

An Online Mechanism for Dynamic Virtual Cluster Provisioning in Geo-Distributed Clouds

Weijie Shi

Department of Computer Science
The University of Hong Kong
Email: wjshi@cs.hku.hk

Chuan Wu

Department of Computer Science
The University of Hong Kong
Email: cwu@cs.hku.hk

Zongpeng Li

Department of Computer Science
University of Calgary
Email: zongpeng@ucalgary.ca

Abstract—It is common for cloud users to require clusters of inter-connected virtual machines (VMs) in a geo-distributed IaaS cloud, to run their services. Compared to isolated VMs, key challenges on dynamic virtual cluster (VC) provisioning (computation + communication resources) lie in two folds: (1) optimal placement of VCs and inter-VM traffic routing involve NP-hard problems, which are non-trivial to solve offline, not to mention if an online efficient algorithm is sought; (2) an efficient pricing mechanism is missing, which charges a market-driven price for each VC as a whole upon request, while maximizing system efficiency or provider revenue over the entire span. This paper proposes efficient online auction mechanisms to address the above challenges. We first design SWMOA, a novel online algorithm for dynamic VC provisioning and pricing, achieving truthfulness, individual rationality, computation efficiency, and $(1 + 2 \log \mu)$ -competitiveness in social welfare, where μ is related to the problem size. Next, applying a randomized reduction technique, we convert the social welfare maximizing auction into a revenue maximizing online auction, PRMOA, achieving $O(\log \mu)$ -competitiveness in provider revenue, as well as truthfulness, individual rationality and computation efficiency. We validate the efficacy of the mechanisms through solid theoretical analysis and trace-driven simulations.

I. INTRODUCTION

With the proliferation of cloud computing, more and more individuals and businesses are resorting to cloud platforms for deploying services and running jobs. Besides purchasing individual virtual machines (VMs), significant demands arise on renting a collection of VMs and the network in-between, to create a virtual cluster (VC) with an inter-connecting virtual private network. Prominent examples include cloud CDNs built on top of virtual clusters across geo-distributed cloud data centers. Enabling technologies such as network virtualization have been studied in the past years [1][2][3][4]. Virtual cluster/network services have been provided by IaaS providers (e.g., Amazon Virtual Cluster [5]), where a user defines a VC containing several VMs and the bandwidth demand among them, and the cloud provider provisions the cluster with exclusive resources and bandwidth guarantee.

The provisioning of a virtual cluster involves VM placement, *i.e.*, assigning each VM to a location (e.g., a data center), and inter-VM traffic routing, *i.e.*, finding path(s) with available bandwidth to send the traffic from one VM to another. There

is not yet an efficient resource allocation algorithm for VC provisioning even in the offline case with all user VC requests known, due to the NP hard nature of the underlying problem. In practice, user requests arrive dynamically over time; an efficient online algorithm is in need, which allocates resources for VC provisioning on the spot, while guaranteeing long-term optimality in resource utilization and user satisfaction. The focus of recent studies on VC provisioning (a.k.a. virtual network embedding in some literature) has been on heuristic offline or online algorithm design to approximate the optimal solutions, with no analytical performance guarantee. We provide a detailed summary of the existing work in Sec. II.

Moreover, an efficient pricing mechanism is missing, to charge users for the VCs on the go. The current practice is to charge an aggregate price of VMs and bandwidth usage in a virtual cluster, based on the fixed unit prices of the computation and communication resources [5]. Such a pricing method lacks market agility to adapt to supply-and-demand changes, risking the provider's revenue as well as social welfare. As a representative market-driven mechanism, auctions have been studied in cloud computing. Compared with fixed pricing, an auction mechanism enables appropriate prices that take real-time demand and supply into consideration, avoiding overpricing or underpricing and achieving revenue or social welfare maximization. The existing cloud auctions focus on allocation of separate VMs [6], or VM bundles which demand computational resources (CPU, RAM, storage) only (but not inter-VM bandwidths) [7]. It is in fact common to send traffic between VMs in a VC, *e.g.*, large replication traffic between VMs in a distributed cloud storage system, requesting bandwidths to be allocated between the VMs as well.

To the best of our knowledge, online auction of an entire virtual cluster, including VMs and the network in-between, has not been studied. The difficulty mainly lies in the NP-hard nature of the resource allocation problem for VC provisioning, which hinders exact solutions as typically needed in designing truthful and social welfare maximizing auction mechanisms, even in the offline case. The challenge escalates when we practically target an online auction requiring timely allocation and pricing decisions on the go, while maximizing social welfare or provider revenue over the entire system span.

This paper designs efficient and competitive online auctions for on-demand provisioning and pricing of VCs deployed over

The project was supported in part by grants from Hong Kong RGC under the contracts HKU 718513 and 17204715, and by Wedge Networks as well as NSERC.

geo-distributed cloud data centers. Users arrive dynamically, specifying potential VCs to deploy with tailor-made VMs in different data centers, as well as the traffic in between, at different willingness-to-pay prices. Two online auction mechanisms are designed for social welfare maximization and provider revenue maximization, respectively. The mechanisms are designed based on an online auction framework which dynamically maintains a cost for each type of resources in each data center. Based on which payments of potential VCs are computed, the best VM placement scheme for each user is selected, and acceptance/rejection decisions are made. Our detailed contributions are summarized as follows.

First, in the design of the social welfare maximizing auction, *SWMOA*, we set a dynamic unit cost for each type of computation and communication resources at and across the data centers, which increases with the depletion of the corresponding type of resource. We find the best VC provisioning scheme for each user among the schemes indicated in his bid, which maximizes his utility, by formulating a VC provisioning linear program (LP). The LP can be reduced to a minimum cost multicommodity flow problem, which is efficiently solvable using existing algorithms. We then compute the overall cost of the obtained VC provisioning scheme and compare the cost with the willingness-to-pay from the user. A user is accepted if the VC acquired provides positive utility. We show that *SWMOA* achieves truthfulness, individual rationality, computation efficiency, and $(1 + 2 \log \mu)$ -competitiveness in social welfare, where μ is related to the problem size.

Second, the revenue maximizing online auction, *PRMOA*, is built on the basis of *SWMOA*. We use *SWMOA* to first obtain a tentative VC allocation and a payment for each user, and then re-examine each tentatively accepted bid with a randomized boosted payment to improve provider revenue. The randomized payment is carefully designed to be still below the user's corresponding true valuation with high probability, without the knowledge of the actual true valuation. In this way, the provider is able to extract almost the largest possible revenue with high probability. *PRMOA* achieves $O(\log \mu)$ -competitiveness in provider revenue, as well as truthfulness, individual rationality and computation efficiency.

In the rest of the paper, we discuss related work in Sec. II and present the system model in Sec. III. Sec. IV presents the social welfare maximizing online auction, *SWMOA*. Sec. V presents the revenue maximizing online auction, *PRMOA*. Sec. VI presents the trace-driven simulation studies and Sec. VII concludes the paper.

II. RELATED WORK

The virtual cluster provisioning (a.k.a. virtual network embedding/mapping) problem has attracted substantial research interest in recent years. Li *et al.* [8] formulate the VM placement problem and consider the traffic cost and physical machine utilization cost. Zhang *et al.* [9] study how to map VMs to servers to minimize the failure probability of the user's virtual data center (*i.e.*, maximize the reliability). Heuristic algorithms are proposed to calculate the failure probability and

minimize it efficiently. Ballani *et al.* [10] propose a pricing mechanism for VMs based on a user's bottleneck resource consumption in VM placement. Esposito *et al.* [11] solve the VC mapping problem using primal and dual decomposition. Chowdhury *et al.* [12] reduce VC mapping to link mapping which is formulated as an integer programming (IP) program, and solve the latter whenever a VC request arrives using LP relaxation and deterministic/randomized rounding. All these work on offline solutions of VC provisioning.

