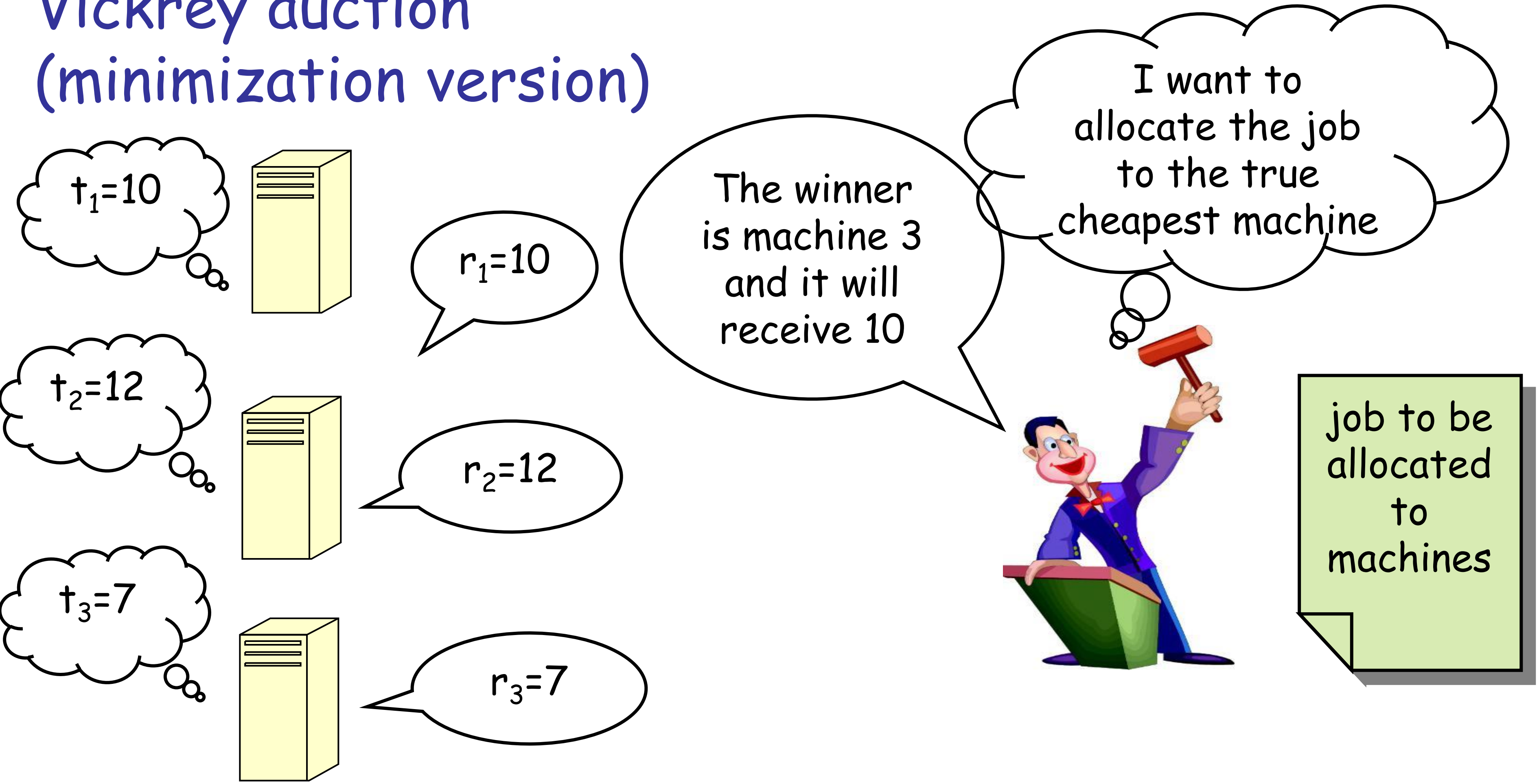
declaring $r_i=t_i$ yields utility $u_i= 0$ (player loses)

declaring any $r_i < R$, $r_i\neq t_i$, yields again utility $u_i= 0$ (player loses)

declaring any $r_i > R$ yields $u_i= t_i-R < 0$ (player wins)

