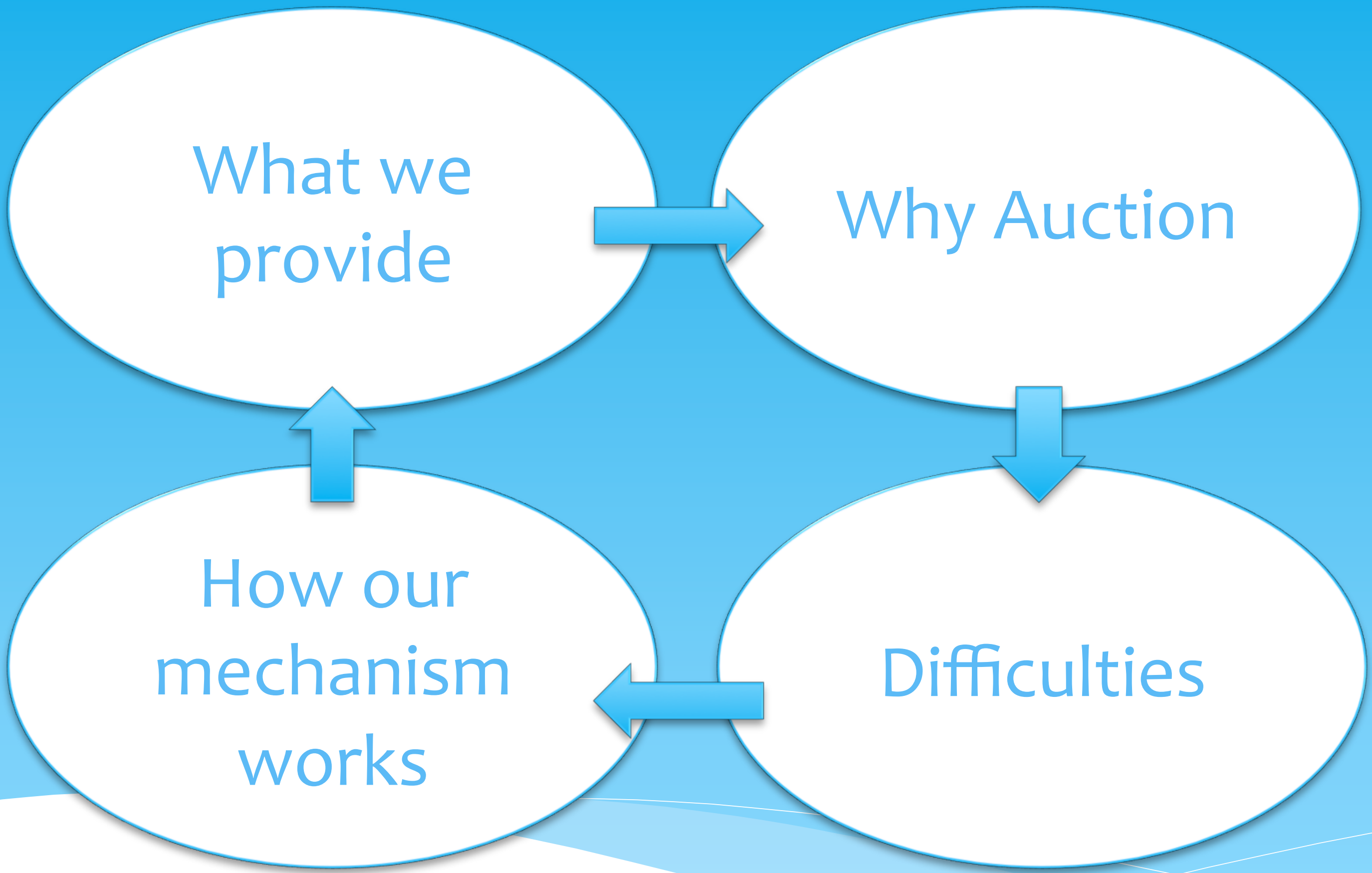
Truthful Auctions for On-demand Cloud Resource Provisioning

Xiaoxi Zhang

Jan. 28, 2015

We will go through:



○ What WE provide
What THEY demand

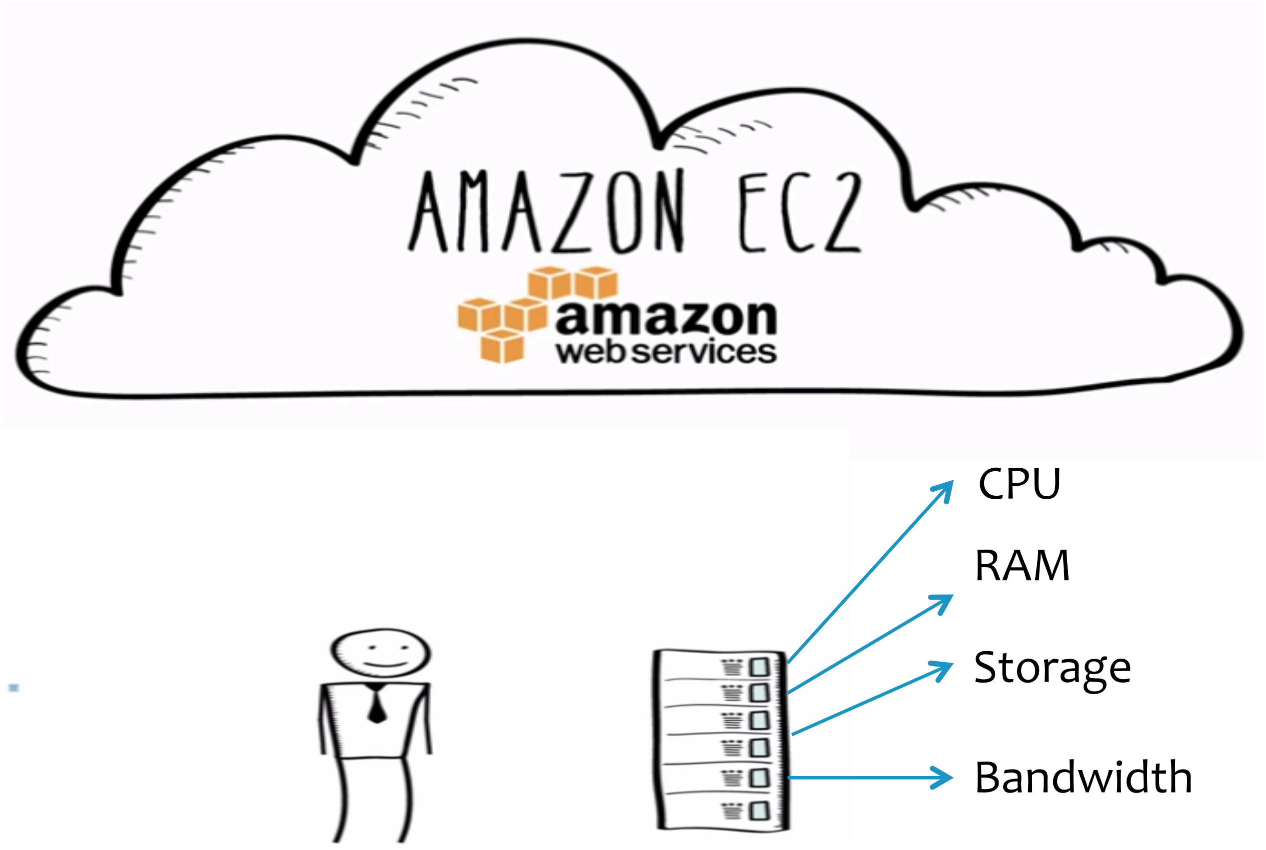
* A user may have resource demand for some task

○ What WE provide What THEY demand

* We are the Cloud providers to own a pool of resources distributed in multiple data centers

○ What WE provide
What THEY demand

* The user's resource demand consist of various types of resource

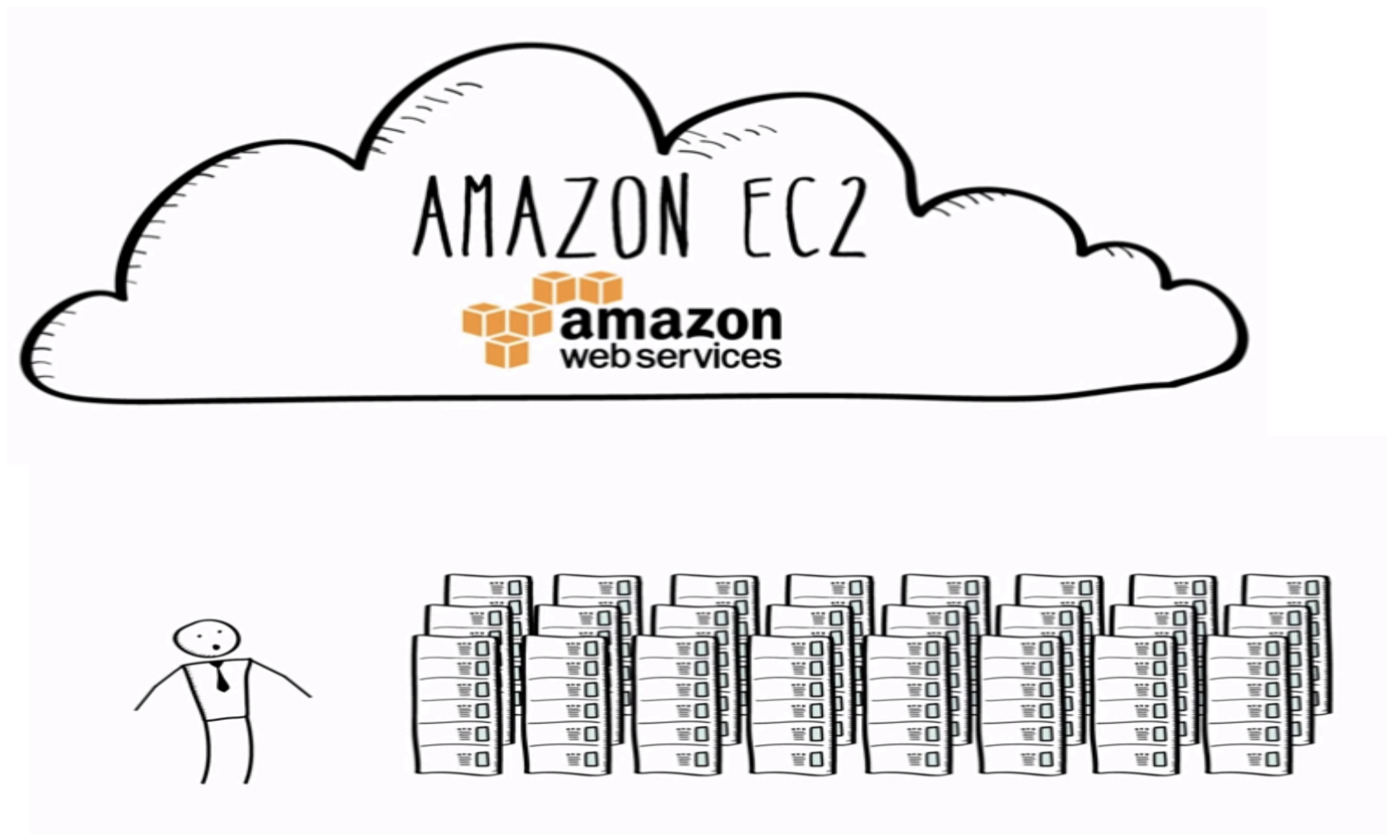


○ What WE provide What THEY demand

* Sometimes, it is tiny.