For handling online VC provisioning, Grandl *et al.* [13] present a scheduling algorithm to assign tasks to machines, which is essentially a multidimensional bin packing algorithm. Even *et al.* [14] study the online multicommodity-flow routing problem and the online VC mapping algorithm. Allowing violation of capacity constraints, a near-optimal online algorithm is proposed. [15] propose a quadratic IP formulation for the VC provisioning problem. Compared to these work, we propose the pricing rule to stimulate users to report their real valuation, and we use solid theoretical analysis to guarantee the performance of our algorithms in the worst case.

Auction mechanisms have been widely applied for resource allocation in various networking systems. To design a truthful auction with an NP-hard underlying allocation problem, a useful technique is to first design an approximation algorithm, and then use the critical bid rule to decide an appropriate price [16], which is an extension of the classic VCG technique [17]. Sun *et al.* [18] adopt this technique and design a dynamic spectrum auction. Wang *et al.* [6] apply the same method, and derive a collusion-resistant mechanism for cloud computing. Another approach is to resort to the LP decomposition technique [19]. Zhang *et al.* [20] and Shi *et al.* [7] design truthful auctions for dynamic VM provisioning using this method. In addition, Zhang *et al.* [21][22] design VM auctions following different approaches, the MIDR algorithm and the primal-dual framework. Fu *et al.* None of the existing cloud auctions consider the allocation of both VM computational resources and inter-VM communication resources simultaneously, which is necessary when users request VCs. In VC auctions, the underlying resource allocation problem is significantly more difficult, involving both VM placement and traffic routing decisions, calling for novel online algorithm design.

III. PROBLEM MODEL

A. Online VC Auction

Consider an IaaS cloud with P geo-distributed data centers (DCs). Let $[X]$ denote the integer set $\{1, 2, \dots, X\}$. Data center $p \in [P]$ has $\hat{A}_{p,r}$ units of type- r computational resources (like CPU, RAM and disk), $\forall r \in [R]$, where R is the number of computational resource types. The data centers are inter-connected through S gateway routers, each of which is located with one data center, and hence $S = P$. An illustration of the network is given in Fig. 1. Let \mathbb{E} be the set of links connecting gateway routers and data centers. The bandwidth capacity of link $(w_1, w_2) \in \mathbb{E}$ is \hat{d}_{w_1, w_2} , where w_1 and w_2 can be either a router in $[S]$ or a data center in $[P]$. Practically we assume the bandwidth on the link connecting a data center and

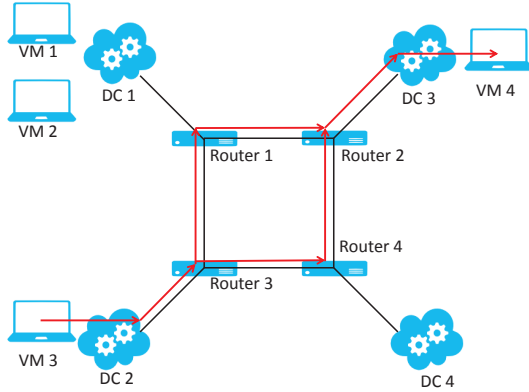


Fig. 1. An example of virtual cluster provisioning.

its collocated gateway router is enough (e.g., the link from DC 1 to Router 1 in Fig. 1) is not a bottleneck in our problem.

N users arrive on the fly, and request for virtual clusters (VCs). User $n \in [N]$ arrives at time T_n^s , submits a bid to demand a VC immediately, and releases the VC at time T_n^f . Let $T_n = T_n^f - T_n^s$ be the usage duration of user n 's VC, and $T = \max_{n \in [N]} \{T_n\}$ denote the largest usage duration among all users. The VC required by user n includes V_n tailor-made VMs, and VM v in the VC consumes $a_{v,r}^n$ units of type- r computational resource, $\forall r \in [R]$. The bandwidth demand to send traffic from VM v_1 to VM v_2 in user n 's VC is Γ_{v_1, v_2}^n , which occupies inter-datacenter bandwidth when VM v_1 and VM v_2 are located in different data centers. User n can specify several VM placement schemes with different preferences. Let \mathbb{B}_n be his set of VM placement schemes. Each scheme $\beta \in \mathbb{B}_n$ specifies the placement of VMs in his VC, represented by $z_{v,p}^{n,\beta} \in \{0, 1\}$, indicating that VM v is placed in data center p if $z_{v,p}^{n,\beta} = 1$ and not if $z_{v,p}^{n,\beta} = 0$. Together with each scheme, the user submits a valuation $b_{n,\beta}$, which is his willingness-to-pay if his VC bid is successful the VMs are allocated in the data centers specified in scheme β . Different values of $b_{n,\beta}$'s indicate his preferences among the schemes, decided by the need of his workload or services. For example, if the user is running MapReduce workloads, he may specify to place all his VMs in a selected datacenter; if the user is operating an online video service, his VMs are preferred to be located close to large population of the service users.

In summary, the bid of user n can be expressed as $(T_n^s, T_n^f, \{a_{v,r}^n\}_{v \in [V_n], r \in [R]}, \{\Gamma_{v_1, v_2}^n\}_{v_1, v_2 \in [V_n]}, \{z_{v,p}^{n,\beta}\}_{\beta \in \mathbb{B}_n, v \in [V_n], p \in [P]}, \{b_{n,\beta}\}_{\beta \in \mathbb{B}_n})$. Fig. 1 shows an example of a user requesting a VC of 4 VMs and plots one VM placement scheme.

The cloud provider acts as the auctioneer. Upon arrival of a user's bid, the provider immediately responds with whether to serve this user, which VM placement scheme to be adopted, and what price to charge this user for. The decision variables include the following: (i) $y_{n,\beta} \in \{0, 1\}, \forall n \in [N], \beta \in \mathbb{B}_n$, indicating whether user n 's request is accepted according to his placement scheme β ($y_{n,\beta} = 1$) and not otherwise ($y_{n,\beta} = 0$). At most one VM placement scheme can be accepted for each bid. (ii) $\hat{b}_n, \forall n \in [N]$, payment of user n if his bid is accepted. (iii) $f_{v_1, v_2, w_1, w_2}^n, \forall n \in [N], v_1, v_2 \in [V_n], (w_1, w_2) \in \mathbb{E}$,

indicating the routing traffic on each link, which will be illustrated in detail in Sec. III-B.

B. Goals of Mechanism Design

Our online auction design targets the following properties. (i) *Truthfulness and individual rationality*: The auction mechanism is truthful if for any user n , bidding a different valuation other than $b_{n,\beta}$ does not increase his utility, which is the difference between his valuation and his payment $b_{n,\beta} - \hat{b}_n$. Truthfulness ensures that selfish buyers are stimulated to reveal their true valuations of the VCs they demand, simplifying the bidding strategy and the auction design. Individual rationality requires that any user's utility is non-negative. (ii) *Computation efficiency*: The mechanism should run in polynomial time, in order to be practically applied in an online fashion. (iii) *Competitive in social welfare or provider revenue*: The provider's revenue equals the total payment from all users in the online auction, $\sum_{n \in [N]} \hat{b}_n$. Since the provider's revenue and the users' payment part in the aggregate user utility $(\sum_{n \in [N], \beta \in \mathbb{B}_n} b_{n,\beta} y_{n,\beta} - \sum_{n \in [N]} \hat{b}_n)$ cancel each other, the social welfare is the total valuation of accepted users, i.e., $\sum_{n \in [N], \beta \in \mathbb{B}_n} b_{n,\beta} y_{n,\beta}$. Let S_{online} denote the social welfare achieved under an online mechanism, and S_{opt} be the offline optimum social welfare. An online mechanism is c -competitive in social welfare if the ratio of S_{opt}/S_{online} is upper-bounded by c for any input instance. On the other hand, Let R_{online} be the total provider revenue obtained under the online auction. An online auction is c -competitive in provider revenue if the ratio between the offline optimal social welfare and the provider revenue achieved by the online auction, S_{opt}/R_{online} , is upper-bounded by c for any input instance. Here comparing to S_{opt} instead of the offline revenue in computing the competitive ratio in revenue is a general practice, since the optimal truthful auction generating largest revenue cannot be identified [23]. We also note that in fact no truthful auction can achieve a revenue at the amount of S_{opt} , which is only achievable when the users bid true valuations and the provider always charges users according to their bid prices. However, the latter leads to untruthful bidding, and hence a contradiction.

