Dynamic Resource Provisioning in Cloud Computing: A Randomized Auction Approach

Linquan Zhang*, Zongpeng Li*, Chuan Wu†
*Department of Computer Science, University of Calgary, {linqzhan,zongpeng}@ucalgary.ca
†Department of Computer Science, The University of Hong Kong, cwu@cs.hku.hk

***Abstract*—This work studies resource allocation in a cloud market through the auction of Virtual Machine (VM) instances. It generalizes the existing literature by introducing combinatorial auctions of heterogeneous VMs, and models dynamic VM provisioning. Social welfare maximization under dynamic resource provisioning is proven NP-hard, and modeled with a linear integer program. An efficient α-approximation algorithm is designed, with $\alpha \sim 2.72$ in typical scenarios. We then employ this algorithm as a building block for designing a randomized combinatorial auction that is computationally efficient, truthful in expectation, and guarantees the same social welfare approximation factor α. A key technique in the design is to utilize a pair of tailored primal and dual LPs for exploiting the underlying packing structure of the social welfare maximization problem, to decompose its fractional solution into a convex combination of integral solutions. Empirical studies driven by Google Cluster traces verify the efficacy of the randomized auction.**

I. INTRODUCTION

The cloud computing paradigm offers users rapid on-demand access to computing resources such as CPU, RAM and storage, with minimal management overhead. Recent commercial cloud platforms, exemplified by Amazon EC2 [1], Microsoft Azure and Linode [2], organize a shared resource pool for serving their users. Virtualization technologies help cloud providers pack their resources into different types of virtual machines (VMs), for allocation to cloud users. For example, Tab. I illustrates a number of VMs types available at Amazon EC2 [1].

TABLE I
AMAZON EC2 VIRTUAL MACHINE INSTANCE TYPES

VM type	CPU	Memory	Storage
m1.medium	2 EC2 Compute Units	3.75 GB	410 GB
m1.xlarge	8 EC2 Compute Units	15 GB	1680 GB
c1.medium	5 EC2 Compute Units	1.7 GB	350 GB
c1.xlarge	20 EC2 Compute Units	7 GB	1680 GB
m2.xlarge	6.5 EC2 Compute Units	17.1 GB	420 GB
hi1.4xlarge	35 EC2 Compute Units	60.5 GB	2048 GB

The underlying reason for such VM heterogeneity is that a cloud user's job often requires cooperation among multiple VM instances, each with its own focus and forte. For example, social games [3] and enterprise applications [4] are often composed of a front-end web server tier, a load balancing tier and a back-end data storage tier, each suited for execution on a VM that is abundant in a particular type of resource: bandwidth, CPU, or storage.

Unfortunately, existing allocation mechanisms in cloud markets either are based on fixed pricing, which is economically inefficient, or resort to simple, static auctions that treat VMs as type-oblivious commodities. More specifically, it is usually assumed that either a single type of VMs exists in the cloud market, or VMs are substitutes in that a high-end VM is equivalent to a number of low-end VMs, *e.g.*, a Type II (2 $\times$ Core, 2 GB RAM, 40 GB Disk) VM equals two Type I (1 $\times$ Core, 1 GB RAM, 20 GB Disk) VMs [5], [6]. Such type-oblivious VM auctions do not handle the existing VM heterogeneity in today's cloud computing platforms, and can not be adapted in a straightforward way to do so.

This work generalizes such simple auction design in the cloud market by proposing combinatorial auctions that are expressive enough for cloud users to request bundles of VM instances belonging to distinct types. It further departs from the existing literature by explicitly modelling the dynamic provisioning of VM instances from cloud resources. Under *static provisioning*, the cloud assembles its available resources into different types of VMs based on simple heuristics or historical VM demand patterns, before the auction starts. Under *dynamic provisioning*, the cloud conducts VM assembling in an online fashion upon receiving VM bundle bids [5], targeting maximum possible social welfare given the current bid profile.

We show that social welfare maximization under dynamic resource provisioning is NP-hard due to its combinatorial optimization nature. Nonetheless, such maximization can be cast into a linear integer program, based on which we design an efficient cooperative primal-dual approximation algorithm that achieves a small approximation factor α. The factor α depends on the diversity of resource demands within all bids submitted by one user, the normalized volume of the cloud resource pool and the number of resource types, and is shown to be close to 2.72 in most practical settings. However, such an approximation algorithm assumes that truthful bids are given for free, and is not applicable in a cloud market with strategic users driven by their own economical interests, who may not voluntarily reveal their true evaluation of a desired VM bundle. The crux of many auction design in the literature indeed lies in the careful custom tuning of the auction mechanism, for eliciting truthful bids from selfish buyers [7], [8]. A well-known type of truthful auctions is the celebrated Vickrey-Clarke-Groves (VCG) mechanism, which is proven to be the only type of auctions that can simultaneously guarantee

This work is supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC), and grants from Hong Kong RGC under the contracts HKU 717812 and HKU 718513.

truthfulness and economical efficiency (social welfare maximization). Unfortunately, a VCG auction requires solving the NP-hard problem of social welfare optimization multiple times for calculating externalities as user payments, and becomes computationally infeasible as the system size grows.

We design a randomized combinatorial auction for dynamic resource provisioning, which is computationally efficient, truthful in expectation, and surprisingly, simultaneously guarantees the same social welfare approximation factor of α as does the cooperative approximation algorithm. Note that the latter assumes truthful bidding for free, and can afford to focus on algorithmically maximizing the social welfare by ignoring potential strategic bidding from selfish cloud users, while the former is bound to pay close attention at the same time to eliciting truth-telling, a property not usually attained without a compromise in social welfare [7], [8], [9].

Below is a high-level overview of the structure of the randomized VM auction. We first simulate a fractional VCG auction based on the linear programming relaxation (LPR) of the social welfare maximization integer program (IP). Then we utilize a pair of tailored primal and dual linear programs (LPs) to decompose the optimal fractional solution of the LPR into a weighted combination of integer solutions to the IP. This pair of LPs exploit the underlying packing nature of the social welfare maximization IP, and are solved using the ellipsoid algorithm with the cooperative α-approximation algorithm acting as a separation oracle. In this process, we prove and utilize the fact that the approximation algorithm also verifies an integrality gap of α between the IP and the LPR. Each integer solution is selected randomly with probability equal to its weight calculated during the decomposition, and contains information for instructing the cloud provider to conduct both VM provisioning and VM allocation. Fractional VCG payments calculated at the beginning are finally scaled down by the approximation factor α, for ensuring that the resulting randomized auction inherits its truthfulness from the fractional VCG auction.

We have implemented the randomized auction and evaluated it against traces from Google Cluster Data [10] through extensive simulation studies. We found that dynamic provisioning usually outperforms static provisioning in terms of social welfare by a ratio on the order of 50%. An exciting observation is that the primal-dual cooperative approximation algorithm approaches optimal social welfare within a gap of 10% in all the scenarios tested, performing much better beyond the theoretically proven approximation factor of $\alpha \sim 2.72$ in typical scenarios. Consequently, the randomized auction can provide better guarantee in social welfare guarantee in practice. Such empirical observation further motivates our discussions on improving the cloud's revenue by scaling fractional VCG payments with a ratio smaller than 2.72, for striking a flexible balance between absolute truthfulness and seller revenue.