○ What WE provide
What THEY demand

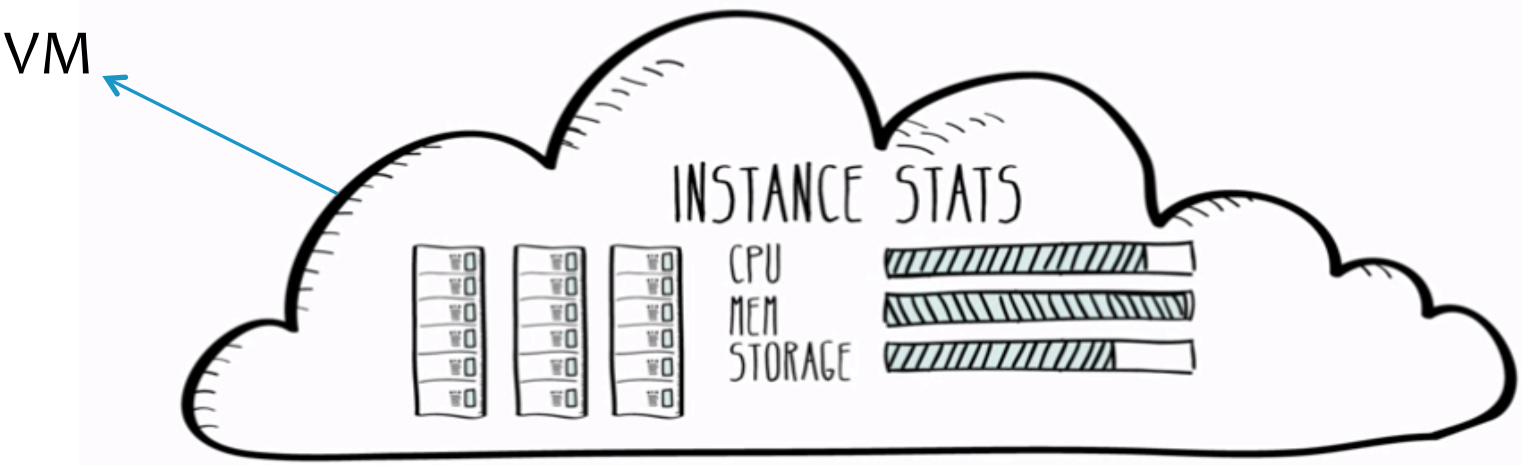
* Sometimes, it is huge.



○ What WE provide What THEY demand

* And most of the time, there are many users.

○ What WE provide
What THEY demand



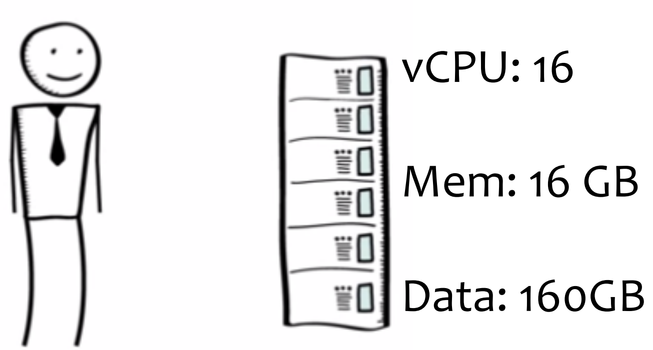
* Virtualization technology packs resources into VMs

○ What WE provide What THEY demand

* Cloud Providers charge users according to their actual usage.

What WE provide
What THEY demand

* Pre-determined types



Model	vCPU	Mem (GiB)	SSD Storage (GB)
c3.large	2	3.75	2 x 16
c3.xlarge	4	7.5	2 x 40
c3.2xlarge	8	15	2 x 80
c3.4xlarge	16	30	2 x 160
c3.8xlarge	32	60	2 x 320

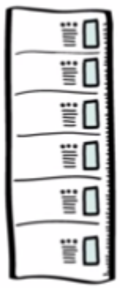
What WE provide
What THEY demand

* Pre-determined types

Model	vCPU	Mem (GiB)	SSD Storage (GB)
c3.large	2	3.75	2 x 16
c3.xlarge	4	7.5	2 x 40
c3.2xlarge	8	15	2 x 80
c3.4xlarge	16	30	2 x 160
c3.8xlarge	32	60	2 x 320

○ What WE provide
What THEY demand

* Pre-determined types



vCPU: 16

Mem: 16 GB

Data: 160GB

Model	vCPU	Mem (GiB)	SSD Storage (GB)
c3.large	2	3.75	2 x 16
c3.xlarge	4	7.5	2 x 40
c3.2xlarge	8	15	2 x 80
c3.4xlarge	16	30	2 x 160
c3.8xlarge	32	60	2 x 320

○ Current Development

* More VM Types **(Amazon EC2: 7 categories, 23 types)**

* Fixed pricing

c4.large	2	8	3.75	EBS Only	$0.116 per Hour
c4.xlarge	4	16	7.5	EBS Only	$0.232 per Hour
c4.2xlarge	8	31	15	EBS Only	$0.464 per Hour
c4.4xlarge	16	62	30	EBS Only	$0.928 per Hour
c4.8xlarge	36	132	60	EBS Only	$1.856 per Hour
c3.large	2	7	3.75	2 x 16 SSD	$0.105 per Hour
c3.xlarge	4	14	7.5	2 x 40 SSD	$0.210 per Hour
c3.2xlarge	8	28	15	2 x 80 SSD	$0.420 per Hour
c3.4xlarge	16	55	30	2 x 160 SSD	$0.840 per Hour
c3.8xlarge	32	108	60	2 x 320 SSD	$1.680 per Hour

The problems

- How many VM Types do we need?