We next formulate the offline VC provisioning and winner determination problem, supposing all bids within system span are known and truthful bidding is guaranteed. The objective in (1) indicates social welfare maximization, whose optimal value is S_{opt} . It can be easily changed to revenue maximization by replacing the social welfare with the provider's revenue.

Here $z_{v,p}^n$ is an auxiliary variable defined in the first constraint (1a), representing if data center p is selected to host VM v in user n 's VC, which is 1 if user n 's bid is accepted, one VM placement scheme is picked, and v is placed in p according to this scheme, and 0 otherwise. f_{v_1, v_2, w_1, w_2}^n represents the allocated bandwidth on link (w_1, w_2) for the traffic flow from VM v_1 to VM v_2 in user n 's VC. Let N_t be the set of active users at time t , whose VCs are in use at t : $N_t = \{n | t \in [T_n^s, T_n^f]\}$. Constraint (1b) guarantees that at most one scheme is accepted for each user. Constraint (1c) requires that at any time the allocated computational

resources at each data center do not exceed their respective capacity. Constraints (1d) (1e) and (1f) model routing of user n 's traffic flow from VM v_1 to VM v_2 . We allow multi-path routing of each inter-VM flow. An illustration of two paths taken by the flow from VM 3 to VM 4 is given in Fig. 1. Constraint (1d) specifies that at the data center p where v_1 of user n is placed ($z_{v_1,p}^n = 1$), the total out-bound bandwidth from data center p allocated for user n 's flow from v_1 to v_2 (LHS of (1d)) should equal his specified bandwidth demand (Γ_{v_1,v_2}^n). Similarly, constraint (1e) specifies the total in-bound bandwidth at data center p equal to the bandwidth demand Γ_{v_1,v_2}^n if v_2 is placed at data center p . Constraint (1f) is the flow conservation constraint at each router for user n 's flow from v_1 to v_2 . Since routers are intermediate nodes, in-bound and out-bound flow rates should be equal. Constraint (1g) requires that the aggregated bandwidth allocated on each link in \mathbb{E} does not exceed the link capacity.

$$\begin{aligned}
& \text{maximize} \quad \sum_{n \in [N]} \sum_{\beta \in \mathbb{B}_n} b_{n,\beta} y_{n,\beta} & (1) \\
& \text{s.t.} \quad z_{v,p}^n = \sum_{\beta \in \mathbb{B}_n} y_{n,\beta} z_{v,p}^{n,\beta} \quad \forall n \in [N], v \in [V_n], p \in [P] & (1a) \\
& \quad \sum_{\beta \in \mathbb{B}_n} y_{n,\beta} \leq 1 \quad \forall n \in [N] & (1b) \\
& \quad \sum_{n \in N_t} \sum_{v \in [V_n]} z_{v,p}^n a_{v,r}^n \leq \hat{A}_{p,r} \quad \forall p \in [P], r \in [R], t \in [T] & (1c) \\
& \quad \sum_{w:(p,w) \in \mathbb{E}} f_{v_1,v_2,p,w}^n = \Gamma_{v_1,v_2}^n z_{v_1,p}^n \quad \forall p \in [P], n \in [N], v_1, v_2 \in [V_n] & (1d) \\
& \quad \sum_{w:(w,p) \in \mathbb{E}} f_{v_1,v_2,w,p}^n = \Gamma_{v_1,v_2}^n z_{v_2,p}^n \quad \forall p \in [P], n \in [N], v_1, v_2 \in [V_n] & (1e) \\
& \quad \sum_{w:(w,\nu) \in \mathbb{E}} f_{v_1,v_2,w,\nu}^n = \sum_{w:(\nu,w) \in \mathbb{E}} f_{v_1,v_2,w}^n & (1f) \\
& \quad \quad \quad \forall n \in [N], v_1, v_2 \in [V_n], \nu \in [S] & (1f) \\
& \quad \sum_{n \in N_t} \sum_{v_1, v_2 \in [V_n]} f_{v_1,v_2,w_1,w_2}^n \leq \hat{d}_{w_1,w_2} \quad \forall (w_1, w_2) \in \mathbb{E}, t \in [T] & (1g) \\
& \quad y_{n,\beta} \in \{0, 1\} \quad \forall n \in [N], \beta \in \mathbb{B}_n & (1h) \\
& \quad f_{v_1,v_2,w_1,w_2}^n \geq 0 \quad \forall n \in [N], v_1, v_2 \in [V_n], (w_1, w_2) \in \mathbb{E} & (1i)
\end{aligned}$$

The offline optimization problem in (1) is a mixed integer linear program and can be proven NP-hard by a reduction to the knapsack problem. The proof is given in Appendix A.

Theorem 1. *The offline optimization problem (1) is NP-hard.*

Hence, it is very challenging even to solve the VC provisioning problem in an offline fashion, with all information known.

C. Preliminaries for Online Mechanism Design

To design an efficient online auction, we introduce a few concepts which will be useful later. We regard each computational resource at each data center and the bandwidth on each link as different resources. Thus there are $M = PR + E$ types of resources in total (here $E = |\mathbb{E}|$), *i.e.*, PR types of computational resources at different data centers and E bandwidth resources on different links. We use $r(n, m, t)$ to denote the amount of type m resource consumed by user n at time t , which is a value depending on VM placement and traffic routing decisions made for the user's bid. For example, let m correspond to computational resource r at data center p . If the bid is accepted and one or more VMs are allocated

in data center p , then the amount of resource m consumed by user n in a $t \in T_n$ is: $r(n, m, t) = \sum_{v \in [V_n]} z_{v,p}^n a_{v,r}^n$. If m denotes the bandwidth resource on link $(w_1, w_2) \in \mathbb{E}$, then the bandwidth consumed on the link by the user during T_n is: $r(n, m, t) = \sum_{v_1, v_2 \in [V_n]} f_{v_1, v_2, w_1, w_2}^n$.

For ease of presentation, we normalize the scale of the capacity of each type of resource, so that the total capacity of resource m at each time t is equal to 1, *i.e.*, $\hat{A}_{p,r} = 1, \hat{d}_{w_1, w_2} = 1, \forall p \in [P], r \in [R], (w_1, w_2) \in \mathbb{E}$. We then divide all the demands of resource by $\hat{A}_{p,r}$ or \hat{d}_{w_1, w_2} , respectively. Then the resource constraints (1c) and (1g) require that $\sum_{n \in [N]} r(n, m, t) \leq 1$, for all $m \in [M]$ and t in $[T]$. For example, if a user needs 2 VMs on one DC, and each VM uses 4 CPUs, then the amount of CPU resource consumed by this user on this DC is 8. And if there are totally 320 CPUs in that DC, then $r(n, m, t) = 1/40$ after normalization.