$\Rightarrow$ In all the cases, reporting a false type produces a not better utility, and so telling the truth is a dominant strategy! ■

Vickrey auction (minimization version)



The cheapest bid wins and the winner will get the second cheapest bid

t_i: cost incurred by i if i does the job

if machine i is selected and receives a payment of p its utility is p-t_i

Mechanism Design Problem: ingredients (1/2)

- N agents; each agent has some **private** information $t_i \in T_i$ (actually, the **only** private info) called **type**
- A set of **feasible outcomes** F
- For each vector of types $t=(t_1, t_2, \ldots, t_N)$, a **social-choice function** $f(t) \in F$ specifies an output that should be implemented (the problem is that types are unknown…)
- Each agent has a **strategy space** S_i and performs a strategic action; we restrict ourself to *direct revelation mechanisms*, in which the action is reporting a value r_i from the type space (with possibly $r_i \neq t_i$), i.e., $S_i = T_i$

Example: the Vickrey Auction

- The set of feasible outcomes is given by all the bidders
- The social-choice function is to allocate to the bidder with lowest **true cost**:

$$f(t) = \arg\min_i (t_1, t_2, \ldots, t_N)$$

Mechanism Design Problem: ingredients (2/2)

- For each feasible outcome $x \in F$, each agent makes a **valuation** $v_i(t_i, x)$ (in terms of some common currency), expressing its preference about that output
- For each reported vector r, each agent receives a **payment** $p_i(r)$ in terms of the common currency; payments are used by the system to incentive agents to be collaborative. Then, the **utility** of the agent if the outcome for r is x(r) will be:

$$u_i(t_i, x(r)) = p_i(r) - v_i(t_i, x(r))$$

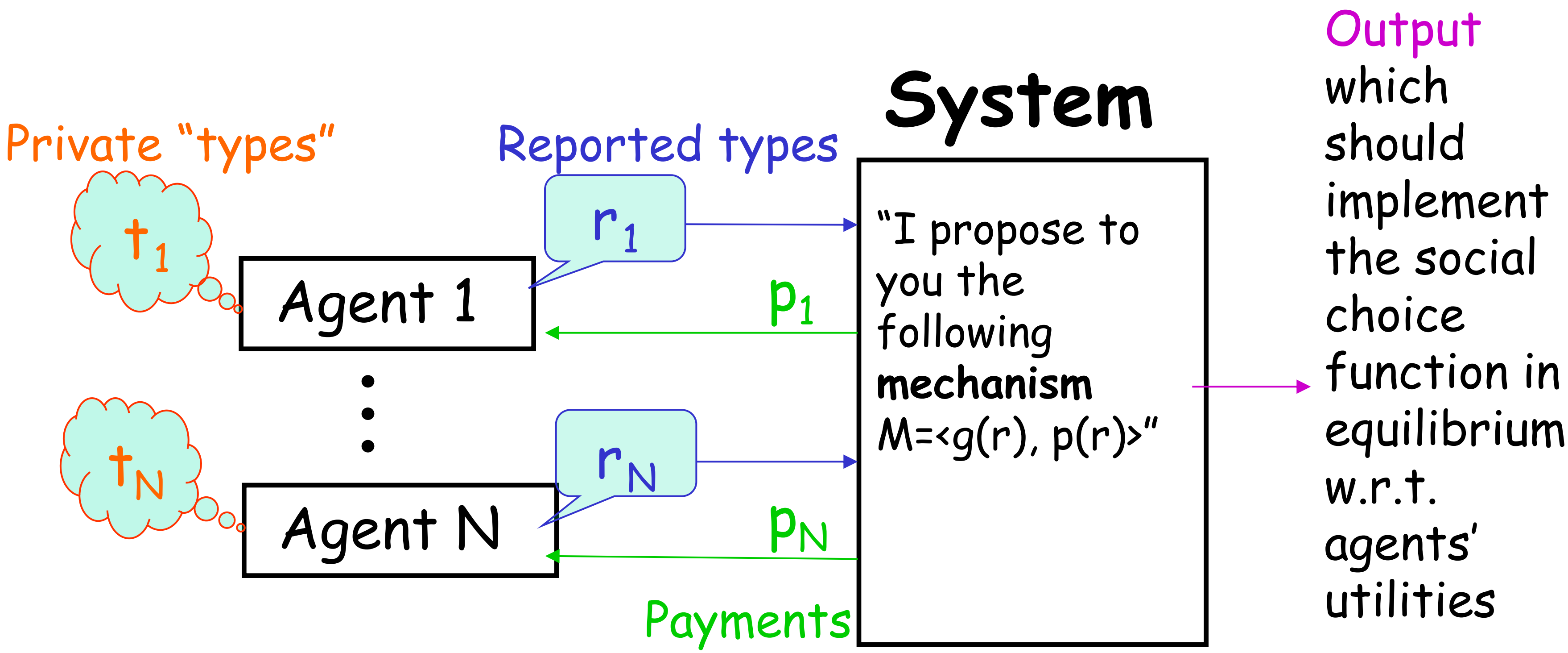
Mechanism Design Problem: the goal

Implement (according to a given equilibrium concept) the social-choice function, i.e., provide a **mechanism** $M=\langle g(r), p(r)\rangle$, where:

- $g(r)$ is an **algorithm** which computes an outcome $x=g(r)$ as a function of the reported types r
- $p(r)$ is a **payment scheme** specifying a payment (to each agent) w.r.t. the reported types r

such that $x=g(r)=f(t)$ is provided in equilibrium w.r.t. to the utilities of the agents.

Mechanism Design: a picture



Each agent reports strategically to maximize its utility…

…which depends (also) on the payment…

…which is a function of the reported types!

Implementation with dominant strategies

Def.: A mechanism $M=\langle g(),p()\rangle$ is an *implementation with dominant strategies* if there exists a reported type vector $r^*=(r_1^*, r_2^*, \ldots, r_N^*)$ such that $f(t)=g(r^*)$ in dominant strategy equilibrium, i.e., for each agent i and for each reported type vector $r=(r_1, r_2, \ldots, r_N)$, it holds:

$$u_i(t_i,(r_{-i},r_i^*)) \geq u_i(t_i,(r_{-i},r_i))$$

Strategy-Proof Mechanisms

- If *truth telling* is the dominant strategy in a mechanism then the mechanism is called *Strategy-Proof* or *truthful* or *incentive compatible*
  - ⇒ $r^*=t$.
  - ⇒ Agents report their true types instead of strategically manipulating it
  - ⇒ The algorithm of the mechanism runs on the true input

Truthful Mechanism Design: **Economics Issues**

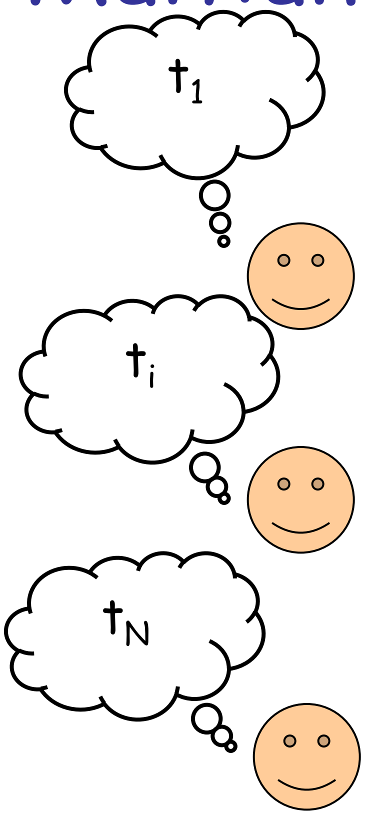
QUESTION: How to design a truthful mechanism? Or, in other words:

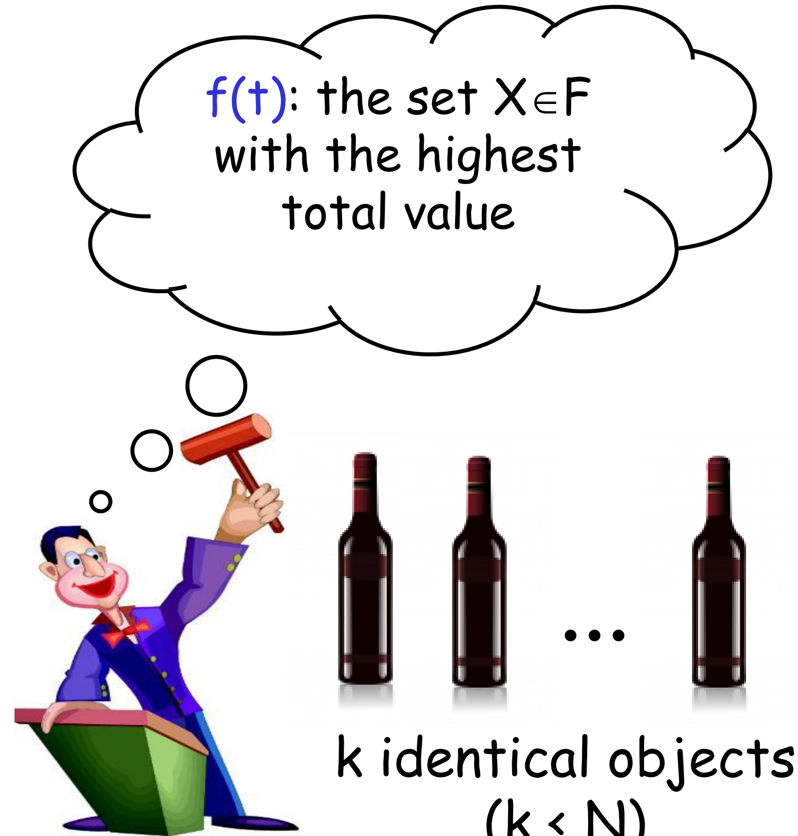
1. How to design g(r), and
2. How to define the payment scheme

in such a way that the underlying social-choice function is implemented truthfully? Under which conditions can this be done?

Some examples

Multiunit auction





k identical objects
(k < N)