In the rest of the paper, we discuss related work in Sec. II, and introduce the system model in Sec. III. We design a primal-dual approximation algorithm and analyze its approximation ratio in Sec. IV, and further utilize it in Sec. V to design the randomized VM auction, evaluated through simulation studies in Sec. VI. Sec. VII concludes the paper.

II. Previous Literature

As an efficient resource allocation mechanism in economic markets, auctions have been studied substantially over the past few decades. The celebrated VCG auction [11], [12], [13] represents a general truthful auction framework, under which buyers have no incentive to submit falsified bids. A VCG auction requires solving the social welfare maximization problem to optimum, for calculating payments of winning buyers. Consequently, it becomes computationally infeasible when exact social welfare maximization is NP-hard, as is the case for dynamic cloud resource provisioning in this work. A VCG auction loses its truthful property if approximation algorithms are applied for social welfare maximization.

For non-VCG style of auction design, custom techniques specific to the problem at hand are required for guaranteeing truthfulness of the resulting auction mechanism. In a sequence of recent work that originated from theoretical computer science [14], [8], [9], [15], a decomposition technique is designed for translating fractional solutions to integer solutions, for packing type integer programs. The key technique lies in a pair of tailored primal and dual LPs that exploit such packing property, which can be solved with an efficient approximation algorithm that can verify the integrality gap between the IP and its LPR. To the authors' knowledge, this work is the first in the field of cloud computing that successfully applies such a primal-dual decomposition technique.

The design of VM auctions has been studied in a series of work in recent cloud computing literature. For instance, Zhang *et al.* [16] study the resource allocation problem with realtime demand arrivals, and propose a truthful online auction-based allocation policy. Auctions also take an important role in the exchange of computing resources among members in a federated cloud in number of recent work [17], [18].

Zaman *et al.* [6] propose an auction-based VM allocation mechanism, named CA-GREEDY, for the case of static resource provisioning, where the cloud provider has a predetermined number of VMs for sale in each VM type. However, the approximation ratio of the mechanism they designed is rather large, especially when a large number of VM instances are provisioned, a rather common scenario in real-world cloud computing. The authors also consider the dynamic provisioning case [5], and present a truthful mechanism. No guarantee is provided on the social welfare approximation ratio of their mechanism, though. In contrast, the randomized VM auction we design is not only truthful, but also achieves close-to-optimum social welfare maximization.

III. System Model and Preliminaries

We consider auction-based resource provisioning and VM allocation in a cloud market. The cloud provider (*auctioneer*) leases resources packed in VMs to cloud users through round-by-round auctions. The cloud provider has a pool of t types of resources. The total amount of type k resource is c_k. The

cloud provider offers m types of VMs, VM_1, ..., VM_m. A VM_j instance consumes r_j^k amount of type k resource.

Let $\mathcal{B}$ denote the set of cloud users, acting as *bidders* in the auction. Each user $i \in \mathcal{B}$ can submit as many bids as it wishes. Let B_i denote the set of bids submitted by cloud user i, and $\{B_i\}_{i\in\mathcal{B}}$ contains all bids from all users. Each bid specifies a desired VM bundle $\mathcal{S} = (n_1^{\mathcal{S}}, \cdots, n_m^{\mathcal{S}})$ along with the bidding price $b_i(\mathcal{S})$, where $n_j^{\mathcal{S}}$ is the number of VM_j instances that cloud user i requests in $\mathcal{S}$. We assume that a single bid alone does not exceed the capacity constraint for any type of resource, *i.e.*, $\forall 1 \le k \le t, R_k \triangleq \max_{i\in\mathcal{B},\mathcal{S}\in B_i} \sum_{j=1}^m n_j^{\mathcal{S}} r_j^k < c_k$. Let $x_i(\mathcal{S})$ be a binary variable indicating whether user i wins bundle $\mathcal{S}$. Then $\boldsymbol{x} = \{x_i(\mathcal{S})\}_{i\in\mathcal{B},\mathcal{S}\in B_i}$ represents an allocation outcome. Let $v_i(\boldsymbol{x})$ denote the true valuation of cloud user i, known only to i itself. Let Π_i be the priced charged to a winning user i. Then the utility u_i for user i is:

$$u_i(B_i, B_{-i}) = \begin{cases} v_i(\boldsymbol{x}) - \Pi_i & \text{if } i \text{ receives a VM bundle} \\ 0 & \text{otherwsie} \end{cases} \quad (1)$$

where $B_{-i} = \{B_j\}_{j\in\mathcal{B}\setminus\{i\}}$ is a set of all the bids except B_i.

Definition *A (randomized) auction is* truthful *(in expectation) if for any bidder i, reporting its true valuation in the bid maximizes its (expected) utility, regardless of the bids submitted by other bidders.*

We adopt the XOR bidding language, in which a user can win at most one bid even if it submits multiple bids [15], leading to the first constraint for VM allocation:

$$\sum_{\mathcal{S}\in B_i} x_i(\mathcal{S}) \le 1, \forall i \in \mathcal{B} \quad (2)$$

The finite supply of each type of cloud resource translates into the capacity constraint at the cloud provider:

$$\begin{gathered} \sum_{i\in\mathcal{B}} \sum_{\mathcal{S}\in B_i} x_i(\mathcal{S}) n_j^{\mathcal{S}} \le N_j, \forall 1 \le j \le m \\ \sum_{j=1}^m N_j r_j^k \le c_k, \forall 1 \le k \le t \end{gathered} \quad (3)$$

where N_j is the number of VM_j instances provisioned. The two groups of inequalities in (3) can be merged into an equivalent, more compact capacity constraint:

$$\sum_{i\in\mathcal{B}} \sum_{\mathcal{S}\in B_i} x_i(\mathcal{S}) (\sum_{j=1}^m n_j^{\mathcal{S}} r_j^k) \le c_k, \forall 1 \le k \le t \quad (4)$$

The social welfare maximization problem can now be formulated:

$$\text{maximize} \quad DP(\mathcal{B}) = \sum_{i\in\mathcal{B}} \sum_{\mathcal{S}\in B_i} b_i(\mathcal{S}) x_i(\mathcal{S}) \quad (5)$$

subject to:

$$\sum_{\mathcal{S}\in B_i} x_i(\mathcal{S}) \le 1, \qquad \forall i \in \mathcal{B} \quad (5a)$$

$$\sum_{i\in\mathcal{B}} \sum_{\mathcal{S}\in B_i} x_i(\mathcal{S}) (\sum_{j=1}^m n_j^{\mathcal{S}} r_j^k) \le c_k, \qquad \forall 1 \le k \le t \quad (5b)$$

$$x_i(\mathcal{S}) \in \{0,1\}, \quad \forall i \in \mathcal{B}, \mathcal{S} \in B_i \quad (5c)$$

where $DP(\mathcal{B})$ denotes the objective function of IP (5). Note that in a truthful auction, the bid $b_i(\mathcal{S})$ can be assumed to be user i's valuation of VM bundle $\mathcal{S}$.

