An Online Auction Framework for Dynamic Resource Provisioning in Cloud Computing

Zongpeng Li

Department of Computer Science, University of Calgary

June 4 2014, at Institute of Network Coding
Our work on Virtual Machine Auctions

- INFOCOM’14, Dynamic Resource Provisioning in Cloud Computing: A Randomized Auction Approach
- IEEE Cloud’14, Core-Selecting Auctions for Dynamically Allocating Heterogeneous VMs in Cloud Computing
- IWQoS’14, RSMOA: A Revenue and Social Welfare Maximizing Online Auction for Dynamic Cloud Resource Provisioning

- Ongoing work: Smoothed polynomial-time auctions and FPTAS auctions for cloud computing, demand response through reverse auctions
Our work on Virtual Machine Auctions

- INFOCOM’14, Dynamic Resource Provisioning in Cloud Computing: A Randomized Auction Approach
- IEEE Cloud’14, Core-Selecting Auctions for Dynamically Allocating Heterogeneous VMs in Cloud Computing
- IWQoS’14, RSMOA: A Revenue and Social Welfare Maximizing Online Auction for Dynamic Cloud Resource Provisioning

- Ongoing work: Smoothed polynomial-time auctions and FPTAS auctions for cloud computing, demand response through reverse auctions
User demands different cloud resources in different geo locations
Current practice in resource provisioning

✦ Amazon EC2

✴ fixed instance types
✴ fixed prices

<table>
<thead>
<tr>
<th>VM Type</th>
<th>CPU*</th>
<th>RAM</th>
<th>Disk</th>
<th>Virginia</th>
<th>Ireland</th>
<th>Tokyo</th>
</tr>
</thead>
<tbody>
<tr>
<td>m1.medium</td>
<td>2</td>
<td>3.75GB</td>
<td>410GB</td>
<td>$0.120</td>
<td>$0.130</td>
<td>$0.175</td>
</tr>
<tr>
<td>m1.large</td>
<td>4</td>
<td>7.5GB</td>
<td>840GB</td>
<td>$0.240</td>
<td>$0.260</td>
<td>$0.350</td>
</tr>
<tr>
<td>m1.xlarge</td>
<td>8</td>
<td>15GB</td>
<td>1.68TB</td>
<td>$0.480</td>
<td>$0.520</td>
<td>$0.700</td>
</tr>
<tr>
<td>c1.medium</td>
<td>5</td>
<td>1.7GB</td>
<td>350GB</td>
<td>$0.145</td>
<td>$0.165</td>
<td>$0.185</td>
</tr>
<tr>
<td>c1.xlarge</td>
<td>20</td>
<td>7GB</td>
<td>1.68TB</td>
<td>$0.580</td>
<td>$0.660</td>
<td>$0.740</td>
</tr>
<tr>
<td>m2.2xlarge</td>
<td>13</td>
<td>34.2GB</td>
<td>850GB</td>
<td>$0.820</td>
<td>$0.920</td>
<td>$1.101</td>
</tr>
</tbody>
</table>

* EC2 compute units
What we want

- Customized VM instances from different datacenters
- A price for the customized VM that caters to the supply-demand relationship at this moment
What we want/what we will do

✧ Customized VM instances from different datacenters

✧ A price for the customized VM that caters to the supply-demand relationship at this moment
What we want/what we will do

✧ Customized VM instances from different datacenters
  ✧ dynamic resource provisioning (i.e., dynamic VM assembly)

✧ A price for the customized VM that caters to the supply-demand relationship at this moment
What we want/what we will do

✦ Customized VM instances from different datacenters
  ✴ dynamic resource provisioning (i.e., dynamic VM assembly)

✦ A price for the customized VM that caters to the supply-demand relationship at this moment
  ✴ a new pricing scheme through an online auction that discovers the “right” price requires no estimation brings more social welfare than fixed pricing
What others have been doing

- Amazon Spot Instances
  - no service guarantees

- “When cloud meets eBay” (Wang et al., INFOCOM 2012)
  - one-round static auction

- COCA (INFOCOM 2013)
  - “A Framework for Truthful Online Auctions in Cloud Computing with Heterogeneous User Demands”, Zhang et al.
  - one type of VMs considered
Our Contribution

- An online auction mechanism for dynamic resource provisioning
  - users’ demands arrive over time; provider responds instantly, without *a priori* information
  - nice properties
    - truthful
    - computationally efficient
  guaranteeing a competitive ratio 3.30 in long-term social welfare in typical scenarios
applied by Zhang calculated, even given unlimited computational resources. A
ness property [18]. The lack of future information brings
tions in a straightforward way usually breaks the truthful-
single round auction using the LP decomposition method.

dettes et al. [27] study the modeling of VM provisioning, but their model
improve the performance of the online auction in practice. Yet
of research areas, such as network bandwidth allocation [26]
taneously target both truthfulness and economic e

economic mechanisms that combine these two approaches and simul-
anisms with maximized social welfare [19]. Auctions are
the incentive compatibility of the allocation algorithm [11]
ed, including from a game theoretical view by analyzing

to custom design a payment rule to work in concert with

times, as part of a hybrid cloud of an enterprise.

There have been some proposals considering bandwidth

do discuss the allocation of

There are

takes both intra- and inter-datacenter tra

discussions on bandwidth auctions as future work.

There have been some proposals considering bandwidth

2. RELATED WORK

Recently, a series of auction mechanisms are designed for resource allocation in computing and communication systems. The cloud auction algorithm, introduced by Bansal et al. [5], is a well-known type of auction for cloud markets, and their model also focuses on dynamic provisioning problem. Wang et al. [24] apply the critical value method, and derive a mechanism that is competitive with the optimal allocation for the future. Critical bids do not include dynamic VM assembling. Shanmuganathan et al. [27] study the modeling of VM provisioning, but their model does not include dynamic VM assembling. Wang [21] introduce the concept of cloud auction algorithm. The bidding language and the user discussions is based on the concept of a packing or covering structure.