Each of N players wants an object

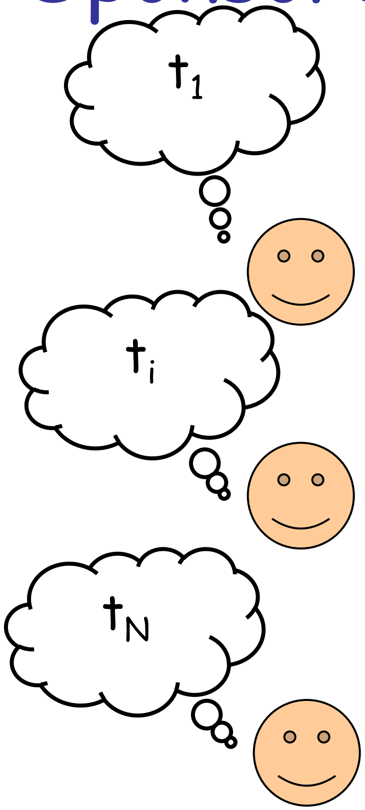
t_i: value player i is willing to pay

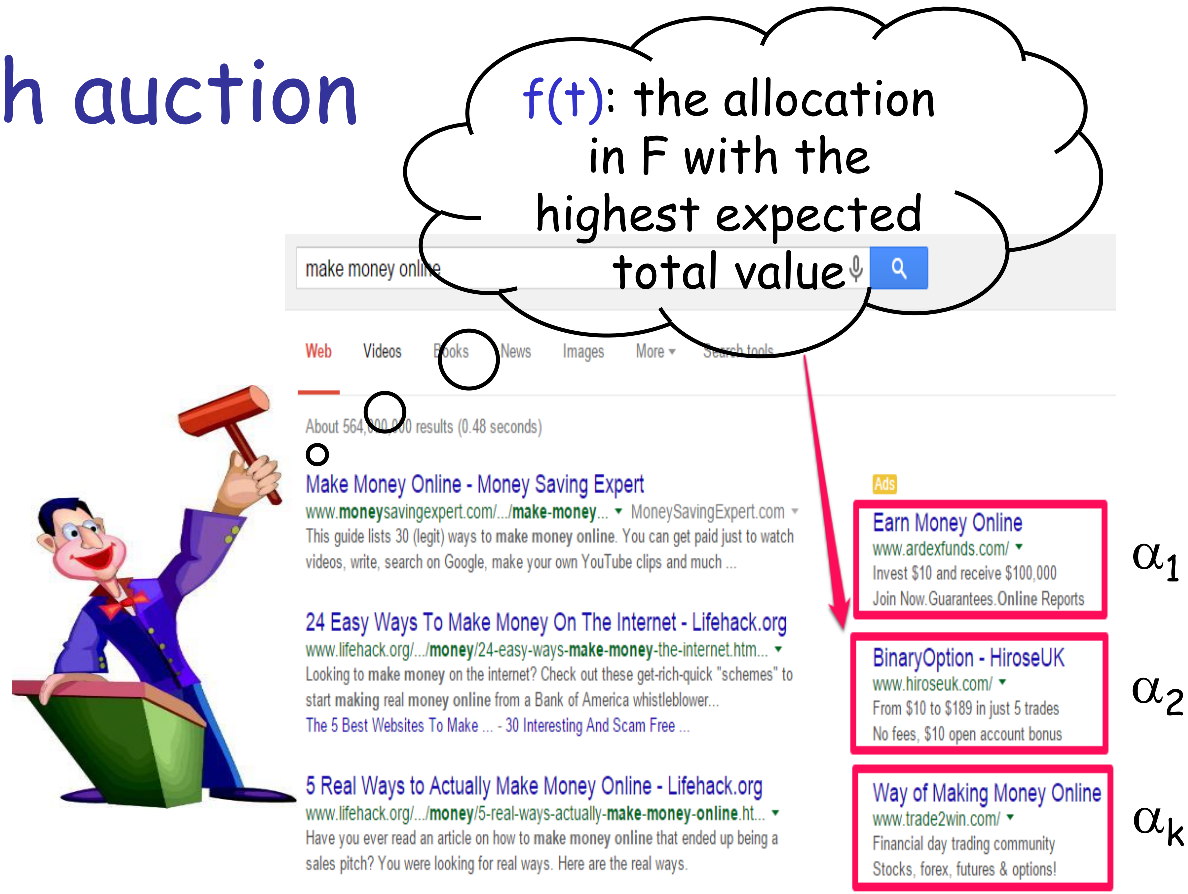
if player i gets an object at price p
his utility is $u_i = t_i - p$

the mechanism decides
the set of k winners and the
corresponding payments

$F = \{ X \subseteq \{1,\dots,N\} : |X| = k \}$

Sponsored search auction





k slots

players want a slot (higher is better)