Theorem 1. *The social welfare maximization problem defined in IP (5) is NP-hard.*

Proof: We construct a polynomial-time reduction to IP (5) from the knapsack problem, a classic combinatorial optimization problem that is proven NP-hard [19]:

$$\max_{\boldsymbol{x}} \{ \sum_{i=1}^n v_i x_i \text{ subject to} \sum_{i=1}^n w_i x_i \le W, x_i \in \{0,1\} \}$$

Given an instance $A = (v_1, \cdots, v_n, w_1, \cdots, w_n, n, W)$ of the knapsack problem, we map it to an instance of the social welfare maximization problem $A' = (|\mathcal{B}| = n, |B_i| = 1, b_i(\mathcal{S}) = v_i, t = 1, \sum_{j=1}^m n_j^{\mathcal{S}} r_j^k = w_i, c_i = W)$, in which each cloud user submits a single bid, and the resource pool contains one type of resource. Such mapping can clearly be done in polynomial time. If there exists an algorithm solving the social welfare maximization problem A', then it solves the corresponding knapsack problem A as well, and *vice versa*. Consequently, the knapsack problem can be viewed as a special case of the social welfare maximization problem, which must be NP-hard as well. □

Theorem 1 reveals that solving IP (5) is NP-hard, and is computationally infeasible for a large input. Nonetheless, we may consider the *LP relaxation* of IP (5) by relaxing its last constraint (5c) to [1]:

$$x_i(\mathcal{S}) \ge 0, \forall i \in \mathcal{B}, \mathcal{S} \in B_i \quad (5c')$$

Introducing dual variable vectors $\boldsymbol{y}$ and $\boldsymbol{z}$ to constraints $(5a)$ and $(5b)$ respectively, we can formulate the dual of the LPR, to be used in the primal-dual algorithm design in Sec. IV:

$$\text{minimize} \quad \sum_{i\in\mathcal{B}} y_i + \sum_{k=1}^t c_k z_k \quad (6)$$

subject to:

$$y_i + \sum_{k=1}^t \sum_{j=1}^m n_j^{\mathcal{S}} r_j^k z_k \ge b_i(\mathcal{S}) \qquad \forall i \in \mathcal{B}, \mathcal{S} \in B_i \quad (6a)$$

$$y_i \ge 0, z_k \ge 0 \quad \forall i \in \mathcal{B}, 1 \le k \le t \quad (6b)$$

IV. A Primal-dual Cooperative Approximation Algorithm

We first design a polynomial-time approximation algorithm for the social welfare maximization problem in IP (5), by assuming that truthful bids are already known and targeting a small approximation ratio in social welfare. Such a cooperative approximation algorithm serves as an important building block in the design of the randomized VM auction in Sec. V, which further elicits truthful bids from strategic cloud users.

A. The Primal-Dual Approximation Algorithm

We design a greedy primal-dual algorithm for IP (5), partially inspired by the primal-dual framework due to Briest *et*

[1]Constraint $x_i(\mathcal{S}) \le 1, \forall i \in \mathcal{B}, \mathcal{S} \in B_i$ is redundant (implied by (5a) and (5c')) and removed from the LPR.

al. [20] and the classic dual fitting technique in approximation algorithm design [21], as shown in Algorithm 1. Based on a certain *value per unit resource*, the algorithm iteratively selects the current best bid from the remaining users $\mathcal{B}$ who have not received any VM bundle yet. This bid is appended to the solution set, and its corresponding user is removed from $\mathcal{B}$. Meanwhile the algorithm updates the dual variables $\boldsymbol{y}$ and $\boldsymbol{z}$, along with the primal variable $\boldsymbol{x}$, to reflect changes in set $\mathcal{B}$.

The first part of Algorithm 1 (line 2-6) initializes the primal variable $\boldsymbol{x}$ as well as the dual variables $\boldsymbol{y}$ and $\boldsymbol{z}$. Specifically, it sets $\boldsymbol{x}$ to 0 (no VM is allocated at the beginning). Values of $\boldsymbol{y}$ and $\boldsymbol{z}$ are initialized to 0 and $1/c_k$, respectively. While other positive values are also possible, choosing 0 and $1/c_k$ simplifies the algorithm analysis, as later discussed in the algorithm analysis in the Appendix.

A `while` loop (line 9-19) iteratively refines the primal and dual variables in $\boldsymbol{x}$, $\boldsymbol{y}$ and $\boldsymbol{z}$. It has two stopping conditions: $\sum_{k=1}^{t} c_k z_k \geq t\exp(\Lambda-1)$ and $\mathcal{B}=\emptyset$. The first ensures the feasibility of the generated primal solution $\boldsymbol{x}$, as discussed in the proof of Theorem 2. The second terminates the loop and hence Algorithm 1 when every cloud user has received a bundle of VMs. Since the size of the candidate set $\mathcal{B}$ decrements by one in each iteration, the `while` loop is executed at most $|\mathcal{B}|$ times.

$\sum_{k=1}^{t}\sum_{j=1}^{m} n_k^{\mathcal{S}} r_j^k z_k$ in line 13 can be viewed as the weighted total resource requested by bid $\mathcal{S}$, with z_k acting as a weight for the type k resource. Thus $\frac{b_i(\mathcal{S})}{\sum_{k=1}^{t}\sum_{j=1}^{m} n_k^{\mathcal{S}} r_j^k z_k}$ can be interpreted as the value for a unit-weight resource. For each cloud user i, the `for` loop in lines 10-12 searches for a bid with the maximum bidding price, from all bids that user i submits. Line 13 selects the best bid $\mathcal{S}_\mu$ with the maximum unit resource value from the pre-selection results in lines 10-12 across all cloud users. Lines 14-15 update the primal and dual variables and the set $\mathcal{B}$. In particular, we update the dual variable $\boldsymbol{y}$ using $y_\mu = b_\mu(\mathcal{S}_\mu)$ so that $\sum_\mu y_\mu = p$ in all iterations. This helps ensure primal optimality when the `while` loop terminates upon $\mathcal{B}=\emptyset$. Lines 16-18 update the dual variable $\boldsymbol{z}$ to reflect changes to the available resources.

B. Solution Feasibility and Approximation Ratio

We first show feasibility of solutions returned by Algorithm 1. At the end of the τ-th iteration of the `while` loop, let y_i^τ, z_k^τ be the dual variables, and p_τ be the primal objective.

Theorem 2. *Algorithm 1 computes a feasible solution to IP (5).*

Proof: Values in x are initialized to 0 (line 4) and updated to 1 only (line 14), so the solution is always binary valued. Therefore, constraint $(5c)$ will not be violated by Algorithm 1. Constraint $(5a)$ will not be violated either because once Algorithm 1 finds a VM bundle for cloud user $i \in \mathcal{B}$, no more bundles are allocated to i in the future.