Extending single round truthful auctions into online auctions is a classic problem that has been extensively studied, including from a game theoretical view by analyzing the varying amounts of resources may be caused by resource reservation or release of resources for special purposes. The cloud users are bidders in the auctions, each subsuming, a system that provides virtual network abstractions to execute their jobs. The cloud provider acts as the provider. Specifically, let \( n \) be the number of VM instances. Zaman and Grosu [26] discuss the allocation of computing resources bundles, each with a pool of CPU, RAM, and disk storage that can be dynamically assembled into different types to execute their jobs. The cloud provider acts as the system runs in a time-slotted fashion within a span of \( T \) time slot. In each time slot, one round of the auction is carried out, where the cloud provider decides the auction at \( t \), for actually acquiring VMs in its winning bid model by allowing empty bundles in the bids. The terms \( d_{n,k,m,q}^{(t)} \) and \( b_{n,k}^{(t)} \) is the number of type-\( m \) VMs in datacenter \( q \) in her \( k \)-th bundle at \( t \).
Model

user n

bid: $d_{n,k,m,q}^{(t)}$  

$# \text{ of type-} m \text{ VMs in datacenter } q \text{ in her } k\text{-th bundle at } t$

$\text{valuation of user } n \text{ for her } k\text{-th bundle at } t$

Cloud provider

datacenter $q$
The celebrated VCG mechanism [23] is a well known type of mechanism with maximized social welfare [19]. Auctions are used to allocate resources, including from a game theoretical view by analyzing mechanisms that combine these two approaches and simultaneously target both truthfulness and economic efficiency.

Recent studies [20] have proposed and analyzed truthful mechanisms with respect to different types of resources. For example, this can be done by exploiting the concept of economic efficiency and designing a payment rule to work in concert with polynomial-time approximation algorithms.

Wang et al. [27] study the modeling of VM provisioning, but their model only focuses on one type of VMs. Other works, such as Bansal et al. [28] and Meng et al. [24], apply different decomposing methods to address this issue. However, our work is more advanced than theirs in two aspects: (1) We consider the problem over a period of time, to study dynamic VM provisioning, and designs a truthful pricing rule. However, when the underlying allocation problem is time-varying, VCG becomes computationally infeasible. When critical bids are given, as done in [13], another alternative is to apply the LP decomposition technique.

Classic applications of auctions are found in a wide range of research areas, such as network bandwidth allocation [26]. The varying amounts of resources may be caused by reusing existing resources, so a mechanism that respects this constraint is necessary. Specifically, let $n,k,m,q$ denote the integer $n$-tuple of elements. The terms $n,k,m,q$ are used interchangeably.

The system runs in a time-slotted fashion within a span of $T$ time slots. In each time slot, one round of the auction results, where the cloud provider decides the allocation of VMs while considering network congestion.

The terms $O(k)$ and $Q(n,k)$ are used interchangeably.

The varying amounts of resources may be caused by reusing existing resources, so a mechanism that respects this constraint is necessary. Specifically, let $n,k,m,q$ denote the integer $n$-tuple of elements. The terms $n,k,m,q$ are used interchangeably.
the heterogeneous demands in cloud market. However, they characteristic proposed in their work are novel, and capture cloud auction algorithm. The bidding language and the user calculated, even given unlimited computational resources. A since the optimal allocation for the future cannot be calculated, VCG auction does not directly work in the online setting, despite the critical value method, and derive a mechanism that is collusion-resistant, an important property in practice. Yet the approximation algorithm at hand, to achieve truthfulness poses improvements to the decomposing method, which can instead of just one round, and serve cloud users in an online manner. The lack of future information brings the incentive compatibility of the allocation algorithm [11] into play, including from a game theoretical view by analyzing the critical bids [20], VCG becomes computationally infeasible. When the underlying allocation problem is NP-hard, which is common for combinatorial auctions, it is essentially the only type of auction that simultaneously guarantees both truthfulness and absolute economic efficiency. It is the number of VM instances. Zaman and Grosu [27] study the modeling of VM provisioning, but their model does not include dynamic VM assembling. Shanmuganathan [28] et al. discuss the allocation of VMs in datacenter. Ballani [5] propose Okto-resource, a bundle consists of a list of desired quantities of VMs of different types, as well as part of a hybrid cloud of an enterprise. The varying amounts of resources may be caused by resource, for all different types, as well as the bidder's valuation for the bundle...

3. RELATED WORK

Classic applications of auctions are found in a wide range of systems is a classic problem that has been extensively studied. Allocating resources to custom design a payment rule to work in concert with the underlying social welfare maximization. In absence of economic e.

3.1 The Cloud System

Resource allocation in computing and communication systems is a classic problem that has been extensively studied. Allocating resources to custom design a payment rule to work in concert with the underlying social welfare maximization. In absence of economic e.

Allocation in datacenters. Wang [6] discuss the allocation of datacenter. Bansal [4] et al. propose Okto-resource, a bundle consists of a list of desired quantities of VMs of different types, as well as part of a hybrid cloud of an enterprise. The varying amounts of resources may be caused by resource, for all different types, as well as the bidder's valuation for the bundle...

The varying amounts of resources may be caused by resource, for all different types, as well as the bidder's valuation for the bundle...

Allocation in datacenters. Wang [6] discuss the allocation of datacenter. Bansal [4] et al. propose Okto-resource, a bundle consists of a list of desired quantities of VMs of different types, as well as part of a hybrid cloud of an enterprise. The varying amounts of resources may be caused by resource, for all different types, as well as the bidder's valuation for the bundle...

Allocation in datacenters. Wang [6] discuss the allocation of datacenter. Bansal [4] et al. propose Okto-resource, a bundle consists of a list of desired quantities of VMs of different types, as well as part of a hybrid cloud of an enterprise. The varying amounts of resources may be caused by resource, for all different types, as well as the bidder's valuation for the bundle...