We make two assumptions. *First*, a user's valuation is approximately proportional to the amount of resources his VC requires, *i.e.*, the per unit per time slot valuation is bounded: $1 \leq \frac{b_{n,\beta}}{Mr(n,m,t)T_n} \leq F$, where F is a constant, $\forall n, m, t$ and $r(n, m, t) \neq 0$. We also assume that the ratio between the highest and lowest valuations is bounded: $\frac{\max_{n \in [N], \beta \in \mathbb{B}_n} \{b_{n,\beta}\}}{\min_{n \in [N], \beta \in \mathbb{B}_n} \{b_{n,\beta}\}} \leq F$. Second, there is an upper bound on the amount of resources required by each VC in each time slot at each data center or link, *i.e.*, $r(n, m, t) \leq \frac{1}{\log \mu}$ where $\mu = 2MTF + 1$, which implies that the resource demand of each individual user is small as compared to the total capacity of each data center. Here μ (related to F) is an important parameter to appear in our competitive ratios. We summarize important notation in Table I, for ease of reference.

TABLE I
KEY NOTATION

N	# of users	$[X]$	integer set $\{1, \dots, X\}$
P	# of data centers	b_n	user n 's payment
M	$M = PR + \mathbb{E} $	T_n	duration of user n 's VC
R	# of computational resource types		
T_n^s	arriving time of user n		
T_n^f	departure time of user n		
T	$\max_n \{T_n\}$		
$b_{n,\beta}$	user n 's valuation if scheme β is accepted		
\mathbb{B}_n	user n 's set of VM placement schemes		
$r(n, m, t)$	demand of resource m at t by user n		
$\lambda_m(t, n)$	load factor of resource m at t before user n		
F	constant related to valuation variation		
μ	$2MTF + 1$		
$c_m(t, n)$	cost of resource m at t before user n		
$C(n, \beta)$	total cost for scheme β of user n		
V_n	# of VMs in user n 's VC		
$a_{v,r}^n$	amount of resource r required by VM v of user n		
$\Gamma_{v,v'}^n$	traffic from VM v to VM v' of user n		
$y_{n,\beta}$	scheme β of user n is accepted or not		
$z_{v,p}^{n,\beta}$	VM v is assigned to data center p or not in β of user n		
\mathbb{E}	set of links		
\hat{d}_{w_1, w_2}	bandwidth capacity of link (w_1, w_2)		
$\hat{A}_{p,r}$	capacity of type r resource at DC p		
f_{v_1, v_2, w_1, w_2}^n	traffic from v_1 to v_2 of user n on link (w_1, w_2)		

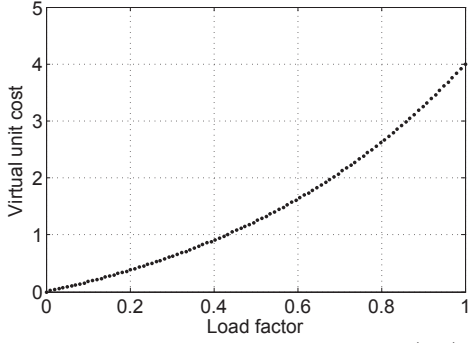


Fig. 2. An example of the virtual unit cost function $c_m(t, n)$, where $\mu = 5$.

IV. SOCIAL WELFARE MAXIMIZING ONLINE AUCTION

We now design a social welfare maximizing online mechanism, *SWMOA*.

Main Idea. At a high level, our strategy of finding a competitive VC allocation solution for problem (1) is to maintain a dynamic *virtual unit cost* $c_m(t, n)$ for each type of resource m at each time slot $t \in [T]$ (containing future time slots with resource reservation), before user n arrives. For any VM placement scheme β of the user, we calculate a total virtual cost $C(n, \beta)$ based on the unit costs and the amount of resources the scheme consumes. Scheme β is a candidate to be accepted if its cost is smaller than its valuation: $C(n, \beta) \leq b_{n, \beta}$.

The virtual unit cost $c_m(t, n)$ is designed to increase when the remaining available amount of resource m at time t depletes. Thus a higher virtual cost indicates that type of resource is scarce. As a result, the cost of a VC is higher if it consumes resources in shortage, reducing the likelihood that it is chosen by the online algorithm. On the other hand, by comparing the total cost with the valuation of VC, the online mechanism can pick users with higher valuation compared with the amount of resources consumed.

We define the *load factor* (before user n arrives) of resource m for each time $t \in [T]$ to be the amount of already allocated (reserved) resources in the respective time slot before user n arrives: $\lambda_m(t, n) = \sum_{n' < n, n' \in [N]} r(n', m, t)$, where $n' < n$ indicates that user n' arrives earlier than user n . We define the virtual unit cost of resource m at time t , computed before user n arrives (i.e., before counting in resource consumption of user n in each time slot), as follows:

$$c_m(t, n) = \mu^{\lambda_m(t, n)} - 1, \forall t \in [T], n \in [N], m \in [M]. \quad (2)$$

$c_m(t, n)$ is designed to increase exponentially with the increase of consumed resources. This allows the user to consume resources freely when resources are abundant, since the cost is close to 0 when $\lambda_m(t, n)$ is small. However when the load factor is close to 1, the cost increases fast to a very large value, so that the user is almost forbidden to use the resource in shortage. The idea of the unit cost formula comes from the study of packing/covering LPs [24]. We show an example of the virtual unit cost function in Fig. 2.

We define the total cost that user n incurs if his scheme β is accepted as the sum of the costs of all resources consumed:

$$C(n, \beta) = \sum_{m \in [M]} \sum_{t \in T_n} c_m(t, n) r(n, m, t), \forall n \in [N], \beta \in \mathbb{B}_n. \quad (3)$$

The mechanism accepts a user if there exists a scheme β with cost less than its valuation. If there are more than one schemes satisfying the condition, we choose the one with the largest user utility: $b_{n, \beta} - C(n, \beta)$, which is the scheme maximizing user's utility under truthful bidding when we set the payment to be $\tilde{b}_n = C(n, \beta)$.

Cost Computation. The total cost $C(n, \beta)$ for each VM placement scheme β includes the cost of computation resources and the cost of bandwidth consumption. Suppose $c_{r,p}(t, n)$ denote the virtual unit cost for resource r in data center p at time t . The total cost of computational resource is $\sum_{t \in T_n} \sum_{r \in [R]} \sum_{p \in [P]} \sum_{v \in [V_n]} c_{r,p}(t, n) a_{v,r} z_{v,p}^n$. The bandwidth cost differs depending on the routing plan of flows among VMs in user n 's VC. We find the best traffic routing plan for VM placement scheme β , that minimizes the total bandwidth cost, by solving the LP (4).