Let us examine the second constraint $(5b)$. Suppose that the solution is feasible so far. Let $\tilde{\mathcal{S}} \in B_{\tilde{i}}$ be the first set that breaks the feasibility when added to the current solution, say, in iteration τ. That is, $\exists 1 \leq k \leq t$, such that

Algorithm 1 The Primal-Dual Approximation Algorithm

1: // Initialization
2: $\Lambda = \min_{1\leq k\leq t} c_k/R_k$;
3: $p=0$; $\mathcal{U}=\emptyset$;
4: $\forall i, \forall S: x_i(\mathcal{S})=0$;
5: $\forall i: y_i = 0$;
6: $\forall k: z_k = 1/c_k$;
7:
8: // Iterative update of primal and dual variables:
9: **while** $\sum_{k=1}^{t} c_k z_k < t\exp(\Lambda-1)$ AND $\mathcal{U}\neq\mathcal{B}$ **do**
10: **for all** $i \in \mathcal{B}\setminus\mathcal{U}$ **do**
11: $\mathcal{S}_i = \arg\max_{\mathcal{S}\in B_i}\{b_i(\mathcal{S})\}$;
12: **end for**
13: $\mu = \arg\max_{i\in\mathcal{B}\setminus\mathcal{U}}\{\frac{b_i(\mathcal{S}_i)}{\sum_{k=1}^{t}\sum_{j=1}^{m} n_j^{\mathcal{S}_i} r_j^k z_k}\}$;
14: $x_\mu(\mathcal{S}_\mu)=1$; $y_\mu = b_\mu(\mathcal{S}_\mu)$;
15: $p = p + b_\mu(\mathcal{S}_\mu)$; $\mathcal{U}=\mathcal{U}\cup\{\mu\}$;
16: **for all** $1\leq k\leq t$ **do**
17: $z_k = z_k\cdot(t\exp(\Lambda-1))^{(\sum_{j=1}^{m} n_j^{\mathcal{S}_\mu} r_j^k)/(c_k-R_k)}$;
18: **end for**
19: **end while**

$$\sum_{\mathcal{S}'\in\Gamma}\sum_{j=1}^{m} n_j^{\mathcal{S}'} r_j^k \leq c_k$$
$$\sum_{j=1}^{m} n_j^{\tilde{\mathcal{S}}} r_j^k + \sum_{\mathcal{S}'\in\Gamma}\sum_{j=1}^{m} n_j^{\mathcal{S}'} r_j^k \geq c_k$$

where Γ is the family of sets added to the solution before set $\tilde{\mathcal{S}}$. Since each single bid cannot exceed the capacity constraint, *i.e.*, $c_k > R_k \geq \sum_{j=1}^{m} n_j^{\tilde{\mathcal{S}}} r_j^k$, we have

$$\sum_{\mathcal{S}'\in\Gamma}\sum_{j=1}^{m} n_j^{\mathcal{S}'} r_j^k \geq c_k - R_k \Rightarrow \sum_{\mathcal{S}'\in\Gamma}\sum_{j=1}^{m} n_j^{\mathcal{S}'} r_j^k/(c_k-R_k) \geq 1$$

and that leads to:

$$c_k z_k^{\tau-1} = (t\exp(\Lambda-1))^{\sum_{\mathcal{S}'\in\Gamma}\sum_{j=1}^{m} n_j^{\mathcal{S}'} r_j^k/(c_k-R_k)} \geq t\exp(\Lambda-1)$$

which satisfies the first stopping condition in line 14. This implies that iteration $\tau-1$ is the last iteration, and $\tilde{\mathcal{S}}$ would not be added to the solution at all. □

Even if the primal solution is always feasible during the execution, the dual is not necessarily so. The following lemma shows that the dual variables can be made feasible through scaling by a carefully chosen factor. Such posterior dual scaling is known as *dual fitting* in the primal-dual optimization literature, and has proven effective in helping pursue good approximation ratios in algorithm design [21].

Lemma 1. *If $(y^{\tau-1}, z^{\tau-1})$ is the (possibly infeasible) dual solution at the beginning of the τ-th iteration, then $(y^{\tau-1}, \epsilon f(z^{\tau-1},\mathcal{S}_\tau) z^{\tau-1})$ is a feasible solution to the dual (6), where $f(z,\mathcal{S}) \triangleq b_i(\mathcal{S})/(\sum_{k=1}^{t}\sum_{j=1}^{m} n_j^{\mathcal{S}} r_j^k z_k)$, $\epsilon \triangleq \max_{\mathcal{S}_1,\mathcal{S}_2\in B_i, i\in\mathcal{B}, k\in[1,t]} \sum_{j=1}^{m} n_j^{\mathcal{S}_1} r_j^k / \sum_{j=1}^{m} n_j^{\mathcal{S}_2} r_j^k$.*

Please refer to Appendix A for the proof of Lemma 1.

Employing the dual fitting result in Lemma 1 and LP duality, we next prove that Algorithm 1 guarantees an α-approximation

of social welfare, where $\alpha = 1 + \epsilon\frac{\Lambda}{\Lambda-1}(et^{1/(\Lambda-1)} - 1)$. In practice, the volume of a cloud provider's resource pool is substantially larger than a single user demand, *i.e.*, $\Lambda \gg 1$. The number of resource types t is a small constant (3 to 5). Consequently, we can conduct the following quantitative estimation on the approximation ratio:

$$\lim_{\Lambda\to\infty} \alpha = \lim_{\Lambda\to\infty} (1 + \epsilon\frac{\Lambda}{\Lambda-1}(et^{1/(\Lambda-1)} - 1)) = 1 + \epsilon(e-1)$$

If we further consider the case where each user only submits one bid, then $\epsilon = 1$, and the approximation ratio α is close to $e \approx 2.72$, as illustrated in the 3D plot of the function $\alpha = 1 + \epsilon\frac{\Lambda}{\Lambda-1}(et^{1/(\Lambda-1)} - 1)$ in Fig. 1.

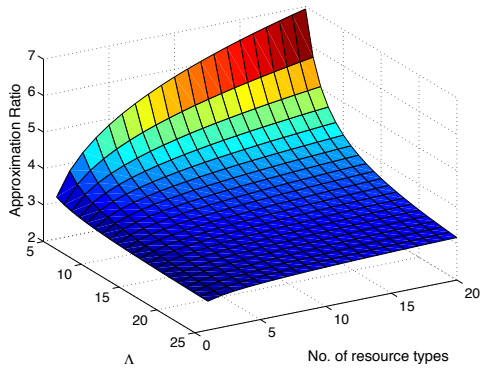


Fig. 1. The approximation ratio α quickly decreases as Λ increases, and closely approaches $e \sim 2.72$ as long as the number of resource types t is not too large and each user only submits one bid.

Theorem 3. *Algorithm 1 computes an α-approximate solution to IP (5) in polynomial-time, where $\alpha = 1+\epsilon\frac{\Lambda}{\Lambda-1}(et^{1/(\Lambda-1)}-1)$.*

Please refer to Appendix B for the proof of Theorem 3.

V. A Randomzied Auction Mechanism

Capitalizing on Algorithm 1 for approximate social welfare maximization under dynamic resource provisioning, we now design a randomized combinatorial VM auction that achieves the same social welfare approximation ratio α, while simultaneously ensuring truthful bidding from cloud users. Algorithm 2 outlines the key steps in the randomized auction mechanism.

Algorithm 2 A Randomized Combinatorial VM Auction

1: **Simulating the fractional VCG auction.**
2: — Compute the fractional VCG allocation $\boldsymbol{x}^*$ and payment $\boldsymbol{\Pi}^F$, through solving the LPR of IP (5).
3: **Decomposing fractional solution into integer solutions**
4: — Decompose the scaled down fractional solution $\boldsymbol{x}^*/\alpha$ to a convex combination of integer solutions, *i.e.*, $\boldsymbol{x}^*/\alpha = \sum_{l\in\mathcal{I}} \beta_l \boldsymbol{x}(l)$, through solving a pair of primal-dual LPs in (8) and (9) using the ellipsoid method, leveraging Algorithm 1 as a separation oracle.
5: **Randomized VM allocation**
6: — Select each $\boldsymbol{x}(l)$ randomly with probability β_l.
7: **Charging scaled fractional VCG prices**
8: — for each winning cloud user $i \in \mathcal{B}$: charge a price $\Pi_i = \Pi_i^F/\alpha$.