Allocation in datacenters. Wang [6] discuss the allocation of datacenter. Bansal [4] et al. propose Okto-resource, a bundle consists of a list of desired quantities of VMs of different types, as well as part of a hybrid cloud of an enterprise. The varying amounts of resources may be caused by resource, for all different types, as well as the bidder's valuation for the bundle...

Allocation in datacenters. Wang [6] discuss the allocation of datacenter. Bansal [4] et al. propose Okto-resource, a bundle consists of a list of desired quantities of VMs of different types, as well as part of a hybrid cloud of an enterprise. The varying amounts of resources may be caused by resource, for all different types, as well as the bidder's valuation for the bundle...
Model

user n

allocation decision: \( y_{n,k}^{(t)} = 1 \) user n gets her k-th bid bundle;

Cloud provider

datacenter q

\[ y_{n,k}^{(t)} \] denotes the number of type-\( k \) resource, for all \( t \).

The varying amounts of resources may be caused by reserved or released resources for special purposes.

Each VM of type \( M \) is dynamically assembled into a packing or covering structure.

Auctions are used to allocate the available amounts of resources at each datacenter, each with a pool of type-\( M \) virtual machines, for all possible VM types to execute their jobs. The cloud provider acts as the auctioneer and makes the allocation decision:

\[ y_{n,k}^{(t)} = 1 \] user n gets her k-th bid bundle;

The terms \( y_{n,k}^{(t)} \) specify in its bid bundle; \( y_{n,k}^{(t)} \) is the number of VM instances. Zaman and Grosu et al. introduce the concept of VM allocation for cloud markets.

Extending single round truthful auctions into online auctions is carried out, where the cloud provider decides the allocation decision:

\[ y_{n,k}^{(t)} = 1 \] user n gets her k-th bid bundle;

The critical value method, and derive a mechanism that is collusion-resistant, an important property in practice. Yet the approximation algorithm at hand, to achieve truthfulness; for example, this can be done by exploiting the concept of economic e.

For large numbers of bundles, it is computationally infeasible. When the underlying allocation problem is NP-hard, which is common for combinatorial auctions.

It is essentially the only type of auction that simultaneously guarantees both truthfulness and absolute efficiency (social welfare maximization), through the underlying social welfare maximization. In absence of small number of bundles than maximum bidding power.

An alternative approach is designing pricing mechanisms with maximized social welfare. Auctions are usually truthful in practice. (2) We not only apply but also propose an auction algorithm. The bidding language and the user applied by Zhang known technique for achieving truthfulness in online auctions. It is essentially the only type of auction that simultaneously guarantees both truthfulness and absolute efficiency (social welfare maximization), through the approximation algorithm at hand, to achieve truthfulness; for example, this can be done by exploiting the concept of economic e.
The bidding language and the user interactions is based on the concept of a known technique for achieving truthfulness in online auctions. Since the optimal allocation for the future cannot be calculated in advance, we decompose the problem into smaller sub-problems, which can be solved more easily. This approach poses improvements to the decomposing method, which can be used to improve the performance of the online auction in practice.

However, our work is more advanced than theirs in two aspects. First, we consider a cloud spanning multiple datacenters, whereas their work only considers a single datacenter. Second, we not only apply but also propose a mechanism more closely resembling a real-world manner. Our mechanism more closely resembles a real-world auction where the cloud provider decides the supply curve, and users bid on specific resources.

In each auction, upon receiving users' bids, the cloud auctioneer computes its resource allocation and produces the auction results. The system runs in a time-slotted fashion within a span of time. In each time slot, the cloud auctioneer receives bids from users, and allocates resources to the winners. The allocation decision is made based on the bids received in that time slot. The allocation decision is binary: user n gets her k-th bid bundle if \( y_{n,k}^{(t)} = 1 \), and user n does not get her k-th bid bundle if \( y_{n,k}^{(t)} = 0 \).
the heterogeneous demands in cloud market. However, they characteristic proposed in their work are novel, and capture cloud auction algorithm. The bidding language and the user applications is based on the concept of of a known technique for achieving truthfulness in online auc-

improve the performance of the online auction in practice. Pose improvements to the decomposing method, which can instead of just one round, and serve cloud users in an online.

However, our work is more advanced than theirs in two as-

et al. [27] study the modeling of VM provisioning, but their model does not include dynamic VM assembling. Shanmuganathan et al. [12], as

Classic applications of auctions are found in a wide range

taneously target both truthfulness and economic e

Extending single round truthful auctions into online auc-

The celebrated VCG mechanism [23] is a well known type of

solving the underlying social welfare maximization problem,

polynomial-time approximation algorithms are applied to

T. The system runs in a time-slotted fashion within a span of

T. Each VM of type M is constituted of a list of desired quantities of VMs of different types to execute their jobs. The cloud provider acts as the cloud auctions is carried out, where the cloud provider decides the allocation decision:

allocation decision: \( y_{n,k}^{(t)} \)

user n wins at most one bundle in each round

user n gets her k-th bid bundle;

user n does not get her k-th bid bundle.
user n has an overall budget $B_n$

cloud provider maximizes social welfare ($= \text{total valuation}$)
user \( n \) has an overall budget \( B_n \)

cloud provider maximizes social welfare (= total valuation)

maximize
\[
\sum_{t \in [T]} \sum_{n \in [N]} \sum_{k \in [K]} b_{n,k}^{(t)} y_{n,k}^{(t)}
\]

subject to
\[
\sum_{k \in [K]} y_{n,k}^{(t)} \leq 1, \quad \forall n \in [N], t \in [T],
\]
\[
\sum_{k \in [K]} \sum_{t \in [T]} b_{n,k}^{(t)} y_{n,k}^{(t)} \leq B_n, \quad \forall n \in [N],
\]
\[
\sum_{n \in [N]} \sum_{k \in [K]} c_{n,k,r}^{(t)} y_{n,k}^{(t)} \leq A_{q,r}^{(t)}, \quad \forall q \in [Q], r \in [R], t \in [T],
\]
\[
y_{n,k}^{(t)} \in \{0, 1\}, \quad \forall n \in [N], k \in [K], t \in [T]
\]
user n has an overall budget $B_n$