  -- Difficult to estimate since users' demands are fluctuating

- Under fixed pricing for each type of VMs, it is impossible to:

  -- come up with the appropriate prices, *i.e.*, priorities ;
  -- maximize the social welfare .

The problems

* How many VM Types do we need?

Ideally, the users determine his own VM type.

* Fixed pricing for each VM type is not economically efficient

The providers hope to price according to the current users (demand & willingness to pay).

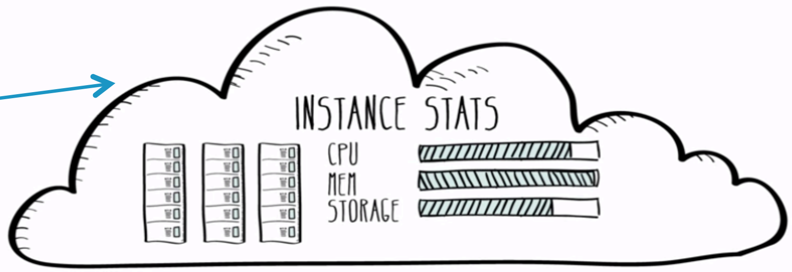
That is why we need auction.

Why auction

One user, one bid set.

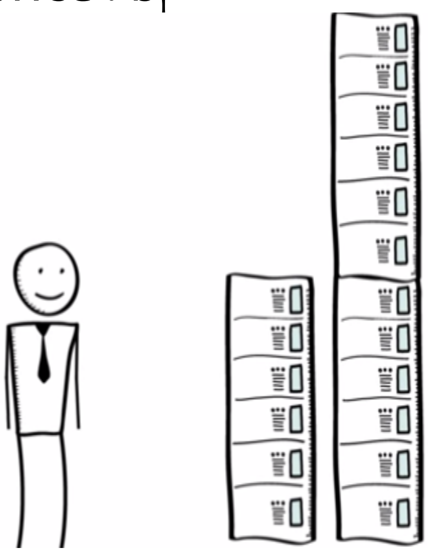
$B = \{B_1, B_2, \ldots, B_i, \ldots\}$,

at most one bid of each B can win

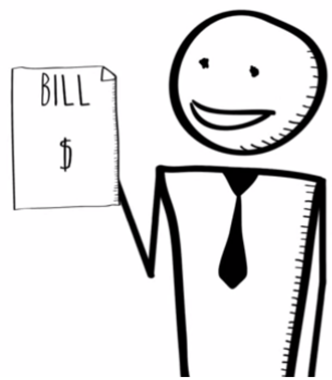


$B_i = \{ (d_1, d_2, \ldots, d_K),\ b_i \}$

Bidding price : b_i



If some B_i wins,
resource is allocated,
He pays for it.

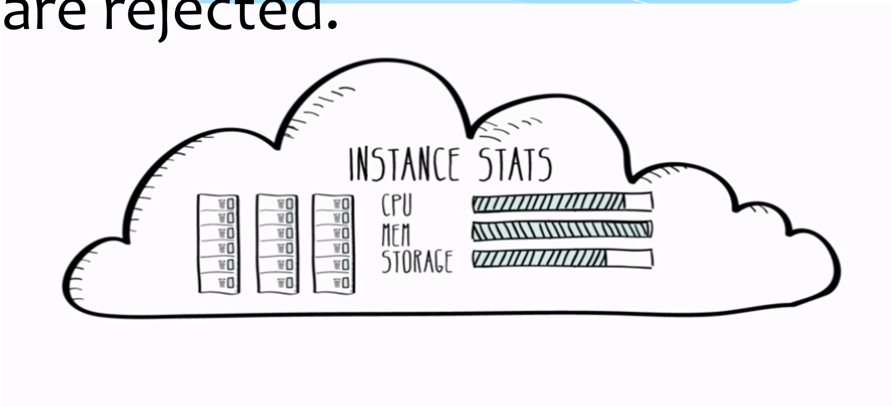


Payment $\neq$ b_i

Why auction

Resource is limited, some requests are rejected.

-- Goal: social welfare maximization



What if the bidding price is fake?

-- An appropriate pricing scheme guarantees truthfulness.

-- Also, the total payment is the provider's revenue

○ Difficulties

Each type of resource has a capacity.
Who can win?

-- Goal: social welfare maximization

- ❖ Allocation is An NP-hard combinatorial optimization problem

What if the bidding price is fate?

-- Pricing scheme guarantees truthfulness in valuation

- ❖ VCG requires an exact optimal allocation

○ Existing Mechanisms

* Amazon spot instances

Lack of truthfulness and service guarantees

* S. Zaman et al. (*CloudCom*'11)

"Combinatorial Auction-Based Dynamic VM Provisioning and Allocation in Clouds,"

Truthful

Lack of approximation guarantee

○ Existing Mechanisms

* Qian Wang et. al (INFOCOM’12)

“When Cloud Meets eBay: Towards Effective Pricing for Cloud Computing,”

Greedy allocation + well-design payment (critical value)

Truthful

Large approximation ratio

○ Existing Mechanisms

* Hong Zhang et al. (*INFOCOM*'13)

"A Framework for Truthful Online Auctions in Cloud Computing with Heterogeneous User Demands,"