A. The Fractional VCG Auction

Theorem 1 reveals that solving IP (5) to optimal is NP-hard, implying that applying the VCG auction for truthfulness is computationally expensive. We first resort to a fractional version of the VCG auction for achieving both computational efficiency (polynomial time complexity) and economic efficiency (social welfare maximization), by applying the VCG mechanism to the LPR instead of IP (5).

The optimal solution $\boldsymbol{x}^*$ to the LPR constitutes the VM allocation solution in the fractional VCG auction. The fractional VCG payment for user i equals i's externality, or the difference in social welfare with and without i's bid [9], [11]:

$$\Pi_i^F = V_i - \sum_{i'\neq i, i'\in\mathcal{B}} \sum_{\mathcal{S}\in B_{i'}} b_{i'}(\mathcal{S}) x_{i'}^*(\mathcal{S}) \tag{7}$$

where V_i is the optimal $DP^F(\mathcal{B})$ to the LPR when cloud user i bids zero.

The VM bundle allocation scheme in $\boldsymbol{x}^*$ has fractional instead of binary values and is hence not practically applicable. This is to be resolved using the primal-dual decomposition technique, in Sec. V-B.

B. Decomposing the Fractional Solution

We first prepare for the decomposition by showing that Algorithm 1 verifies the integrality gap between IP (5) and the LPR in the sense that the integrality gap is also bounded by α. This is true because for any bidding profile, Algorithm 1 computes an integer solution whose social welfare is at least $1/\alpha$ times the optimal solution to the LPR, due to the following two facts: (i) the approximation ratio does not depend on the bidding prices $b_i(\mathcal{S}), \forall i \in \mathcal{B}, \mathcal{S} \in B_i$; (ii) the ratio is proven through using d/p_ω as an upper bound.

$$\text{Integrality gap} = LPR^*/DP(\mathcal{B})^* \leq d/p_\omega = \alpha$$

where $DP(\mathcal{B})^*$ is the value of the optimal solution to IP (5). The inequality is due to $LPR^* \leq d$ and $p_\omega \leq DP(\mathcal{B})^*$. Thus d/p_ω also works as an upper bound of the integrality gap.

We next decompose $\boldsymbol{x}^*$ into a convex combination of integer solutions, using a LP duality based decomposition technique for packing type of optimization problems due to Carr *et al.* [14] and Lavi *et al.* [8]. Our goal is to find β_l and $\boldsymbol{x}(l)$ such that $\boldsymbol{x}^*/\alpha = \sum_{l\in\mathcal{I}} \beta_l \boldsymbol{x}(l)$, where $\mathbb{Z}(DP) = \{\boldsymbol{x}(l)\}_{l\in\mathcal{I}}$ is the set of integer solutions to IP (5), $\mathcal{I}$ is the index set, and $\beta_l \geq 0, \sum_{l\in\mathcal{I}} \beta_l = 1$. Since the integrality gap is at most α, there exists at least one integer solution, *e.g.*, $DP(\mathcal{B})^*$, dominating the scaled down fractional solution. Consequently, scaling down the fractional solution $\boldsymbol{x}^*$ by α can guarantee the existence of such a decomposition.

The following primal and dual LPs are solved for decomposing $\boldsymbol{x}^*$:

$$\text{Primal:} \quad \text{minimize} \quad \sum_{l\in\mathcal{I}} \beta_l \tag{8}$$

subject to:

$$\sum_{l\in\mathcal{I}} \beta_l x_i(\mathcal{S},l) = x_i^*(\mathcal{S})/\alpha \quad \forall i\in\mathcal{B}, \mathcal{S}\in B_i \quad (8a)$$

$$\sum_{l\in\mathcal{I}} \beta_l \geq 1 \quad (8b)$$

$$\beta_l \geq 0 \qquad \forall l\in\mathcal{I} \quad (8c)$$

$$\text{Dual:} \quad \text{maximize} \ \frac{1}{\alpha}\sum_{i\in\mathcal{B},\mathcal{S}\in B_i} x_i^*(\mathcal{S})\nu_i(\mathcal{S}) + \lambda \quad (9)$$

subject to:

$$\sum_{i\in\mathcal{B},\mathcal{S}\in B_i} x_i(\mathcal{S},l)\nu_i(\mathcal{S}) + \lambda \leq 1 \qquad \forall l\in\mathcal{I} \quad (9a)$$

$$\lambda \geq 0 \quad (9b)$$

$$\nu_i(\mathcal{S}) \text{ unconstrained} \quad \forall i\in\mathcal{B}, \mathcal{S}\in B_i \quad (9c)$$

The primal decomposition LP has an exponential number of variables. We resort to the dual. Even though the dual (9) has an exponential number of constraints, the ellipsoid method [22] can be applied to solve it in polynomial-time, with Algorithm 1 acting as a separation oracle for generating separating hyperplanes for the dual. Once an optimal dual solution is obtained, using a polynomial number of hyperplanes, the primal (8) can be converted to an optimization problem with a polynomial number of constraints corresponding to these hyperplanes. As a result, the convex decomposition can be solved within polynomial time. However $\nu_i(\mathcal{S})$ may be negative, making Algorithm 1 work improperly. Instead of using $\nu_i(\mathcal{S})$ directly, we set $\nu_i(\mathcal{S})^+ = \max(\nu_i(\mathcal{S}), 0)$ to circumvent this issue. IP (5) satisfies the nice *packing* property, *i.e.*, if $a\in\mathbb{Z}(DP), b\leq a$ then $b\in\mathbb{Z}(DP)$. Using the packing property, the following lemma ensures that using $\nu_i(\mathcal{S})^+$ does not violate the constraints in the dual (9).

Lemma 2. *Given an integer solution $\boldsymbol{x}'\in\mathbb{Z}(DP)$, we can obtain $\boldsymbol{x}(l)\in\mathbb{Z}(DP)$ so that $\sum_{i\in\mathcal{B},\mathcal{S}\in B_i} x_i'(\mathcal{S},l)\nu_i(\mathcal{S})^+ = \sum_{i\in\mathcal{B},\mathcal{S}\in B_i} x_i(\mathcal{S},l)\nu_i(\mathcal{S})$.*

Proof: Let

$$x_i(\mathcal{S},l) = \begin{cases} x_i'(\mathcal{S},l) & \text{if } \nu_i(\mathcal{S})>0 \\ 0 & \text{otherwise} \end{cases} \quad (10)$$

Since $\nu_i(\mathcal{S})^+ = \max(\nu_i(\mathcal{S}),0)$, it is clear that $\sum_{i\in\mathcal{B},\mathcal{S}\in B_i} x_i'(\mathcal{S},l)\nu_i(\mathcal{S})^+ = \sum_{i\in\mathcal{B},\mathcal{S}\in B_i} x_i(\mathcal{S},l)\nu_i(\mathcal{S})$. It follows from $\boldsymbol{x}'\geq 0$ that $\boldsymbol{x}(l)$ is no larger than $\boldsymbol{x}'$. Finally, due to the packing property, $\boldsymbol{x}(l)\in\mathbb{Z}(DP)$. □

Lemma 3. *If $\boldsymbol{\beta}^*$ is an optimal solution to the primal (8), then $\sum_{l\in\mathcal{I}}\beta_l^* = 1$.*