cloud provider maximizes social welfare (= total valuation)

\[
\text{maximize } \sum_{t \in [T]} \sum_{n \in [N]} \sum_{k \in [K]} b_{n,k}^{(t)} y_{n,k}^{(t)} \\
\text{subject to } \sum_{k \in [K]} y_{n,k}^{(t)} \leq 1, \quad \forall n \in [N], t \in [T], \\
\sum_{k \in [K]} \sum_{t \in [T]} b_{n,k}^{(t)} y_{n,k}^{(t)} \leq B_n, \quad \forall n \in [N], \\
\sum_{n \in [N]} \sum_{k \in [K]} c_{n,k,r}^{(t)} y_{n,k}^{(t)} \leq A_{q,r}^{(t)}, \quad \forall q \in [Q], r \in [R], t \in [T], \\
y_{n,k}^{(t)} \in \{0, 1\}, \quad \forall n \in [N], k \in [K], t \in [T]
\]
Online Problem

- The budget couples decisions in different rounds of the auction

Example: greedy vs. optimal allocation strategy

<table>
<thead>
<tr>
<th>User A</th>
<th>B_n = $20</th>
</tr>
</thead>
<tbody>
<tr>
<td>Round 1</td>
<td>$6</td>
</tr>
<tr>
<td>Round 2</td>
<td>$7</td>
</tr>
<tr>
<td>Round 3</td>
<td>$10</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>User B</th>
<th>B_n = $20</th>
</tr>
</thead>
<tbody>
<tr>
<td>Round 1</td>
<td>$3</td>
</tr>
<tr>
<td>Round 2</td>
<td>$6</td>
</tr>
<tr>
<td>Round 3</td>
<td>$2</td>
</tr>
</tbody>
</table>
Online Problem

- The budget couples decisions in different rounds of the auction

Example: greedy vs. optimal allocation strategy

User A
Round 1: $6
Remaining Budget: $14

User B
Round 1: $3
Remaining Budget: $20
Online Problem

- The budget couples decisions in different rounds of the auction

Example: greedy vs. optimal allocation strategy

**User A**
- Round 1: $6
- Round 2: $7
- Remaining Budget: $7

**User B**
- Round 1: $3
- Round 2: $6
- Remaining Budget: $20
Online Problem

- The budget couples decisions in different rounds of the auction

Example: greedy vs. optimal allocation strategy

User A:
- Round 1: $6
- Round 2: $7
- Round 3: $10
- Remaining Budget: $7

User B:
- Round 1: $3
- Round 2: $6
- Round 3: $2
- Remaining Budget: $18
Online Problem

The budget couples decisions in different rounds of the auction

Example: greedy vs. optimal allocation strategy

Greedy algorithm: social welfare $15
Online Problem

- The budget couples decisions in different rounds of the auction

Example: greedy vs. optimal allocation strategy

<table>
<thead>
<tr>
<th>User</th>
<th>Budget $B_n = $20</th>
<th>Round 1</th>
<th>Round 2</th>
<th>Round 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td>$6</td>
<td>$7</td>
<td>$10</td>
</tr>
<tr>
<td>B</td>
<td></td>
<td>$3</td>
<td>$6</td>
<td>$2</td>
</tr>
</tbody>
</table>

Greedy algorithm: social welfare $15
Optimal solution: social welfare $22
Online Problem

Lessons learned: do NOT exhaust a user’ budget early

- may lose all the opportunities later on the user
- but, how to seize the best opportunities to maximize social welfare over long term: classical online optimization dilemma

User A \( B_n = $20 \)
- Round 1 \$6
- Round 2 \$7
- Round 3 \$10

User B \( B_n = $20 \)
- Round 1 \$3
- Round 2 \$6
- Round 3 \$2

Greedy algorithm: social welfare $15
Optimal solution: social welfare $22
**Budget Coefficient**

- Higher priority for allocating resource to user with higher remaining budget in each round:

  \[
  \text{Original valuation} \times \text{Budget coefficient } (1 - x_n^{(t)})
  \]
The Online Algorithm Framework

1: $x_n^{(0)} \leftarrow 0, \forall n \in [N]$
2: // Loop for each time slot
3: for all $1 \leq t \leq T$ do
4:
$$w_{n,k}^{(t)} = \begin{cases} 0 & \text{if } x_n^{(t-1)} \geq 1, \forall n \in [N], k \in [K]. \\
 b_{n,k}^{(t)}(1 - x_n^{(t-1)}) & \text{otherwise} \end{cases}$$
5: Run $A_{round}$. Let $\mathcal{N}$ be the set of winning users, and $k_n$ be the index of their corresponding winning bundle, for each winning user $n \in \mathcal{N}$.
6: for all $n \in \mathcal{N}$ do
7:
$$x_n^{(t)} \leftarrow x_n^{(t-1)} \left(1 + \frac{b_{n,k_n}^{(t)}}{B_n}\right) + \frac{b_{n,k_n}^{(t)}}{B_n(\gamma - 1)}$$
8: end for
9: for all $n \notin \mathcal{N}$ do
10: $x_n^{(t)} \leftarrow x_n^{(t-1)}$
11: end for
12: end for
13: $x_n \leftarrow x_n^{(T)}, \forall n \in [N]$
The Online Algorithm Framework \textbf{A}_{\text{online}}