Only consider a single type of resource

○ Existing Mechanisms

* Linquan Zhang et al. (*INFOCOM, 2014*)

"Dynamic Resource Provisioning in Cloud Computing: A Randomized Auction Approach,"

Offline

Approximation algorithm for allocation

LP decomposition

* Weijie Shi et al. (*SIGMETRICS'14*)

"An Online Auction Framework for Dynamic Resource Provisioning in Cloud Computing,"

Zhang's one round

Online: multiple rounds

Budget

○ Our work

* A truthful (1-epsilon)-Optimal Mechanism for On-demand Cloud Resource Provisioning

* Online Auctions in IaaS Clouds: Welfare and Profit Maximization with Server Costs

Our work

Three major differences:

- Offline setting vs online setting

Our work

Three major differences:

- Offline setting vs online setting
- Maximization goals:

1. Social welfare
2. Social welfare and/or revenue of the provider

- Social welfare :

1. (the total bidding price – ~~payment) + payment~~
2. (the total bidding price – ~~payment)~~ + (payment – operation cost)

Our Mechanism in the first work

* **Model:**

N users: Each user submit as many bids as he wishes with at most one accepted

K types of resource

D data centers (capacities known).

Achieves:

* Truthfulness in expectation
* (1-ε)-optimal of social welfare
* Polynomial expected running time

Our Mechanism in the first work

* Model:

N users.

Each user submit as many bids as he wishes with at most one accepted.

K types of resource, D data centers, Capacities known.

$$\text{maximize} \quad \sum_{n \in [N]} \sum_{i \in \mathcal{B}_n} b_i x_i \tag{1}$$

subject to:

$$\sum_{n \in [N]} \sum_{i \in \mathcal{B}_n} x_i R^i_{k,d} \leq c_{k,d}, \quad \forall k \in [K], \forall d \in [D], \tag{1a}$$

$$\sum_{i \in \mathcal{B}_n} x_i \leq 1, \quad \forall n \in [N], \tag{1b}$$

$$x_i \in \{0, 1\}, \quad \forall i \in \mathcal{B}_n, \forall n \in [N].$$

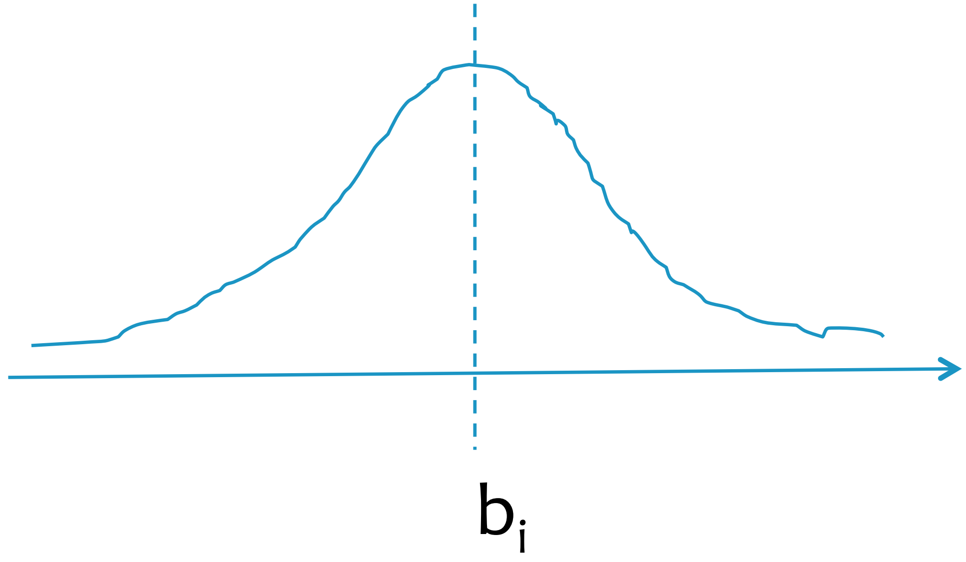
○ High-level Idea

- Randomly Transform O to PO (perturbed b_i → \bar{bi})
- Exactly solve PO (x^p)
- Random $y^{\varepsilon} \sim \Omega(x^p)$
- Random VCG