The LP identifies the best routing paths and bandwidth allocation for flows among VMs in user n 's VC (decided by f_{v_1, v_2, w_1, w_2}^n 's), when the VMs are placed in data centers according to scheme β . Here $c_{w_1, w_2}(t, n)$ represents the virtual unit cost for bandwidth on link (w_1, w_2) at time t , before user n submits his bid. $\sum_{v_1, v_2 \in [V_n]} f_{v_1, v_2, w_1, w_2}^n$ is the overall bandwidth consumed on link (w_1, w_2) by flows among user n 's VMs. p_{v_1} and p_{v_2} represent the assigned data center for VM v_1 and VM v_2 , respectively. Constraints (4a) and (4b) ensure that the total out-bound/in-bound traffic from p_{v_1} to p_{v_2} equals the traffic demand from v_1 to v_2 . Constraint (4c) is the flow conservation constraint. Solving (4) gives us the optimal routing as well as the minimum total bandwidth cost under scheme β . In fact, (4) is a minimum cost multi-commodity flow problem which allows fractional flows, and can be solved using efficient algorithms [25] in $O((E + V_n^2)P)$ time.

$$\begin{aligned} \text{minimize} \quad & \sum_{(w_1, w_2) \in \mathbb{E}} \sum_{t \in T_n} (c_{w_1, w_2}(t, n) \sum_{v_1, v_2 \in [V_n]} f_{v_1, v_2, w_1, w_2}^n) \quad (4) \\ & \sum_{w: (p_{v_1}, w) \in \mathbb{E}} f_{v_1, v_2, p_{v_1}, w}^n = \Gamma_{v_1, v_2}^n \quad \forall v_1, v_2 \in [V_n] \quad (4a) \\ & \sum_{w: (w, p_{v_2}) \in \mathbb{E}} f_{v_1, v_2, w, p_{v_2}}^n = \Gamma_{v_1, v_2}^n \quad \forall v_1, v_2 \in [V_n] \quad (4b) \\ & \sum_{w: (w, \nu) \in \mathbb{E}} f_{v_1, v_2, w, \nu}^n = \sum_{w: (\nu, w) \in \mathbb{E}} f_{v_1, v_2, \nu, w}^n \quad \forall v_1, v_2 \in [V_n], \nu \in [S] \quad (4c) \\ & f_{v_1, v_2, w_1, w_2}^n \geq 0 \quad \forall (w_1, w_2) \in \mathbb{E}, v_1, v_2 \in [V_n] \quad (4d) \end{aligned}$$

Online Auction. The social welfare maximizing auction is summarized in Alg. 1. Upon receiving a new user's bid, we calculate the minimum total cost for each β by solving (4). We choose the scheme β^* with largest user utility and accept the user if his utility is positive. Then we upgrade the virtual unit costs. We present properties achieved by *SWMOA* in Thm. 2, with proof given in Appendix B. The payment of each winning user is computed based on virtual unit costs before counting in resources required by the user. Such independence of payments from the bids guarantees truthfulness of the mechanism.

Theorem 2. *The online auction mechanism SWMOA (Alg. 1) is truthful and individually rational, never violates the resource constraints, runs in $O((E + V_n^2)PB)$ time in each round, and achieves a $(1 + 2 \log \mu)$ -competitive ratio in social*

welfare, where $B = \max_{n \in [N]} |\mathbb{B}_n|$ is the largest number of schemes submitted by any user.

Algorithm 1 Social Welfare Maximization Online Auction *SWMOA*

```

1: if user  $n \in [N]$  arrives then
2:   for all schemes  $\beta \in \mathbb{B}$  do
3:     Solve the minimum cost multi-commodity flow
       problem (4) and find optimal routing
4:     Compute the total cost  $C(n, \beta)$ 
5:     Calculate the utility  $b_{n,\beta} - C(n, \beta)$  of scheme  $\beta$ 
6:   end for
7:   Let  $\beta^*$  be the scheme with maximal utility
8:   if  $C(n, \beta^*) \leq b_{n,\beta^*}$  then
9:     Accept user  $n$  with scheme  $\beta^*$ , i.e., set  $y_{n,\beta^*} = 1$ ,
        $y_{n,\beta} = 0, \forall \beta \neq \beta^*$ 
10:    Charge user  $n$  the payment  $\tilde{b}_n = C(n, \beta^*)$ 
11:    Update unit costs  $c_m(t, n + 1)$  according to (2)
12:   else
13:     Reject user  $n$ , i.e., set  $y_{n,\beta} = 0, \forall \beta \in \mathbb{B}_n$ 
14:   end if
15: end if

```

V. REVENUE MAXIMIZING ONLINE AUCTION

We next design a revenue maximizing online auction *PRMOA*, inspired by a randomized reduction technique [23]. **Main Idea.** We use *SWMOA* to obtain a tentative VC allocation and a payment for each user, and then re-examine each tentatively accepted bid with a randomized boosted payment to improve revenue. The boosted payment is carefully designed to be still below the user’s corresponding true valuation with high probability, without the knowledge of the true valuation.

Obviously, the ideal payment which maximizes the revenue is a value just below the user’s true valuation. However, a straightforward payment rule, e.g., 90% of the user’s bid price, breaks the truthfulness of the mechanism, since users can gain more from under-reporting their valuation. The payment has to be independent of the users’ bids in order to guarantee truthfulness. So the principle of designing a revenue maximization auction is to set the payment close to the valuation but without knowing the valuation in the decision process.

Recall the payment decision process of *SWMOA*: We calculate the cost for a scheme $C(\beta, n)$, accept and charge the user $\tilde{b}_n = C(n, \beta)$ if his valuation for this scheme is larger. We use the original threshold $C(n, \beta)$ as a foundation, and add a randomized value δ on it: $\tilde{b}_n = C(\beta, n) + \delta$. Setting δ is the key in achieving revenue competitiveness. If δ is too small, not enough revenue is extracted. On the other hand, if δ is too large, the payment may exceed the valuation, forcing our auction mechanism to reject the user, which brings great loss to the revenue. In order to upper-bound the occurrence probability of either cases, we let $\delta = 0$ with probability 1/2, which means that our algorithm will be exactly the same as *SWMOA*, both in allocation and payment, with probability 1/2. With the other half probability, we try to set the payment as close to the valuation as possible, using a power-of-2 rule: Let

$\delta = 2^i b_{min}$ with probability $\frac{1}{2 \log F}$, for $i = 1, 2, \dots, \log F$, where $b_{min} = \min_{n \in [N], \beta \in \mathbb{B}_n} b_{n,\beta}$ is the minimal valuation among all users. This guarantees that with probability $\frac{1}{2 \log F}$, the payment is at least half of the valuation but still less than the valuation: $b_{n,\beta}/2 \leq \tilde{b}_n \leq b_{n,\beta}$, which is the ideal payment we want – this will be verified in our proof of Thm. 3.

Another point worth noting is, since we increase the payment \tilde{b}_n , some users will be rejected in *PRMOA*, while accepted in *SWMOA*. In those cases, we still update the virtual unit costs as if these users were accepted. That is, the virtual unit costs are updated just like running *SWMOA*, no matter whether a user is actually accepted by *PRMOA* or not. In this way, we guarantee that the costs of the user schemes are always the same as in *SWMOA*, which is needed to show the competitiveness of our mechanism.

Online Auction. The online auction is summarized in Alg. 2, with properties given in Thm. 3. and the proof is given in Appendix C.

Algorithm 2 Provider Revenue Maximization Online Auction *PRMOA*

```

1: if user  $n \in [N]$  arrives then
2:   Set  $\delta = 0$  with probability 1/2, and  $\delta = 2^i b_{min}$  with
       probability  $\frac{1}{2 \log F}$ , for  $i = 1, 2, \dots, \log F$ 
3:   for all schemes  $\beta \in \mathbb{B}$  do
4:     Solve the minimum cost multi-commodity flow
       problem (4) and find optimal routing
5:     Compute the total cost  $C(n, \beta)$ 
6:     Calculate the utility  $b_{n,\beta} - C(n, \beta)$ 
7:   end for
8:   Let scheme  $\beta^*$  be the scheme with maximal utility
9:   if  $C(n, \beta^*) \leq b_{n,\beta^*}$  then
10:    Update unit costs  $c_m(t, n + 1)$  according to (2),
       supposing user  $n$  is accepted
11:   end if
12:   Set payment  $\tilde{b}_n = C(n, \beta^*) + \delta$ 
13:   if  $\tilde{b}_n \leq b_{n,\beta^*}$  then
14:     Accept user  $n$  with scheme  $\beta^*$ . Charge the pay-
       ment  $\tilde{b}_n$ .
15:   else
16:     Reject user  $n$ 
17:   end if
18: end if