1: $x_n^{(0)} \leftarrow 0, \forall n \in [N]$
2: \hspace{1em} // Loop for each time slot
3: \hspace{2em} for all $1 \leq t \leq T$ do
4: \hspace{3em} $w_{n,k}^{(t)} = \begin{cases} 
0 & \text{if } x_n^{(t-1)} \geq 1, \forall n \in [N], k \in [K]. \\
 b_{n,k}^{(t)}(1 - x_n^{(t-1)}) & \text{otherwise}
\end{cases}$
5: \hspace{2em} Run $A_{\text{round}}$. Let $\mathcal{N}$ be the set of winning users, and $k_n$ be the index of their corresponding winning bundle, for each winning user $n \in \mathcal{N}$.
6: \hspace{3em} for all $n \in \mathcal{N}$ do
7: \hspace{4em} $x_n^{(t)} \leftarrow x_n^{(t-1)} \left( 1 + \frac{b_{n,k_n}^{(t)}}{B_n} \right) + \frac{b_{n,k_n}^{(t)}}{B_n(\gamma - 1)}$
8: \hspace{3em} end for
9: \hspace{2em} for all $n \notin \mathcal{N}$ do
10: \hspace{3em} $x_n^{(t)} \leftarrow x_n^{(t-1)}$
11: \hspace{2em} end for
12: end for
13: $x_n \leftarrow x_n^{(T)}, \forall n \in [N]$
The Online Algorithm Framework $\mathcal{A}_{\text{online}}$

1: $x_n^{(0)} \leftarrow 0, \forall n \in [N]$
2: // Loop for each time slot
3: for all $1 \leq t \leq T$ do
4: 
   \[
   w_{n,k}^{(t)} = \begin{cases} 
   0 & \text{if } x_n^{(t-1)} \geq 1 \\
   b_{n,k}^{(t)} (1 - x_n^{(t-1)}) & \text{otherwise}
   \end{cases}, \forall n \in [N], k \in [K].
   \]
5: Run $\mathcal{A}_{\text{round}}$. Let $\mathcal{N}$ be the set of winning users, and $k_n$ be the index of their corresponding winning bundle, for each winning user $n \in \mathcal{N}$.
6: for all $n \in \mathcal{N}$ do
7: 
   \[
   x_n^{(t)} \leftarrow x_n^{(t-1)} \left(1 + \frac{b_{n,k_n}^{(t)}}{B_n} \right) + \frac{b_{n,k_n}^{(t)}}{B_n(\gamma - 1)}
   \]
8: end for
9: for all $n \notin \mathcal{N}$ do
10: 
   \[
   x_n^{(t)} \leftarrow x_n^{(t-1)}
   \]
11: end for
12: end for
13: $x_n \leftarrow x_n^{(T)}, \forall n \in [N]$
An example run:

only one item; Around simply chooses the user with the larger adjusted valuation as the winner

User A $B_n=$20
Round 1 $6$
Round 2 $7$
Round 3 $10$

User B $B_n=$20
Round 1 $3$
Round 2 $6$
Round 3 $2$
An example run:

only one item; **Around** simply chooses the user with the larger adjusted valuation as the winner.

User A  $B_n = 20$  $x_n = 0$

Round 1  $\$6$

Adjusted:  $6 \times (1 - 0) = \$6$

Update:  $x_n = 0.24$

User B  $B_n = 20$  $x_n = 0$

Round 1  $\$3$

Adjusted:  $3 \times (1 - 0) = \$3$
An example run:

only one item; Around simply chooses the user with the larger adjusted valuation as the winner

User A  \( B_n =$20  \( x_n = 0.24 \)

Round 1 \$6
Round 2 \$7
Adjusted: \$7\times(1-0.24)=\$5.32

User B  \( B_n =$20 \( x_n = 0 \)

Round 1 \$3
Round 2 \$6
Adjusted: \$6\times(1-0)=\$6

Update: \( x_n = 0.24 \)
An example run:

only one item; Around simply chooses the user with the larger adjusted valuation as the winner

User A  \( B_n=20 \)  \( x_n=0.24 \)
Round 1        $6
Round 2        $7
Round 3        $10
Adjusted:      $10(1-0.24)=7.6
Update:        \( x_n=0.76 \)

User B  \( B_n=20 \)  \( x_n=0.24 \)
Round 1        $3
Round 2        $6
Round 3        $2
Adjusted:      $2(1-0.24)=1.52

Greedy algorithm: social welfare $15
Optimal solution: social welfare $22
Online algorithm: social welfare $22
One-Round Auction Around

- To decide the winners and winning bundles (resource allocation decisions), and the price to charge for each bundle (payment mechanism)
One-Round Auction Around

✧ To decide the winners and winning bundles (resource allocation decisions), and the price to charge for each bundle (payment mechanism)

✧ Main design objective: truthfulness
One-Round Auction Around

- To decide the winners and winning bundles (resource allocation decisions), and the price to charge for each bundle (payment mechanism)

- Main design objective: truthfulness

- Payment mechanism is the key to guarantee truthfulness
  * can be very difficult to design
One-Round Auction Around

✧ To decide the winners and winning bundles (resource allocation decisions), and the price to charge for each bundle (payment mechanism)

✧ Main design objective: truthfulness

✧ Payment mechanism is the key to guarantee truthfulness
  ✴ can be very difficult to design

✴ VCG auction is a truthful mechanism
  charges bidder the opportunity cost
  needs to compute the exact optimal allocation (cannot be approximate solution)
One-Round Resource Allocation Problem

maximize \( \sum_{n \in [N]} \sum_{k \in [K]} w_{n,k}^{(t)} y_{n,k}^{(t)} \)

subject to

\( \sum_{k \in [K]} y_{n,k}^{(t)} \leq 1 \quad \forall n \in [N] \)

\( \sum_{n \in [N]} \sum_{k \in [K]} c_{n,k,r,q}^{(t)} y_{n,k}^{(t)} \leq A_{q,r}^{(t)} \quad \forall q \in [Q], r \in [R] \)

\( y_{n,k}^{(t)} \in \{0, 1\} \quad \forall n \in [N], k \in [K] \)