○ Premise

Polytime: The worst case of PO is perturbed

Smooth analysis



High-level Idea

Our perturbation is well-designed

(1-ε)-approximation: **Smooth perturbation**

Perturbation matrix: P

One rule to rule them all

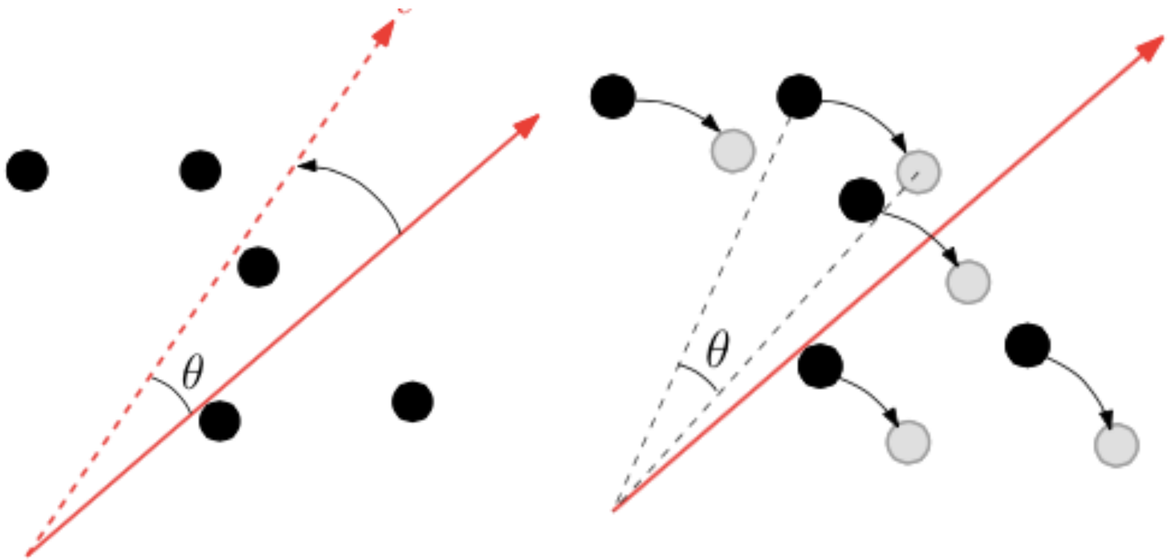
perturb

sample

$$\vec{\bar{b}} = P\vec{b}. \quad \Longleftrightarrow \quad E[\vec{y}^{\epsilon}] = P^T \vec{x}^p$$

○ Illustration

- The perturbation of obj ⟷ The perturbation of solution
- Dughmi (FOCS'10)



$$\vec{\vec{b}} = P\vec{b}.$$

core

$$E[\vec{y}^{\epsilon}] = P^T \vec{x}^p$$

Summary

* Perturbation-based Randomized Allocation
  - Randomly (i.i.d.) perturb each b_i
  - Exactly solve (x^p) perturbed optimization
  - x^p guarantees POPT ≥ (1-ε) OPT
  - Randomly sample $y^\varepsilon \sim \Omega(x^p)$
  - y^ε achieve (1-ε)-approximation.
  - Randomized VCG

$$\vec{\tilde{b}} = P\vec{b}.$$

core

$$E[\vec{y}^{\epsilon}] = P^T \vec{x}^p$$

○ Our Mechanism in the first work

* Randomized VCG-like payment
  - -- Calculate y^ε
  - -- Payment rule: opportunity cost according to

$$p_i(\vec{y}^\epsilon) = \vec{b}^T_{-i}\vec{y}^\epsilon_{-i} - (\vec{b}^T\vec{y}^\epsilon - b_i y_i^\epsilon), \forall i \in [L].$$

  - -- Guarantees truthfulness

Our Mechanism in the second work

- Model：

I bids, R types of resource, S servers

Capacity: C_{rs}

Bid i: t_i , t_i^- , t_i^+ , b_i , $d_{ir}(t)$

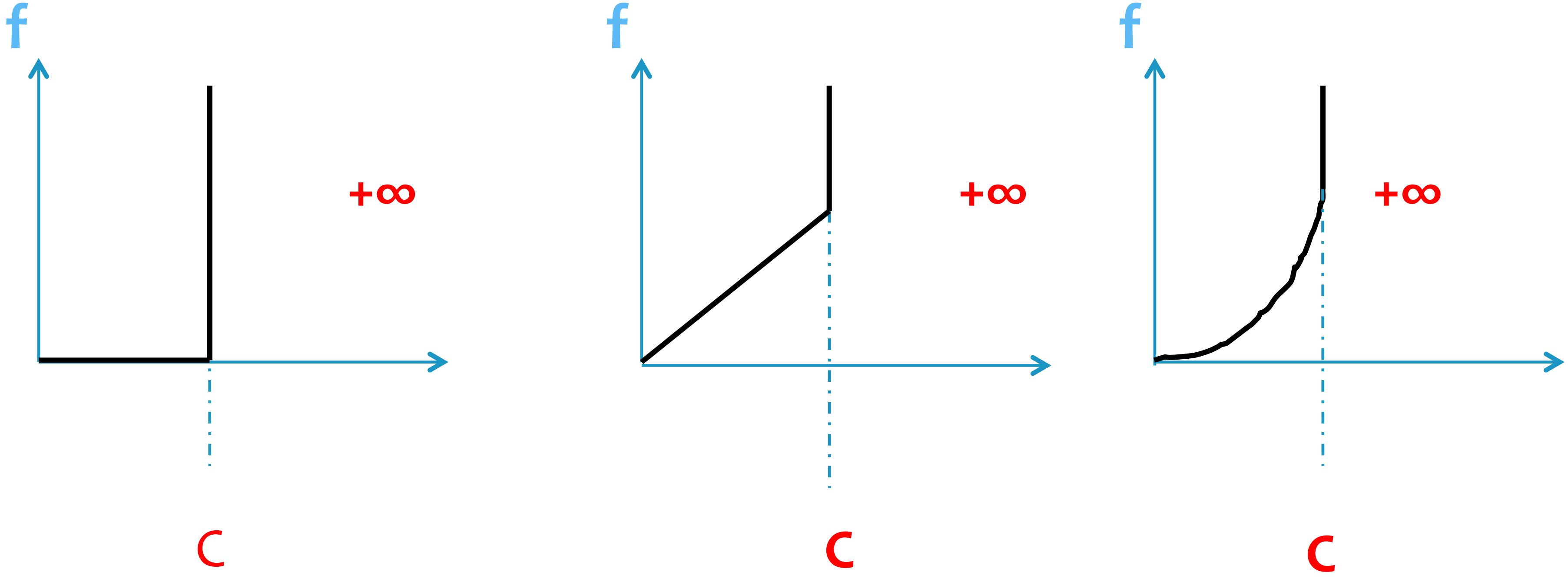
Server cost : f_{rs}

Our Mechanism in the second work

* Convex cost function
* Concave obj function
* Convex optimization

○ Our Mechanism in the second work

* High light: Convex cost function



○ Our Mechanism in the second work

* Online Convex optimization :
  * Primal dual online algorithm

Our Mechanism in the second work

Basic Idea: Regard time as resource

Resource Type: (r,t)