```

Theorem 3. *The online auction mechanism PRMOA (Alg. 2) is truthful and individually rational, never violates the resource constraints, and achieves a $O(\log \mu)$ -competitive ratio in provider revenue in expectation in polynomial time.*

VI. PERFORMANCE EVALUATION

We evaluate our online auction mechanisms using data collected on the Google cluster [26]. We translate each job containing multiple tasks, each demanding different amounts of resources, in the Google cluster data into a VC request with VMs of different resource compositions. There are $R = 3$ types of computational resources (CPU, RAM and disk), and $P = 12$ data centers connected following the topology

of Google data center network [27]. For each user, V_n is randomly chosen within [2, 30]. We generate 5 to 10 schemes for each user, placing VMs at randomly chosen data centers in each scheme. The default system span is $T = 100$. Users arrive following a Poisson distribution with parameter $\lambda = 50$, and their VC usage durations are randomly chosen within [5, 10]. The amount of resources consumed by a VM $a_{v,r}^n$ is set according to the resource demand of each task in Google cluster data. The traffic between VMs $\Gamma_{v,v'}^n$ is randomly generated within a range [0, 1]. The overall resource capacities, \hat{A}, \hat{d} , is set to be approximately [0.2, 0.5] fraction of the respective overall user resource demand, and randomly distributed onto data centers and links. The valuation $b_{n,\beta}$ is uniformly randomly chosen within the interval decided by the constant F , whose default value is 10. We repeat each experiment for 50 times to obtain the average results.

A. Evaluation of SWMOA

We first compare our online algorithm *SWMOA*, Alg. 1, with the offline optimum. We calculate the offline optimal social welfare by solving (1) exactly, and divide it by the social welfare achieved by *SWMOA* to obtain the ratio, under different numbers of users and different values of F . Recall that the parameter μ in our theoretical competitive ratio is related to F , i.e., $\mu = 2MTF + 1$. Fig. 3 shows the ratio is smaller for a larger number of users N . This is because the more users in the system, the less impact a bad decision for a previous user has on later users. In contrast, less users make it more difficult to achieve optimal allocation since each decision involves more resources. We can also see that for larger values of F , the performance ratio is larger. The theoretical competitive ratio, $O(MTF)$, implies such a result. Larger F represents a larger range of user valuation. In online resource allocation, serving one user means that we may need to reject some user in the future, whose valuation may be much higher when F is larger, leading to low competitiveness of the online mechanism. Fig. 4 shows that the ratio is larger when the number of data centers P is larger, which is consistent with our theoretical competitive ratio as well, since $M = PR + |\mathbb{E}|$. Intuitively, more data centers make the allocation harder since more different types of resources need to be considered.

Next we compare *SWMOA* with a heuristic adaptive algorithm TV-VNE in [28], by comparing the social welfare ratio they each achieve when comparing to the offline optimum. We cannot find any auction algorithms designed for DC provisioning. We can only find TV-VNE with a similar model. The design idea of TV-VNE is to provision VCs according to the amount of consumed resources and always try to find a load-balancing allocation. The core technique used by TV-VNE is a heuristic weighted function which achieves a tradeoff between the computational resource usage and the bandwidth usage. When a new user request arrives, TV-VNE chooses an allocation and routing plan which maximizes this function, which intends to balance the usage of all data centers and link bandwidths. This idea is similar to our online algorithm in some sense, but in a heuristic way. Fig. 5 shows that our

algorithm consistently outperforms TV-VNE. We further observe that the performance of TV-VNE does not improve much for larger N , reflecting that its heuristic allocation cannot efficiently adjust resource utilization with the increase of users. In contrast, *SWMOA* performs better when the number of user is large, which makes it more suitable for cloud systems at a large scale. In Fig. 6, we prepare two special scenarios: computation intensive (CI-Case) and bandwidth intensive (BI-Case). In the computation intensive case, computation resource demands of VCs are 2 times larger than the default setting. In the bandwidth intensive scenario, bandwidth demands are 2 times larger than the default. We see that *SWMOA* outperforms TV-VNE in all cases. The advantages in the special cases are larger, implying that our algorithm better handles extreme scenarios by dynamically adjusting to user demands.

B. Evaluation of PRMOA

We first compare *PRMOA* in Alg. 2 with the offline optimum. We divide the offline optimal social welfare by the revenue achieved with *PRMOA* to calculate the performance ratio. Fig. 7 and Fig. 8 show the performance of *PRMOA* under different values of F and P . The trends are similar with those for *SWMOA*: larger N , smaller F and smaller P bring better performance. This is not a coincidence because *PRMOA* uses *SWMOA* as its foundation, and in half of the time adopts the same payment and allocation. In the other half of the time, the success of the attempt to “guess” user’s valuation is independent of other parameters.

Next we compare the provider revenue achieved by *PRMOA* and TV-VNE. However, TV-VNE is just a VC provisioning algorithm without pricing. Fortunately, it is easy to design a payment for it according to the “unit virtual cost” used in the TV-VNE algorithm, which is the only payment rule making the mechanism truthful. Fig. 9 and Fig. 10 show that *PRMOA* performs better in all scenarios. The advantages are even larger than what we show when comparing *SWMOA* with TV-VNE. This validates the efficiency of our algorithm specifically optimized for chasing high revenue.

VII. CONCLUSION

This paper presents efficient and competitive online auction mechanisms for on-demand provisioning and pricing of virtual clusters, taking both computational resources and communication resources into consideration. First, we design an online social welfare maximizing auction, *SWMOA*, which sets a carefully designed virtual unit cost for each type of resources on the go, and decides winners and bid-independent payments based on the total resource cost. Next, we design a revenue maximizing online auction, *PRMOA*, which runs *SWMOA* as a basis, and boosts payments following a carefully designed random distribution, to pursue higher revenue. The mechanisms are truthful, individually rational, time-efficient, and guarantee a $(2 \log \mu + 1)$ competitive ratio in social welfare and a $O(\log \mu)$ competitive ratio in provider revenue in expectation, respectively. Our extensive simulation studies validate our theoretical analysis, and show good performance of our mechanisms in various scenarios.