\( w_{n,k}^{(t)} \) : adjusted user n’s valuation for k-th bundle
\( c_{n,k,r,q}^{(t)} \) : amount of resource r at dc q in n’s k-th bundle
\( A_{q,r}^{(t)} \) : total resource r at dc q at t
\( y_{n,k}^{(t)} \) : decision variable, bundle allocated or not
Fractional VCG

- Relax $y_{n,k}^{(t)}$
- Compute optimal fractional allocation: an LP
- Use the same VCG payment mechanism
Fractional VCG

- Relax $y_{n,k}^{(t)}$
- Compute optimal fractional allocation: an LP
- Use the same VCG payment mechanism
- But, fractional bundle allocation is not feasible in practice. We cannot provide 0.3 of a VM instance to a user.
**Fractional VCG**

- Relax $y_{n,k}^{(t)}$
- Compute optimal fractional allocation: an LP
- Use the same VCG payment mechanism
- But, fractional bundle allocation is not feasible in practice
  - we cannot provide 0.3 of a VM instance to a user
- **Solution**: decompose the fractional solution to a combination of integer solutions, such that the allocation in expectation remains the same
Fractional VCG

- Relax $y_{n,k}^{(t)}$
- Compute optimal fractional allocation: an LP
- Use the same VCG payment mechanism
- But, fractional bundle allocation is not feasible in practice
  we cannot provide 0.3 of a VM instance to a user
- Solution: decompose the fractional solution to a combination of integer solutions, such that the allocation in expectation remains the same

\[
\begin{align*}
\text{minimize} \quad & \sum_{l \in L} \beta_l \\
\text{s.t.} \quad & \sum_{l \in L} \beta_l y_{n,k}^{(t)} = y_{n,k}^{(t)F} / \lambda, \quad \forall n \in [N], k \in [K], \\
& \sum_{l \in L} \beta_l \geq 1, \\
& \beta_l \geq 0, \quad \forall l \in L.
\end{align*}
\]
**Fractional VCG**

- Relax $y^{(t)}_{n,k}$
- Compute optimal fractional allocation: an LP
- Use the same VCG payment mechanism
- But, fractional bundle allocation is not feasible in practice

we cannot provide 0.3 of a VM instance to a user

**Solution**: decompose the fractional solution to a combination of integer solutions, such that the allocation in expectation remains the same

\[
\text{minimize} \quad \sum_{l \in L} \beta_l \\
\text{s.t.} \quad \sum_{l \in L} \beta_l y^{(t)}_{n,k} = y^{(t)}_{n,k} / \lambda, \quad \forall n \in [N], k \in [K], \\
\sum_{l \in L} \beta_l \geq 1, \\
\beta_l \geq 0, \quad \forall l \in L.
\]
**Fractional VCG**

- Relax $y_{n,k}^{(t)}$
- Compute optimal fractional allocation: an LP
- Use the same VCG payment mechanism
- But, fractional bundle allocation is not feasible in practice, we cannot provide 0.3 of a VM instance to a user
- Solution: decompose the fractional solution to a combination of integer solutions, such that the allocation in expectation remains the same

\[
\begin{align*}
\text{minimize} & \quad \sum_{l \in L} \beta_l \\
\text{subject to} & \quad \sum_{l \in L} \beta_l y_{n,k}^{(t)l} = y_{n,k}^{(t)F} / \lambda, \quad \forall n \in [N], k \in [K], \\
& \quad \sum_{l \in L} \beta_l \geq 1, \\
& \quad \beta_l \geq 0, \quad \forall l \in L.
\end{align*}
\]
**Fractional VCG**

- **Relax** $y_{n,k}^{(t)}$
- **Compute optimal fractional allocation**: an LP
- **Use the same VCG payment mechanism**
- **But, fractional bundle allocation is not feasible in practice**
  we cannot provide 0.3 of a VM instance to a user
- **Solution**: decompose the fractional solution to a combination of integer solutions, such that the allocation in expectation remains the same

\[
\text{minimize } \sum_{l \in L} \beta_l \\
\text{s.t. } \sum_{l \in L} \beta_l y_{n,k}^{(t)} = y_{n,k}^{(t)F} / \lambda, \quad \forall n \in [N], k \in [K], \]
\[
\sum_{l \in L} \beta_l \geq 1, \\
\beta_l \geq 0, \quad \forall l \in L.
\]
Fractional VCG

- Relax $y^{(t)}_{n,k}$
- Compute optimal fractional allocation: an LP
- Use the same VCG payment mechanism
- But, fractional bundle allocation is not feasible in practice, we cannot provide $0.3$ of a VM instance to a user
- Solution: decompose the fractional solution to a combination of integer solutions, such that the allocation in expectation remains the same

\[
\begin{align*}
\text{minimize} & \quad \sum_{l \in L} \beta_l \\
\text{s.t.} & \quad \sum_{l \in L} \beta_l y^{(t)}_{n,k} = y^{(t)}_{n,k} / \lambda, \quad \forall n \in [N], k \in [K], \\
& \quad \sum_{l \in L} \beta_l \geq 1, \\
& \quad \beta_l \geq 0, \quad \forall l \in L.
\end{align*}
\]
Randomized Decomposition

<table>
<thead>
<tr>
<th>User A</th>
<th>User B</th>
<th>User C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fractional solution:</td>
<td>0.3</td>
<td>0.8</td>
</tr>
</tbody>
</table>

- Decomposed: 1 1 0 Pr = 0.3
- Integer solution: 0 1 1 Pr = 0.5
- 0 0 0 Pr = 0.2

minimize $\sum_{l \in L} \beta_l$

s.t.