α_j : prob user clicks on slot j

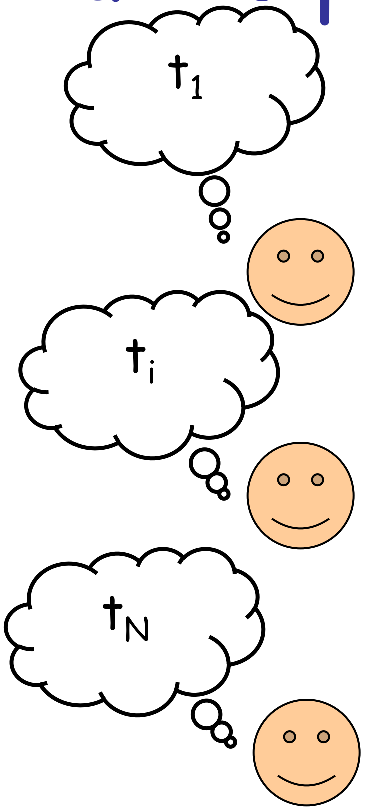
t_i: player i's value per click

the mechanism decides the k winners and the corresponding payments

if player i gets slot j at price p
his (expected) utility is $u_i = \alpha_j(t_i - p)$

$F = \{ (x_1,\dots,x_k) : x_i \in \{1,\dots,N\} \}$

Public project



t_i: value of the bridge for citizen i

if the bridge is built and citizen i has to pay p_i his utility is $u_i=t_i-p_i$

f(t): build only if $\Sigma_i t_i > C$

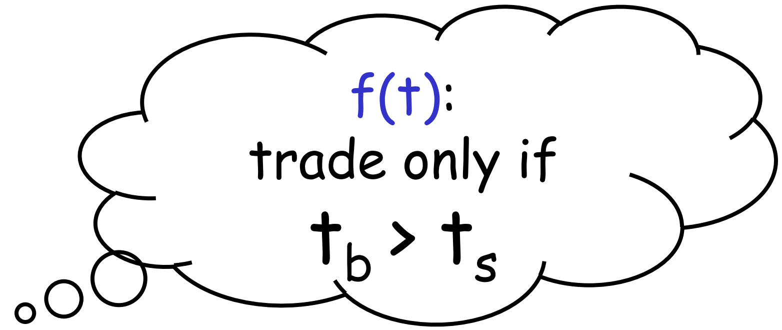
to build or not to build?

C: cost of the bridge

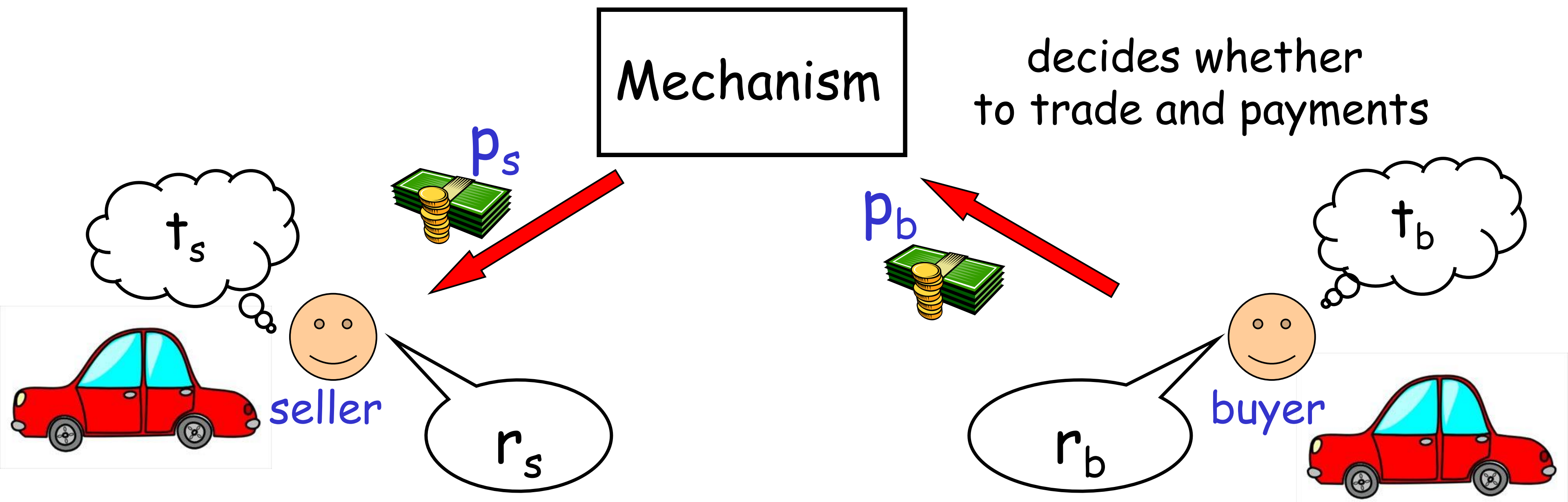
the mechanism decides whether to build and the payments from citizens