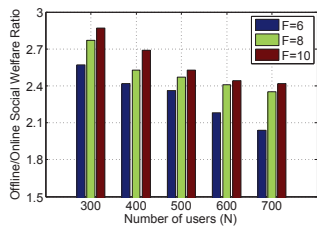


Fig. 3. Performance of *SWMOA* under different values of F

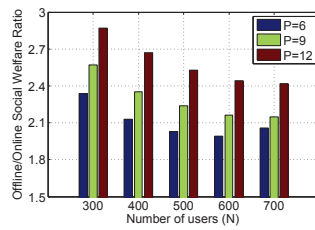


Fig. 4. Performance of *SWMOA* under different values of P

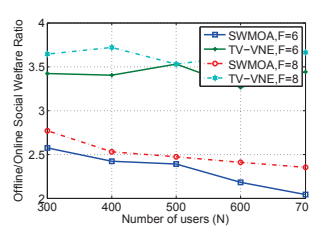


Fig. 5. Comparison between *SWMOA* and *TV-VNE*

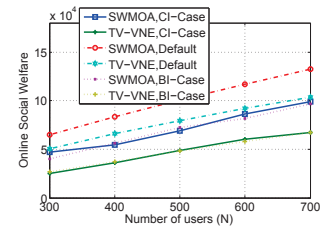


Fig. 6. Comparison between *SWMOA* and *TV-VNE* under special scenarios

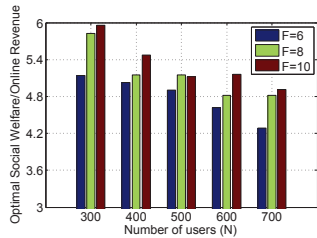


Fig. 7. Performance of *PRMOA* under different values of F

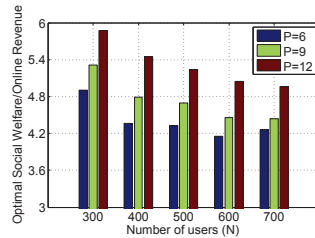


Fig. 8. Performance of *PRMOA* under different values of P

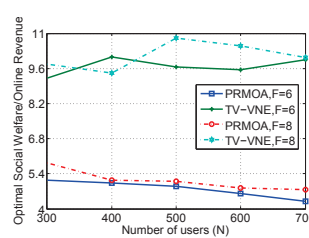


Fig. 9. Comparison between *PRMOA* and *TV-VNE*

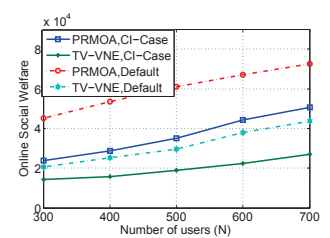


Fig. 10. Comparison between *PRMOA* and *TV-VNE* under special scenarios

REFERENCES

- [1] C. Guo, G. Lu, H. J. Wang, S. Yang, C. Kong, P. Sun, W. Wu, and Y. Zhang, "Secondnet: a Data Center Network Virtualization Architecture with Bandwidth Guarantees," in *Proc. of ACM Co-Next*, 2010.
- [2] H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron, "Towards Predictable Datacenter Networks," in *ACM SIGCOMM Computer Communication Review*, 2011.
- [3] H. Rodrigues, J. R. Santos, Y. Turner, P. Soares, and D. Guedes, "Gate-keeper: Supporting Bandwidth Guarantees for Multi-tenant Datacenter Networks," in *WIOV*, 2011.
- [4] T. Benson, A. Akella, A. Shaikh, and S. Sahu, "CloudNaaS: a Cloud Networking Platform for Enterprise Applications," in *Proc. of ACM SOCC*, 2011.
- [5] "Amazon Virtual Cluster," <http://docs.aws.amazon.com/redshift/latest/mgmt/managing-clusters-vc.html>.
- [6] Q. Wang, K. Ren, and X. Meng, "When Cloud meets eBay: Towards Effective Pricing for Cloud Computing," in *Proc. of IEEE INFOCOM*, 2012.
- [7] W. Shi, L. Zhang, C. Wu, Z. Li, and F. Lau, "An Online Auction Framework for Dynamic Resource Provisioning in Cloud Computing," in *Proc. of ACM SIGMETRICS*, 2014.
- [8] X. Li, J. Wu, S. Tang, and S. Lu, "Lets Stay Together: Towards Traffic Aware Virtual Machine Placement in Data Centers," in *Proc. of IEEE INFOCOM*, 2014.
- [9] Q. Zhang, M. F. Zhani, M. Jabri, and R. Boutaba, "Venice: Reliable Virtual Data Center Embedding in Clouds," in *Proc. of IEEE INFOCOM*, 2014.
- [10] H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron, "The Price Is Right: Towards Location-Independent Costs in Datacenters," in *Proc. of ACM HotNet*, 2011.
- [11] F. Esposito and I. Matta, "A Decomposition-based Architecture for Distributed Virtual Network Embedding," *ACM SIGCOMM DCC Workshop*, 2014.
- [12] N. M. K. Chowdhury, M. R. Rahman, and R. Boutaba, "Virtual Network Embedding with Coordinated Node and Link Mapping," in *Proc. of IEEE INFOCOM*, 2009.
- [13] R. Grandl, G. Ananthanarayanan, S. Kandula, S. Rao, and A. Akella, "Multi-Resource Packing for Cluster Schedulers," in *Proc. of ACM SIGCOMM*, 2014.
- [14] G. Even and M. Medina, "Online Multi-Commodity Flow with High Demands," in *Approximation and Online Algorithms*, 2013.
- [15] Z. Cai, F. Liu, N. Xiao, Q. Liu, and Z. Wang, "Virtual Network Embedding for Evolving Networks," in *Proc. of IEEE GLOBECOM*, 2010.
- [16] D. Lehmann, L. I. O'Callaghan, and Y. Shoham, "Truth Revelation in Approximately Efficient Combinatorial Auctions," *Journal of the ACM (JACM)*, 2002.
- [17] W. Vickrey, "Counterspeculation, Auctions, and Competitive Sealed Tenders," *The Journal of Finance*, 1961.
- [18] Q. Sun, Q. Wang, K. Ren, and X. Jia, "Fair Pricing in the Sky: Truthful Frequency Allocation with Dynamic Spectrum Supply," in *Proc. of IEEE ICNP*, 2014.
- [19] R. Lavi and C. Swamy, "Truthful and near-Optimal Mechanism Design via Linear Programming," in *Proc. of FOCS*, 2005.
- [20] L. Zhang, C. Wu, and Z. Li, "Dynamic Resource Provisioning in Cloud Computing: A Randomized Auction Approach," in *Proc. of IEEE INFOCOM*, 2014.
- [21] X. Zhang, C. Wu, Z. Li, and F. C. Lau, "A Truthful (1- ϵ)-Optimal Mechanism for On-demand Cloud Resource Provisioning," in *Proc. of IEEE INFOCOM*, 2015.
- [22] X. Zhang, Z. Huang, C. Wu, Z. Li, and F. Lau, "Online Auctions in IaaS Clouds: Welfare and Profit Maximization with Server Costs," in *Proc. of ACM SIGMETRICS*, 2015.
- [23] A. Madry, "Reducing Truth-Telling Online Mechanisms to Online Optimization," in *Proc. of ACM STOC*, 2003.
- [24] N. Buchbinder and J. Naor, "The Design of Competitive Online Algorithms via a Primal: Dual Approach," *Foundations and Trends in Theoretical Computer Science*, 2009.
- [25] A. Madry, "Faster Approximation Schemes for Fractional Multicommodity Flow Problems via Dynamic Graph Algorithms," in *Proc. of ACM STOC*, 2010.
- [26] "Google Cluster Data," <https://code.google.com/p/googleclusterdata/>.
- [27] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu *et al.*, "B4: Experience with a Globally-Deployed Software Defined WAN," in *Proc. of ACM SIGCOMM*, 2013.
- [28] L. Bo, T. Huang, Z.-K. Wang, J.-Y. Chen, Y.-J. Liu, and L. Jiang, "Adaptive Scheme Based on Status Feedback for Virtual Network Mapping," *The Journal of China Universities of Posts and Telecommunications*, 2011.

APPENDIX A
PROOF OF THEOREM 1

Proof. We prove NP-hardness by reduction to the 0-1 knapsack problem with capacity $\{A_p\}$, and items with weight $\{a_n\}$ and valuation $\{b_n\}$. We design a VC allocation problem, including datacenters' capacities $\{A_p\}$ and VMs consuming $\{a_n\}$ resources, with valuation $\{b_n\}$. Then we can solve the knapsack problem if we can solve the problem (1). \square

APPENDIX B
PROOF OF THEOREM 2

Proof. First we analyze the running time: solving the multi-commodity problem needs $O((E+V_n^2)P)$ time, and we do this for all users' schemes. So the total time is $O((E+V_n^2)PBN)$. The auction mechanism is individual rational since the payment is always no larger than the valuation. Notice that the unit cost of each types of resource is not affected by the bidding valuation $b_{n,\beta}$, and thus $C(\beta, n)$ is not influenced by users' bidding valuation. The algorithm chooses the scheme which maximizes $b_{n,\beta} - C(n, \beta)$. If a false bidding does not change the chosen scheme β , the user's utility is not changed since the payment only depends on $C(\beta, n)$. Otherwise, if a false bidding causes a different scheme be picked β' . Then the user's utility under β' cannot be larger, since the new utility is $b_{n,\beta'} - C(\beta', n)$, no larger than the utility under the original scheme β . So the mechanism is truthful.