R $\times$ $[t_i^+ - t_i^-]$

$$\text{maximize} \quad \sum_{i\in[I]} \sum_{s\in[S]} b_i x_{is} - \sum_{t\in[T]} \sum_{s\in[S]} \sum_{r\in[R]} f_{rs}(y_{rs}(t)) \qquad (4)$$

subject to:

$$\sum_{s\in[S]} x_{is} \leq 1, \ \ \forall i \in [I] \ (4a)$$

$$\sum_{\substack{i\in[I]:\\ t_i^- \leq t \leq t_i^+}} d_{ir}(t) x_{is} \leq y_{rs}(t), \ \ \forall r \in [R], s \in [S], t \in [T] \ (4b)$$

$$x_{is} \in \{0,1\}, y_{rs}(t) \geq 0, \ \ \forall r \in [R], s \in [S], i \in [I], t \in [T] \ (4c)$$

○ Our Mechanism in the second work

Algorithm:

1. Allocation: maximize user's utility (u= b – p > 0)

2. The more resource allocated, the higher price is.

$P_{rs}(t)=g_{rs}(y_{rs}(t))$ price is a function of allocated amount

$$\text{minimize} \quad \sum_{i\in[I]} u_i + \sum_{t\in[T]} \sum_{r\in[R]} \sum_{s\in[S]} f^*_{rs}(p_{rs}(t)) \qquad (5)$$

subject to:

$$u_i \geq b_i - \sum_{t\in[t_i^-,t_i^+]} \sum_{r\in[R]} d_{ir}(t) p_{rs}(t), \;\; \forall s \in [S], i \in [I] \; (5a)$$

$$p_{rs}(t) \geq 0, \;\; \forall r \in [R], s \in [S], t \in [T] \; (5b)$$

$$u_i \geq 0, \;\; \forall i \in [I] \; (5c)$$

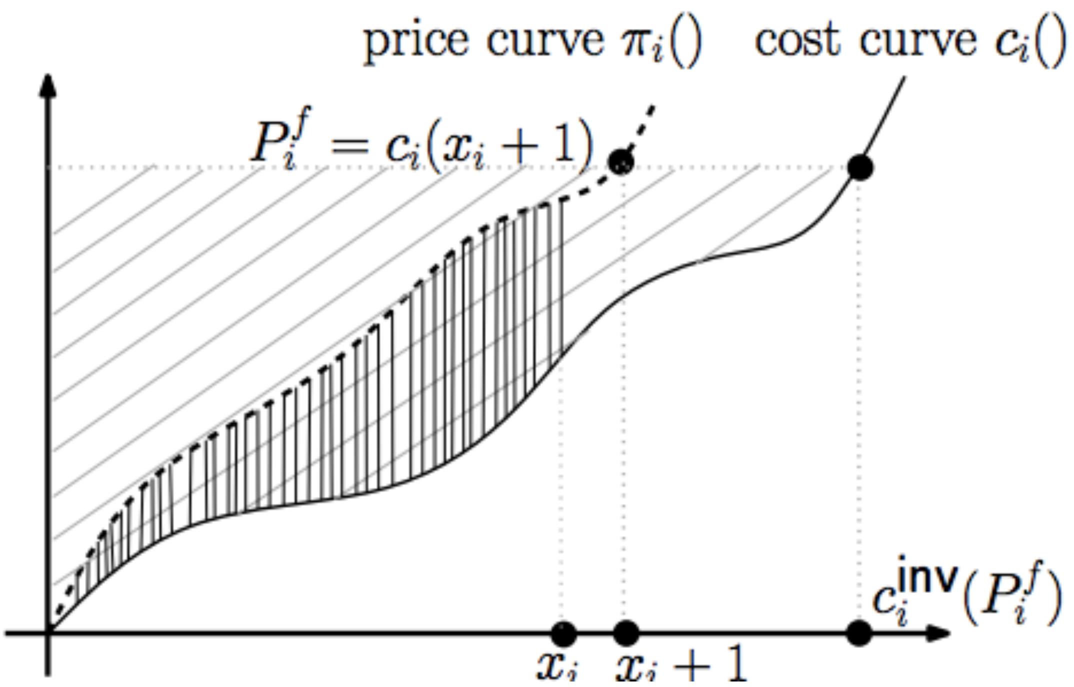
○ Our Mechanism in the second work

Truthfulness: maximize user's utility

$$p_{rs}(y_{rs}(t)) = \begin{cases} f'_{rs}(\delta_{rs} y_{rs}(t)), & y_{rs}(t) \leq \frac{C_{rs}}{\delta_{rs}} \\ f'_{rs}(C_{rs}) e^{\theta_{rs}\left(y_{rs}(t) - \frac{C_{rs}}{\delta_{rs}}\right)}, & y_{rs}(t) > \frac{C_{rs}}{\delta_{rs}} \end{cases}$$

Pricing curve

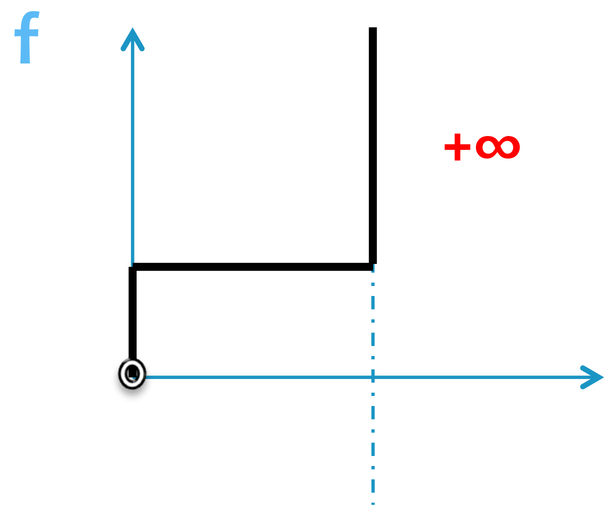
○ Our Mechanism in the second work



Future work

* Topic: Online auction
* Technique: Online primal dual
    * Non-convex optimization, i.e., the following function

Start-up cost



Thank you!