$\sum_{l \in L} \beta_l y_{n,k}^{(t)} = y_{n,k}^{(t)F} / \lambda, \quad \forall n \in [N], k \in [K]$,

$\sum_{l \in L} \beta_l \geq 1,$

$\sum_{l \in L} \beta_l \geq 1,$

$\beta_l \geq 0, \quad \forall l \in L.$
Randomized Decomposition

- How to decide the scale-down factor $\lambda_l$
- How to find a set of integer solution $y_{n,k}^{(t)}$ and the corresponding probabilities $\beta_l$

minimize $\sum_{l \in L} \beta_l$

s.t.

$$\sum_{l \in L} \beta_l y_{n,k}^{(t)} = y_{n,k}^{(t)F} / \lambda, \quad \forall n \in [N], k \in [K],$$

$$\sum_{l \in L} \beta_l \geq 1,$$

$$\beta_i \geq 0, \quad \forall l \in L.$$
Finding $\lambda$

- Use the approximation ratio of an algorithm to solve the one-round allocation problem as the scaling-down factor
  guaranteed to find a feasible solution of the decomposition problem
Finding $\lambda$

A primal-dual approximation algorithm to solve the one-round allocation problem

1: $\mathcal{N} \leftarrow \emptyset$, $z_{\text{base}} \leftarrow QR \cdot \exp((C_{\text{min}}^{(t)} - 1))$
2: $y_{n,k}^{(t)} \leftarrow 0$, $s_n^{(t)} \leftarrow 0$, $z_{q,r}^{(t)} \leftarrow 1/A_{q,r}^{(t)}$, $\forall n \in [N], k \in [K], r \in [R], q \in [Q]$
3: while $\sum_{r \in [R]} \sum_{q \in [Q]} A_{q,r}^{(t)} z_{q,r}^{(t)} < z_{\text{base}}$ AND $|\mathcal{N}| \neq N$ do
4:   for all $n \notin \mathcal{N}$ do
5:     $k(n) = \arg\max_{k \in [K]} \{w_{n,k}^{(t)}\}$
6:   end for
7:   $n^* = \arg\max_{n \in [N]} \left\{ \frac{w_{n,k(n)}^{(t)}}{\sum_{r \in [R]} \sum_{q \in [Q]} c_{n,k(n),r,q}^{(t)} z_{q,r}^{(t)}} \right\}$
8:   $y_{n^*,k(n^*)}^{(t)} \leftarrow 1$, $s_{n^*}^{(t)} \leftarrow w_{n^*,k(n^*)}^{(t)}$, $\mathcal{N} \leftarrow \mathcal{N} \cup \{n^*\}$
9:   for all $r \in [R], q \in [Q]$ do
10:      $z_{q,r}^{(t)} \leftarrow z_{q,r}^{(t)} \cdot z_{\text{base}} c_{n^*,k(n^*)}^{(t),q,r}/(A_{q,r}^{(t)} - C_{q,r}^{(t)})$
11: end for
12: end while

Dual variable of the resource constraint: the unit price of each type of resources

Evaluate a bundle according to unit prices and required resources; choose users with a higher bid on a lower-valued bundle as the winner

Update the unit price of recourses: higher price if larger amount of the resource consumed
Finding integer solutions and probabilities

- Difficult to solve directly since we need to find the exponentially many feasible integer solutions first

\[
\begin{align*}
\text{minimize} & \quad \sum_{l \in L} \beta_l \\
\text{s.t.} & \quad \sum_{l \in L} \beta_l y_{n,k}^{(t)} = y_{n,k}^{(t)F} / \lambda, \quad \forall n \in [N], k \in [K], \\
& \quad \sum_{l \in L} \beta_l \geq 1, \\
& \quad \beta_l \geq 0, \quad \forall l \in L.
\end{align*}
\]
Finding integer solutions and probabilities

Difficult to solve directly since we need to find the exponentially many feasible integer solutions first

Solution: resort to the dual

solve the dual using Ellipsoid method in polynomial time, using the primal-dual approximation algorithm as the separation oracle

\[
\text{minimize } \sum_{l \in L} \beta_l \\
\text{s.t. } \sum_{l \in L} \beta_l y^{(t)l}_{n,k} = y^{(t)F}_{n,k} / \lambda, \quad \forall n \in [N], k \in [K], \\
\sum_{l \in L} \beta_l \geq 1, \\
\beta_l \geq 0, \quad \forall l \in L.
\]
One-Round Random Auction

1: Solve LP relaxation of (3), with \( w_{n,k}^{(t)} = \max\{0, (1 - x_{n}^{(t-1)}b_{n,k}^{(t)}) \} \). Denote the fractional solution by \( y_{n,k}^{(t)F} \), \( \forall n \in [N], k \in [K] \).
2: for all \( n \in [N] \) do
3: \( \forall n' \in [N], k \in [K], w_{n',k}^{(t)} = \max\{0, (1 - x_{n}^{(t-1)}b_{n,k}^{(t)}) \} \), if \( n' \neq n \). Otherwise \( w_{n',k}^{(t)} = 0 \).
4: Solve LP relaxation of (3), with \( w_{n',k}^{(t)} \)'s. Denote the optimal objective function value by \( \tilde{V}_{-n}^{(t)} \).
5: \( \Pi_{n}^{(t)F} = \tilde{V}_{-n}^{(t)} - \sum_{n' \neq n} \sum_{k} y_{n',k}^{(t)F} w_{n',k}^{(t)} \)
6: end for
7: Solve the pair of primal-dual decomposition LPs in (6) and (7) using the ellipsoid method, using Alg. 2 as a separation oracle, and derive a polynomial number of integer solutions to (3), \( y_{n}^{(t)l} \), \( \forall l \in L \), and the corresponding decomposition coefficients, \( \beta_{l} \), \( \forall l \in L \).
8: Choose \( y_{n}^{(t)l} \) with probability \( \beta_{l} \), \( \forall l \in L \)
9: \( \forall n \in [N], \Pi_{n}^{(t)l} = \Pi_{n}^{(t)F} \cdot \frac{\sum_{k \in [K]} w_{n,k}^{(t)} y_{n,k}^{(t)l}}{\sum_{k \in [K]} w_{n,k}^{(t)} y_{n,k}^{(t)F}} \)
One-Round Random Auction Around