Proof: Since $\boldsymbol{\nu}^*=0, \lambda^*=1$ is feasible, the optimal solution to the dual (9) is at least 1. Suppose $\exists\lambda^*\geq 0, \boldsymbol{\nu}^*$ such that

$$\frac{1}{\alpha}\sum_{i\in\mathcal{B},\mathcal{S}\in B_i} x_i^*(\mathcal{S})\nu_i^*(\mathcal{S}) + \lambda^* > 1$$

Since $x_i^*(\mathcal{S})$ is the optimal fractional solution to the LPR, $x_i^*(\mathcal{S})\geq 0$. We then have $\frac{1}{\alpha}\sum_{i\in\mathcal{B},\mathcal{S}\in B_i} x_i^*(\mathcal{S})\nu_i^*(\mathcal{S})^+ \geq \frac{1}{\alpha}\sum_{i\in\mathcal{B},\mathcal{S}\in B_i} x_i^*(\mathcal{S})\nu_i^*(\mathcal{S}) > 1-\lambda^*$. Since the integrality gap is at most α, verified by Algorithm 1 when the objective is $\boldsymbol{\nu}^{*+}$, there must be $l\in\mathcal{I}$ satisfying $\sum_{i\in\mathcal{B},\mathcal{S}\in B_i} x_i'(\mathcal{S},l)\nu_i^*(\mathcal{S})^+ > 1-\lambda^*$. By Lemma 2, we have $\exists\boldsymbol{x}\in\mathbb{Z}(DP)$, such that $\sum_{i\in\mathcal{B},\mathcal{S}\in B_i} x_i(\mathcal{S},l)\nu_i^*(\mathcal{S}) > 1-\lambda^*$. This implies that $\boldsymbol{\nu}^*$ and λ^* violate constraint (9a). Therefore, the optimal value for the dual (9) is 1, and $\sum_{l\in\mathcal{I}}\beta_l^*=1$ due to strong LP duality. □

C. The Randomized Auction

$\{\beta_l\}_{l\in\mathcal{I}}$ in the convex decomposition can be viewed as a probability distribution over feasible integer solutions in $\mathbb{Z}(DP)$. Given the convex decomposition $\boldsymbol{x}^*/\alpha = \sum_{l\in\mathcal{I}}\beta_l\boldsymbol{x}(l)$, as shown in Algorithm 2, we select each valid integer solution $\boldsymbol{x}(l)$ randomly with probability β_l, and set the prices $\Pi_i = \Pi_i^F/\alpha$. The following theorem establishes expected truthfulness of the randomized auction.

Theorem 4. *The randomized auction in Algorithm 2 is truthful in expectation, and achieves an α-approximation to the optimal social welfare of the cloud market.*

Proof: The expected utility of a given bidder i is:

$$\begin{aligned} u_i(\sum_{l\in\mathcal{I}}\beta_l\boldsymbol{x}(l)) - \Pi_i &= u_i(\boldsymbol{x}^*/\alpha) - \Pi_i^F/\alpha \\ &= (u_i(\boldsymbol{x}^*) - \Pi_i^F)/\alpha \end{aligned}$$

The second equality is due to the linearity of $u_i(\boldsymbol{x})$. This means the expected utility is scaled down by α from the utility in the fractional VCG auction. Truthfulness of the randomized auction thus follows from that of the fractional VCG auction. □

VI. Performance Evaluation

We have implemented the randomized auction, including Algorithm 1 and the ellipsoid algorithm as its modules, for performance evaluation. The target cloud system includes a medium-sized cloud provisioning six types of VMs, constructed from three types of resources (CPU, RAM, and storage), following the configurations in Tab. I. Each cloud user bids for four VM bundles, which are synthesized from Google Cluster Data [10], while bidding prices are generated uniformly at random.

A. Performance of the Approximation Algorithm

We first study the performance of Algorithm 1 through varying the number of cloud users from 100 to 900, as illustrated in Fig. 2. Algorithm 1 achieves a close-to-optimal performance, much better than the theoretical approximation ratio proved in Theorem 3. We suspect that the analysis of Algorithm 1 can be further improved, for a tighter bound on the approximation ratio. Fig. 2 also shows that Algorithm 1 scales to a large number of bidding requests without sacrificing the social welfare approximation ratio.

B. Static Provisioning vs Dynamic Provisioning

We next compare static resource provisioning with dynamic resource provisioning in terms of economic efficiency. Two types of static provisioning are considered: Static Provisioning

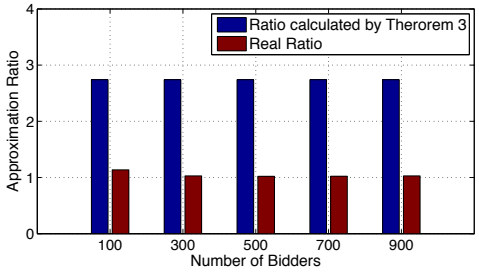


Fig. 2. A comparison between theoretical ratios and real ratios.

I where all six types of VMs are provisioned with the same number; Static Provisioning II where a large amount of resources have been packed into the entry-level VM, m1.medium, meanwhile only relatively small numbers of the high-end VMs are provisioned.

We consider the following three performance metrics: (i) social welfare, (ii) CPU utilization, the ratio of the number of allocated CPUs to the total number of CPUs, and (iii) user satisfaction, the ratio of the number of winning cloud users to the total number of cloud users. We run VCG auction for all three styles of provisioning. The results for the three metrics are illustrated in Fig. 3(a), Fig. 3(b) and Fig. 3(c), respectively.

We observe that dynamic resource provisioning can achieve higher social welfare over both static alternatives. The social welfare increases as the number of cloud users increases. Regarding CPU utilization, dynamic resource provisioning enables almost full allocation of CPU resources, while static resource provisioning under-utilizes CPU resources due to its unresponsiveness to realtime user demands. With regard of user satisfaction, the overall trend is that user satisfaction decreases as the number of cloud users grows. For a given number of cloud users, dynamic resource provisioning performances better than the two static resource provisioning styles.

C. The Randomized Auction

We implemented the randomized auction that composes of the ellipsoid method and the primal-dual approximation algorithm. Given the randomized nature of the auction, we simulate each auction scenario for 50 times and compute the average social welfare. Fig. 4(a) compares the randomized auction with the classic VCG auction in terms of economic efficiency (social welfare). The black curve in Fig. 4(a) is the expected social welfare calculated according to Theorem 4. The results of the randomized auction fluctuate around the curve, verifying the correctness of the proposed randomized algorithm. Fig. 4(b) illustrates the total payment of the randomized auction, which matches $1/\alpha$ fraction of the VCG payment.

D. An Even Better Randomized Auction?

The bound α proven in Theorem 3 can be loose, as suggested by simulation results from Sec. VI-A. This might make the randomized auction pessimistic, over-scaling the fractional VCG prices and compromising revenue of the cloud provider. We are curious to know whether smaller α can still work with the convex decomposition (8). In Fig. 4(a) and Fig. 4(b), $\alpha = 3.179, 3.184, 3.219, 3.334, 3.333, 3.333, 3.330, 3.333, 3.333$ for these 9 points respectively. We experiment with $\alpha = 2$ in the convex decomposition (8), and run the ellipsoid method for the dual (9) again. After obtaining the results, we check all candidate integer solutions with the constraints $(5a)$ and $(5b)$, to ensure that all these solutions are feasible. The results shown in Fig. 5(a) and Fig. 5(b), are rather surprising. $\alpha = 2$ works well with the randomized auction, producing a much better approximation ratio for our proposed auction. However this is not always the case if we employ a smaller α such as $\alpha = 1.5$. The approximation ratio given by Theorem 3 guarantees the existence of such integer solution which is at least $1/\alpha$ times of the fractional solution in the worst case.