Q&A

Backup Slides

○ Details

* Perturbation-based Randomized Allocation
  * Generate L i.i.d. variables $\Theta_i \sim U(0,1)$
  * Randomly perturb the bidding prices

$$\text{maximize} \sum_{n\in[N]} \sum_{i\in\mathcal{B}_n} b_i x_i$$

$$\bar{b}_i = (1-\epsilon) b_i + \frac{\theta_i \sum_{j=1}^{L} b_j}{L}, \forall i \in [L].$$

$$\vec{\bar{b}} = P\vec{b}.$$

$$P = (1-\epsilon)I + \frac{\vec{\theta}\vec{1}^T}{L}$$

Perturbation
Matrix

Details

* Perturbation-based Randomized Allocation
  - Generate L i.i.d. variables $\Theta_i \sim U(0,1)$
  - Randomly perturb the bidding prices

$$\text{maximize} \sum_{n\in[N]} \sum_{i\in\mathcal{B}_n} b_i x_i \xrightarrow{\vec{\bar{b}} = P\vec{b}.} \bar{b}_i = (1-\epsilon)b_i + \frac{\theta_i \sum_{j=1}^{L} b_j}{L}, \forall i \in [L].$$

  - Exactly solve the perturbed optimization (Alg. 1) which gurantees:

$$\begin{aligned} POPT &= (P\vec{b})^T \vec{x}^p \geq (P\vec{b})^T \vec{x}^* = \vec{b}^T((1-\epsilon)I + \frac{\vec{1}\vec{\theta}^T}{L})\vec{x}^* \\ &\geq (1-\epsilon)\vec{b}^T\vec{x}^* = (1-\epsilon)OPT, \end{aligned}$$

○ Details

- Exactly solve the perturbed optimization

$$\begin{array}{ll} \text{maximize} & \sum_{n\in[N]} \sum_{i\in\mathcal{B}_n} \bar{b}_i x_i \\ \text{subject to:} & \text{constraints (1a)(1b)(1c).} \end{array}$$

Algorithm 1:

- Prove some solutions are bad. They can never be optimal.
- All the other solutions in the feasible region are , Pareto Optimal Solutions, which are not dominated by any other solutions.
- Prove the properties of Pareto Optimal Solutions.
- Dynamic programming based enumeration and pruning to recursively construct Pareto Optimal Solutions.
- Brute search among Prareto Optimal Solutions to obtain the optimal solution of (3), which we call it x^p

○ Details

- Randomly sample out the allocation solution y^{ε} following the distribution:

$$\Omega(\vec{x}^p) = \begin{cases} Pr[\vec{y}^{\epsilon} = \vec{x}^p] = 1 - \epsilon, \\ Pr[\vec{y}^{\epsilon} = \vec{l}_i] = \frac{\sum_{j=1}^{L} \theta_j x_j^p}{L}, \forall i \in \{1, ..., L\}, \\ Pr[\vec{y}^{\epsilon} = \vec{0}] = 1 - Pr[\vec{y}^{\epsilon} = \vec{x}^p] - \sum_{i=1}^{L} Pr[\vec{y}^{\epsilon} = \vec{l}_i]. \end{cases}$$

- y^{ε} is proved to be (1-ε)-approximate solution for allocation

$$E[\vec{y}^{\epsilon}] = (1-\epsilon)\vec{x}^p + (\frac{\sum_{j=1}^{L} \theta_j x_j^p}{L})(\sum_{i=1}^{L} \vec{l}_i) = P^T \vec{x}^p$$

○ Details

* Randomized VCG-like payment

  -- Calculate y^{ϵ}

  -- Payment rule: opportunity cost according to

$$p_i(\vec{y}^{\epsilon}) = \vec{b}_{-i}^{T}\vec{y}_{-i}^{\epsilon} - (\vec{b}^{T}\vec{y}^{\epsilon} - b_i y_i^{\epsilon}), \forall i \in [L].$$

  -- Guarantees truthfulness

○ Details

- y^ε is proved to be (1-ε)-approximate solution for allocation
- The time complexity of Algorithm 1 is

  O(KDL × (#of Pareto Optimal Solutions)^2)
- The expectation of #of Pareto Optimal Solutions is bounded by $1+L^4/\varepsilon$
- Thus the expected time complexity is polynomial

○ Existing Mechanisms

* Wei Wang et al. (*IWQoS'13*)
"Revenue Maximization with Dynamic Auctions in IaaS Cloud Markets,",

* Hong Zhang, Bo Li, Hongbo Jiang, Fangming Liu, A V Vasilakos, and Jiangchuan Liu,
"A Framework for Truthful Online Auctions in Cloud Computing with Heterogeneous User Demands,"
in *Proc. of IEEE INFOCOM*, 2013

* Linquan Zhang, Chuan Wu, and Zongpeng Li,
"Dynamic Resource Provisioning in Cloud Computing: A Randomized Auction Approach,"
in *Proc. Of IEEE INFOCOM, 2014*

* Weijie Shi, Linquan Zhang, Chuan Wu, Zongpeng Li, and Francis C.M. Lau,
"An Online Auction Framework for Dynamic Resource Provisioning in Cloud Computing,"
in *Proc. of ACM SIGMETRICS, 2014*