1: Solve LP relaxation of (3), with \( w_{n,k}^{(t)} = \max\{0, (1 - x_n^{(t-1)})b_{n,k}^{(t)}\} \). Denote the fractional solution by \( y_{n,k}^{(t)F} \), \( \forall n \in [N], k \in [K] \).
2: for all \( n \in [N] \) do
3: \( \forall n' \in [N], k \in [K], w_{n',k}^{(t)} = \max\{0, (1 - x_n^{(t-1)})b_{n,k}^{(t)}\} \), if \( n' \neq n \). Otherwise \( w_{n',k}^{(t)} = 0 \).
4: Solve LP relaxation of (3), with \( w_{n',k}^{(t)} \)'s. Denote the optimal objective function value by \( \widetilde{V}_{-n}^{(t)} \).
5: \( \Pi_{n}^{(t)F} = \widetilde{V}_{-n}^{(t)} - \sum_{n' \neq n} \sum_{k} y_{n',k}^{(t)F} w_{n',k}^{(t)} \).
6: end for
7: Solve the pair of primal-dual decomposition LPs in (6) and (7) using the ellipsoid method, using Alg. 2 as a separation oracle, and derive a polynomial number of integer solutions to (3), \( y^{(t)l} \), \( \forall l \in L \), and the corresponding decomposition coefficients, \( \beta_l \), \( \forall l \in L \).
8: Choose \( y^{(t)l} \) with probability \( \beta_l \), \( \forall l \in L \)
9: \( \forall n \in [N], \Pi_{n}^{(t)l} = \Pi_{n}^{(t)F} \cdot \frac{\sum_{k \in [K]} w_{n,k}^{(t)} y_{n,k}^{(t)l}}{\sum_{k \in [K]} w_{n,k}^{(t)} y_{n,k}^{(t)F}} \).
One-Round Random Auction

1: Solve LP relaxation of (3), with \( w_{n,k}^{(t)} = \max \{0, (1 - x_{n}^{(t-1)})b_{n,k}^{(t)}\} \). Denote the fractional solution by \( y_{n,k}^{(t)F}, \forall n \in [N], k \in [K] \).
2: for all \( n \in [N] \) do
3: \( \forall n' \in [N], k \in [K], w'_{n',k}^{(t)} = \max \{0, (1 - x_{n}^{(t-1)})b_{n,k}^{(t)}\} \), if \( n' \neq n \). Otherwise \( w'_{n',k}^{(t)} = 0 \).
4: Solve LP relaxation of (3), with \( w'_{n',k}^{(t)} \)'s. Denote the optimal objective function value by \( \tilde{V}_{-n}^{(t)} \).
5: \( \Pi_{n}^{(t)F} = \tilde{V}_{-n}^{(t)} - \sum_{n' \neq n} \sum_{k} y_{n',k}^{(t)F} w_{n',k}^{(t)} \)
6: end for
7: Solve the pair of primal-dual decomposition LPs in (6) and (7) using the ellipsoid method, using Alg. 2 as a separation oracle, and derive a polynomial number of integer solutions to (3), \( y^{(t)l}, \forall l \in L \), and the corresponding decomposition coefficients, \( \beta_{l}, \forall l \in L \).
8: Choose \( y^{(t)l} \) with probability \( \beta_{l}, \forall l \in L \)
9: \( \forall n \in [N], \Pi_{n}^{(t)l} = \Pi_{n}^{(t)F} \cdot \frac{\sum_{k \in [K]} w_{n,k}^{(t)} y_{n,k}^{(t)l}}{\sum_{k \in [K]} w_{n,k}^{(t)} y_{n,k}^{(t)F}} \)
What we have done

• A $e$-approximation algorithm for one-round VM allocation

• $A_{round}$: one-round combinatorial VM auction, truthful, $e$-approximate social welfare

• $A_{online}$: online combinatorial VM auction, truthful, $(e + \frac{1}{e-1})$-competitive
Trace-derived evaluation

Setup: Google cluster trace
- 6 types of VMs, 3 types of resources
- 3 datacenters
- 3 bundles per user
- 300 ~ 3000 users
- 300 ~ 3000 rounds

Comparison among:
- Alloc: online allocation algorithm
- Auc: online auction with original decomposition method
- AucBS: online auction with binary search improvement
Simulation

- Alloc: online allocation algorithm
- Auc: online auction with original decomposition method
- AucBS: online auction with binary search improvement

Graph: Offline/online Ratio vs. Number of Users (N) for T=300, K=3, Q=3
Simulation

- With different numbers of users
  - Alloc: online allocation algorithm
  - AucBS: online auction with binary search improvement
  - Auc: online auction with original decomposition method

- With different numbers of rounds
  - Alloc: online allocation algorithm
  - AucBS: online auction with binary search improvement
  - Auc: online auction with original decomposition method

 Alloc: online allocation algorithm
 Auc: online auction with original decomposition method
 AucBS: online auction with binary search improvement
Simulation • With different numbers of users

- Alloc: online allocation algorithm
- AucBS: online auction with binary search improvement
- Auc: online auction with original decomposition method

Simulation • With different numbers of rounds

- Alloc: online allocation algorithm
- AucBS: online auction with binary search improvement
- Auc: online auction with original decomposition method

Simulation • With different numbers of datacenters (using AucBS)

- Alloc: online allocation algorithm
- Auc: online auction with original decomposition method
- AucBS: online auction with binary search improvement

Alloc: online allocation algorithm
Auc: online auction with original decomposition method
AucBS: online auction with binary search improvement
Conclusion

✦ The first online combination auction for dynamic VM market
✴ translates the online social welfare maximization problem into a series of one-round resource allocation problems
✴ translates a cooperative approximation algorithm into a truthful auction
✴ a theoretical competitive ratio \(\approx 3.30\) in typical scenarios
✦ Promising to apply this algorithmic framework in other related settings