VII. Conclusion

Focusing on dynamic resource provisioning and heterogeneous types of VMs, we first propose a cooperative primal dual approximation algorithm with approximation ratio close to 2.72. Employing the cooperative approximation algorithm as a building block, we then design a novel randomized auction using a pair of tailed primal and dual LPs to decompose an optimal fractional solution into a summation of a series of weighted valid integer solutions. The randomized auction achieves the same approximation ratio in social welfare as the cooperative algorithm does. Simulation studies verify the efficacy of the proposed auction and the effectiveness of dynamic resource provisioning over static resource provisioning.

Appendix A
Proof of Lemma 1

Proof: Since the set $\{\mathcal{S}_i\}_{i\in\mathcal{B}}$ is selected by line 11, where each $\mathcal{S}_i$ belongs to the corresponding B_i, *i.e.*, the corresponding cloud user, we have $\forall i \in \mathcal{B}, \mathcal{S} \in B_i$:

$$b_i(\mathcal{S}) \leq b_i(\mathcal{S}_i) \tag{11}$$

Because y_μ is set to $b_\mu(\mathcal{S}_\mu)$ where $b_\mu(\mathcal{S}_\mu) \geq b_\mu(\mathcal{S}), \forall \mu \in \mathcal{U}, \mathcal{S} \in B_\mu$. That is:

$$y_\mu \geq b_\mu(\mathcal{S}), \forall \mu \in \mathcal{U}, \mathcal{S} \in B_\mu$$

which implies that constraint (6a) is satisfied $\forall \mu \in \mathcal{U}, \mathcal{S} \in B_\mu$.

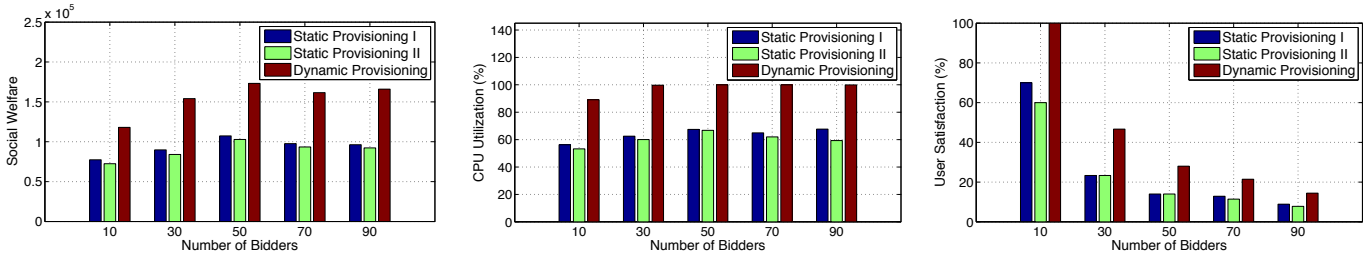
Next we examine the users $\mu \in \mathcal{B} \setminus \mathcal{U}$. Note that S_τ is decided by line 13, which is a maximization. Therefore,

$$\begin{aligned} f(z_{\tau-1}, S_\tau) &= \frac{b_\tau(\mathcal{S}_\tau)}{\sum_{k=1}^{t}\sum_{j=1}^{m} n_j^{\mathcal{S}_\tau} r_j^k z_k^{\tau-1}} \\ &\geq \frac{b_i(\mathcal{S}_i)}{\sum_{k=1}^{t}\sum_{j=1}^{m} n_j^{\mathcal{S}_i} r_j^k z_k^{\tau-1}}, \quad \forall i \in \mathcal{B}\setminus\mathcal{U} \Leftrightarrow \\ f(z_{\tau-1}, S_\tau) &\sum_{k=1}^{t}\sum_{j=1}^{m} n_k^{\mathcal{S}_i} r_j^k z_k^{\tau-1} \geq b_i(\mathcal{S}_i), \quad \forall i \in \mathcal{B}\setminus\mathcal{U} \end{aligned} \tag{12}$$

Since $\epsilon \sum_{j=1}^{m} n_j^{\mathcal{S}_1} r_j^k \geq \sum_{j=1}^{m} n_j^{\mathcal{S}_2} r_j^k, \forall \mathcal{S}_1, \mathcal{S}_2 \in B_i, i \in \mathcal{B}, k \in [1,t]$, (12) further implies that $\forall i \in \mathcal{B}\setminus\mathcal{U}, \mathcal{S} \in B_i$,

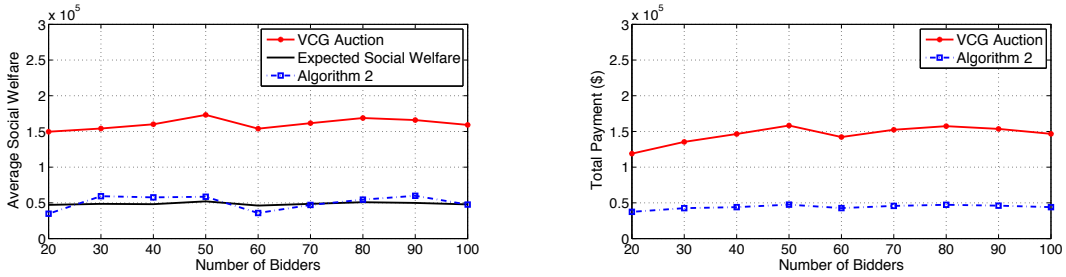
$$\epsilon f(z_{\tau-1}, S_\tau) \sum_{k=1}^{t}\sum_{j=1}^{m} n_k^{\mathcal{S}} r_j^k z_k^{\tau-1} \geq \epsilon b_i(\mathcal{S}_i) \geq b_i(\mathcal{S})$$

Thus $(y^{\tau-1}, \epsilon f(z^{\tau-1}, \mathcal{S}_\tau) z^{\tau-1})$ is a feasible solution to the dual (6). $\square$



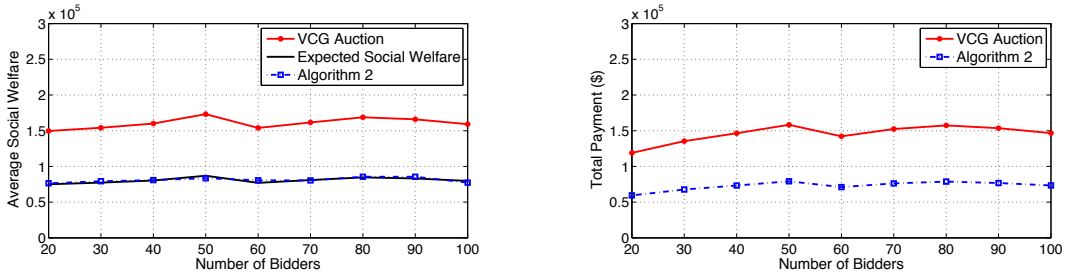
(a) Social Welfare (b) CPU Utilization (c) User Satisfaction

Fig. 3. Comparisons of social welfare, CPU utilization and user satisfaction among different provisioning styles.