F={build, not-build}

Bilateral trade



F={trade, no-trade}



t_s: value of the object

t_b: value of the object

if trade
seller's utility:
$p_s - t_s$

if trade
buyer's utility:
$t_b - p_b$

Buying a path in a network

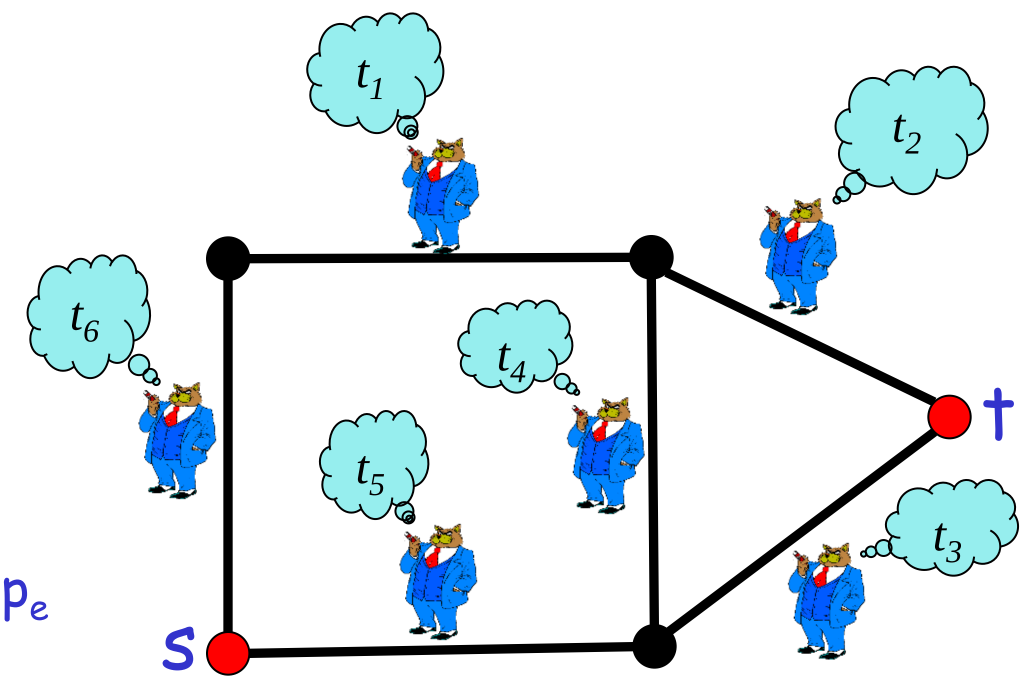
$f(t)$: a shortest path w.r.t. the true edge costs

F: set of all paths between s and t

Mechanism

decides the path and the payments

t_e: cost of edge e



if edge e is selected and receives a payment of p_e

e's utility:

$p_e - t_e$

How to design truthful mechanisms?

Some remarks

- we'll describe results for minimization problems (maximization problems are similar)
- We have:
  - for each $x \in F$, valuation function $v_i(t_i, x)$ represents a cost incurred by player i in the solution x
  - the social function f(t) maps the type vector t into a solution x which minimizes some measure of x
  - payments are from the mechanism to agents

- **Utilitarian Problems:** *A* problem is *utilitarian* if its objective function is such that $f(t) = \arg\min_{x \in F} \sum_i v_i(t_i, x)$

notice: the auction problem is utilitarian

…for utilitarian problems there is a class of truthful mechanisms…

Vickrey-Clarke-Groves (VCG) Mechanisms

- A VCG-mechanism is (the only) strategy-proof mechanism for **utilitarian** problems:
  - Algorithm $g(r)$ computes:

$$x = \arg\min_{y \in F} \sum_i v_i(r_i, y)$$

  - Payment function for player i:

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

where $h_i(r_{-i})$ is an arbitrary function of the reported types of players other than player i.