In order to prove the competitive ratio, we prove 3 claims. (1) $\forall m, t$, the resource constraints are never violated. Or, $\sum_n r(n, m, t) \leq 1$. If user n is rejected, we can assume a virtual scheme with valuation, payment and resource consumption all be 0, without affecting the proof. (2) Total valuation achieved by the online algorithm is lower bounded: $2 \log \mu S_{online} \geq \sum_{t,m} c_m(t, N+1)$, where $S_{online} = \sum_{n \in [N]} b_{n,\beta}$ is the social welfare achieved under the online solution. (3) Let β^* be the scheme chosen for the user in offline optimal solution, and $S_{opt} = \sum_{n \in [N]} b_{n,\beta^*}$ be the social welfare under the offline solution. Then S_{opt} is upper bounded: $S_{opt} \leq S_{online} + \sum_{t,m} c_m(t, N+1)$.

Proof of claim (1): Let user n be the first user breaking the constraint of resource m at time t with scheme β . Then $\lambda_m(t, n) > 1 - r(n, m, t)$. By assumption that $r(n, m, t) \leq \frac{1}{\log \mu}$, we get: $c_m(t, n) \geq \mu/2 + 1 = MTF$. Therefore, by $\frac{b_{n,\beta}}{Mr(n,m,t)T_n} \leq F$, we have $c_m(t, n)r(n, m, t) \geq MTFr(n, m, t) \geq b_{n,\beta}$, contradicting the assumption.

Proof of claim (2): By induction on N , for $N = 0$ the claim is true. For a new user which is rejected, the value of either side of the inequality is not changed. So we only need to show that for any n , if he is accepted with β , then $\sum_t \sum_m (c_m(t, n+1) - c_m(t, n)) \leq 2 \log \mu b_{n,\beta}$. For any m , we have $c_m(t, n+1) - c_m(t, n) = \mu^{\lambda_m(t,n)} (2^{\log \mu r(n,m,t)} - 1)$. Because $2^x - 1 \leq x$ for $x \in [0, 1]$, we have $c_m(t, n+1) - c_m(t, n) \leq \log \mu (c_m(t, n)r(n, m, t) + r(n, m, t))$. Then $\sum_t \sum_m (c_m(t, n+1) - c_m(t, n)) \leq \log \mu \sum_t \sum_m (c_m(t, n)r(n, m, t) + r(n, m, t)) \leq \log \mu (C(n, \beta) + TMr(n, m, t)) \leq 2 \log \mu b_{n,\beta}$.

Proof of claim (3): The fact that β^* is not chosen gives us: $b_{n,\beta} - C(n, \beta) \geq b_{n,\beta^*} - C(n, \beta^*)$, where β is the scheme chosen in the online solution. So $\sum_n b_{n,\beta^*} \leq \sum_n b_{n,\beta} + \sum_n \sum_{m,t} c_m(t, n)r(n, m, t)$. Notice that $c_m(t, n) \leq c_m(t, N+1)$, and $\sum_n r(n, m, t) \leq 1, \forall m, t$. We have $S_{opt} \leq S_{online} + \sum_{t,m} c_m(t, N+1)$. Together with claim (2), we prove that the competitive ratio is $2 \log \mu + 1$. \square

APPENDIX C
PROOF OF THEOREM 3

Proof. First we prove truthfulness. Since the scheme maximizing $b_{n,\beta} - C(\beta, n)$ is also the scheme maximizing user utility $b_{n,\beta} - b_n$, the user cannot increase his utility by changing his receiving scheme. The payment is independent of $b_{n,\beta}$, so if the user does not change the scheme β , his utility remains the same. Individual rationality is guaranteed since we reject users with $b_{n,\beta} < \tilde{b}_n$. Polynomial running time is guaranteed just as in SWMOA.

We consider several categories of the users. Let A be the set of users accepted by our algorithm. Let Q be the set of users with $C(\beta, n) \leq b_{n,\beta}$, i.e., accepted by SWMOA. We further divide Q into Q_1 and Q_2 . Users in Q_1 have $b_{n,\beta} \geq b_{n,\beta^*}/2$, and users in Q_2 have $b_{n,\beta} < b_{n,\beta^*}/2$, where β^* is the scheme provided in the offline solution. Let P be the set of users accepted by the offline solution but not by SWMOA ($C(\beta, n) > b_{n,\beta}$). We need to show that the total revenue $\sum_{n \in A} \tilde{b}_n$ is comparable to the offline optimal social welfare, which is bounded by $\sum_{n \in P} b_{n,\beta^*} + \sum_{n \in Q_1} b_{n,\beta^*} + \sum_{n \in Q_2} b_{n,\beta^*}$. We prove 3 claims: (1) $\sum_{n \in P} b_{n,\beta^*} \leq (4 \log \mu + 2)E[\sum_{n \in A} \tilde{b}_n]$. (2) $\sum_{n \in Q_1} b_{n,\beta^*} \leq (8 \log F)E[\sum_{n \in A} \tilde{b}_n]$. (3) $\sum_{n \in Q_2} b_{n,\beta^*} \leq (8 \log \mu + 4)E[\sum_{n \in A} \tilde{b}_n]$.

Proof of claim (1): First we show $\sum_{n \in P} b_{n,\beta^*} \leq (2 \log \mu + 1) \sum_{n \in Q} C(n, \beta)$: This comes from the social welfare competitiveness of SWMOA. Notice that $\sum_{n \in P} b_{n,\beta^*}$ is a set of feasible offline solutions. And if all users in Q bid $C(n, \beta)$ instead of $b_{n,\beta}$, SWMOA will still choose the users in set Q , exactly the same as if they bid $b_{n,\beta}$. So according to the social welfare competitiveness, a feasible solution is bounded by $(2 \log \mu + 1)$ times the online solution. Next we show $\sum_{n \in Q} C(n, \beta) \leq 2E[\sum_{n \in A} \tilde{b}_n]$. Set Q does not depend on our choose of δ . With $1/2$ probability $\tilde{b}_n = C(\beta, n)$, so for all $n \in Q$, it belongs to A . So the expected value of $\sum_{n \in A} \tilde{b}_n$ is at least half of the value of $\sum_{n \in Q} C(n, \beta)$.

Proof of claim (2): $\forall n \in Q_1$, there is $1/2 \log F$ probability that $b_{n,\beta} \geq \tilde{b}_n \geq b_{n,\beta}/2 \geq b_{n,\beta^*}/4$. So any $n \in Q_1$ contributes at least $b_{n,\beta^*}/8 \log F$ in \tilde{b}_n .

Proof of claim (3): The fact that our algorithm choose β instead of β^* gives us: $b_{n,\beta^*}/2 \geq b_{n,\beta} - C(\beta, n) \geq b_{n,\beta^*} - C(\beta^*, n)$. So $b_{n,\beta^*} \leq 2C(\beta^*, n)$. This means, if we let users in Q_2 bidding β^* with valuation $b_{n,\beta^*}/2$, they will be rejected. Again according to the social welfare competitiveness, $\sum_{n \in Q_2} b_{n,\beta^*}/2 \leq (2 \log \mu + 1) \sum_{n \in Q} C(n, \beta)$. Together with $\sum_{n \in Q} C(n, \beta) \leq 2E[\sum_{n \in A} \tilde{b}_n]$, shown in the proof of claim (1), we prove the claim. 3 claims together prove the theorem. \square