- What about **non-utilitarian** problems? General strategy-proof mechanisms are known only when the type is a single parameter. (We'll see.)

Theorem

VCG-mechanisms are truthful for utilitarian problems

proof

Fix i, r_{-i}, t_i. Let $\check{r}=(r_{-i},t_i)$ and consider a strategy $r_i \neq t_i$

$x=g(r_{-i},t_i) = g(\check{r})$ $\qquad$ $x'=g(r_{-i},r_i)$

$u_i(t_i, (r_{-i},t_i)) = [h_i(r_{-i}) - \Sigma_{j\neq i} v_j(r_j,x)] - v_i(t_i,x) = h_i(r_{-i}) - \Sigma_j v_j(\check{r}_j,x)$

$u_i(t_i, (r_{-i},r_i)) = [h_i(r_{-i}) - \Sigma_{j\neq i} v_j(r_j,x')] - v_i(t_i,x') = h_i(r_{-i}) - \Sigma_j v_j(\check{r}_j,x')$

but x is an optimal solution w.r.t. $\check{r} = (r_{-i},t_i)$, i.e.,

$$x = \arg\min_{y\in F} \sum_i v_i(\check{r},y)$$

➡ $\Sigma_j v_j(\check{r}_j,x) \leq \Sigma_j v_j(\check{r}_j,x')$ ➡ $u_i(t_i, (r_{-i},t_i)) \geq u_i(t_i, (r_{-i},r_i))$. ■

How to define $h_i(\mathbf{r}_{-i})$?

notice: not all functions make sense

what happens if we set $h_i(r_{-i})=0$
in the Vickrey auction?

The Clarke payments

solution minimizing the sum of valuations when i doesn't play

- This is a special VCG-mechanism in which

$$h_i(r_{-i}) = \sum_{j \neq i} v_j(r_j, g(r_{-i}))$$

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

- With Clarke payments, one can prove that agents' utility are always non-negative

$\Rightarrow$ agents are interested in playing the game

Clarke mechanism for the Vickrey auction (minimization version)

- The VCG-mechanism is:
  - $x=g(r):=\arg\min_{x\in F}\sum_i v_i(r_i,x)$
    - allocate to the bidder with **lowest reported cost**
  - $p_i = \sum_{j\neq i} v_j(r_j,g(r_{-i})) - \sum_{j\neq i} v_j(r_j,x)$

…pay the winner the second lowest offer,
and pay 0 the losers

Mechanism Design: **Algorithmic Issues**

QUESTION: What is the time complexity of the mechanism? Or, in other words:

- What is the time complexity of $g(r)$?
- What is the time complexity to calculate the N payment functions?
- What does it happen if it is **NP-hard** to compute the underlying social-choice function?

Algorithmic mechanism design for graph problems

- Following the Internet model, we assume that each agent owns a **single edge** of a graph G=(V,E), and establishes the **cost** for using it

$\Rightarrow$ The agent's type is the **true weight** of the edge

- Classic optimization problems on G become mechanism design optimization problems!
- Many basic network design problems have been faced: shortest path (SP), single-source shortest paths tree (SPT), minimum spanning tree (MST), minimum Steiner tree, and many others

Summary of main results

	Centralized algorithm	Selfish-edge mechanism
SP	$O(m+n \log n)$	$O(m+n \log n)$
SPT	$O(m+n \log n)$	$O(m+n \log n)$
MST	$O(m\ \alpha(m,n))$	$O(m\ \alpha(m,n))$

$\Rightarrow$ For all these basic problems, the time complexity of the mechanism equals that of the canonical centralized algorithm!