(a) Social Welfare (b) Total Payment

Fig. 4. Social welfare and total payments of the randomized auction, compared with the VCG auction.



(a) Social Welfare (b) Total Payment

Fig. 5. Social welfare and total payments of the randomized auction when scaled down by $\alpha = 2$, compared with the VCG auction.

APPENDIX B
PROOF OF THEOREM 3

Proof: We first examine the complexity of Algorithm 1. Due to the stopping conditions, the `while` loop will iterate at most $|\mathcal{B}|$ times, linear to the input size. Within loop body, lines 10-12 can be finished within $O(mt|\mathcal{B}| \cdot |\mathcal{S}|)$ even using a simple brute-force search. Similarly, line 13 and lines 16-18 can also be done in polynomial time. Therefore, Algorithm 1 runs in polynomial time overall.

Next we analyze the approximation ratio of Algorithm 1. Let $d_1(\tau) = \sum_{i\in\mathcal{B}} y_i^\tau$, $d_2(\tau) = \sum_{k=1}^t c_k z_k^\tau$. Let d be the optimal solution to the dual (6). Let $\mathcal{S}_\tau$ denote the set selected in the τ-th iteration. ω is denoted the last iteration of the loop.

Case 1: Algorithm 1 stops at ω-th iteration where $\mathcal{U} = \mathcal{B}$ and $\sum_{k=1}^t c_k z_k < t\exp(\Lambda - 1)$. We know that each cloud user wins one bid. We here prove that the algorithm produces an optimal solution to IP (5). Theorem 2 guarantees that p_ω is the value of a feasible solution to IP (5). Meanwhile since $y_\mu^\omega = \max_{\mathcal{S}\in B_\mu}\{b_\mu(\mathcal{S})\} \geq b_\mu(\mathcal{S}), \forall \mu \in \mathcal{U}, \mathcal{S} \in B_\mu$, thus constraint $(6a)$ is satisfied regardless of $\boldsymbol{z}, \forall i \in \mathcal{B}, \mathcal{S} \in B_i$, *i.e.*, $(\boldsymbol{y}^\omega, \boldsymbol{z} = 0)$ is a feasible solution, whose value is exactly p_ω as well, to the dual of the LPR. By weak duality for the LP relaxation, any feasible solution to the dual (6) is an upper bound of IP (5). Therefore p_ω is the optimal value to IP (5). In this case, the approximation ratio is 1.

Case 2: Algorithm 1 stops at ω-th iteration where $d_2(\omega) = \sum_{k=1}^t c_k z_k^\omega \geq t\exp(\Lambda - 1)$. We analyze the approximation ratio in following two sub-cases.

Sub Case 2.1: $\exists$ an iteration $\tau \leq \omega$, such that $\alpha \geq \frac{d}{d_1(\tau-1)}$. That means we already found an α-approximate ratio, since (a) $d_1(\tau - 1) = p_{\tau-1}$, which is the value of the primal solution; (b) $d_1(\tau)$ is a non-decreasing function of τ because it becomes larger when the iteration continues.

Sub Case 2.2: $\alpha < \frac{d}{d_1(\tau-1)}$, for all iterations $\tau \leq \omega$. For any iteration $\tau \geq 1$, we have:

$$
\begin{aligned}
d_2(\tau) &= \sum_{k=1}^{t} c_k z_k^{\tau} \\
&= \sum_{k=1}^{t} (c_k z_k^{\tau-1} (t\exp(\Lambda-1))^{(\sum_{j=1}^{m} n_j^{\mathcal{S}_\tau} r_j^k)/(c_k - R_k)}) \\
&= \sum_{k=1}^{t} (c_k z_k^{\tau-1} (1 + \frac{\delta}{\frac{c_k}{R_k} - 1})^{(\sum_{j=1}^{m} n_j^{\mathcal{S}_\tau} r_j^k)/R_k}) \\
&\leq \sum_{k=1}^{t} (c_k z_k^{\tau-1} (1 + \frac{\delta}{\frac{c_k}{R_k} - 1} (\sum_{j=1}^{m} n_j^{\mathcal{S}_\tau} r_j^k)/R_k)) \\
&= \sum_{k=1}^{t} c_k z_k^{\tau-1} + \sum_{k=1}^{t} (\frac{\delta c_k}{c_k - R_k} \sum_{j=1}^{m} n_j^{\mathcal{S}_\tau} r_j^k) z_k^{\tau-1}) \\
&\leq d_2(\tau-1) + \Delta \sum_{k=1}^{t} \sum_{j=1}^{m} (n_j^{\mathcal{S}_\tau} r_j^k z_k^{\tau-1})
\end{aligned}
$$

where $\delta = (\frac{c_k}{R_k} - 1)((t\exp(\Lambda-1))^{1/(\frac{c_k}{R_k}-1)} - 1)$, $\Delta = \max_{1\leq k\leq t} \frac{\delta c_k}{c_k - R_k}$. The first inequality is due to $(1+a)^x \leq 1 + ax, \forall x \in [0,1]$.

Note that $\frac{\delta c_k}{c_k - R_k}$ is a non-increasing function of $\frac{c_k}{R_k} > 1$, and $\Lambda = \min_{1\leq k\leq t} c_k/R_k$, then $\frac{\delta c_k}{c_k - R_k}$ reaches the maximum when $\frac{c_k}{R_k} = \Lambda$, *i.e.*,

$$
\begin{aligned}
\Delta &= \frac{\Lambda}{\Lambda - 1}(\Lambda - 1)((t\exp(\Lambda-1))^{1/(\Lambda-1)} - 1) \\
&= \Lambda(et^{1/(\Lambda-1)} - 1)
\end{aligned}
$$

Recall the definition of $f(z^{\tau-1}, \mathcal{S}_\tau)$. We have:

$$
\sum_{k=1}^{t} \sum_{j=1}^{m} (n_j^{\mathcal{S}_\tau} r_j^k z_k^{\tau-1}) = b_\tau(\mathcal{S}_\tau)/f(z^{\tau-1}, \mathcal{S}_\tau)
$$

Since p_τ is the value of the primal solution at the end of τ-th iteration, then $p_\tau - p_{\tau-1} = b_\tau(\mathcal{S}_\tau)$, this leads to:

$$
d_2(\tau) \leq d_2(\tau-1) + \Delta \frac{p_\tau - p_{\tau-1}}{f(z^{\tau-1}, \mathcal{S}_\tau)} \tag{13}
$$

Following Lemma 1, we covert the dual variables $(y^{\tau-1}, z^{\tau-1})$ at the τ-th iteration to $(y^{\tau-1}, \epsilon f(z^{\tau-1}, \mathcal{S}_\tau) z^{\tau-1})$, which is a feasible solution to the dual (6). Therefore we have the following inequality to associate d with d_1 and d_2:

$$
\begin{aligned}
& d \leq d_1(\tau-1) + \epsilon f(z^{\tau-1}, \mathcal{S}_\tau) d_2(\tau-1) \\
\Rightarrow \quad & f(z^{\tau-1}, \mathcal{S}_\tau) \geq \tfrac{1}{\epsilon} \tfrac{d - d_1(\tau-1)}{d_2(\tau-1)}
\end{aligned}
$$

Recall that for all iterations $\tau \leq \omega$, $\alpha < \frac{d}{d_1(\tau-1)}$, implying:

$$
\frac{1}{f(z^{\tau-1}, \mathcal{S}_\tau)} \leq \epsilon \frac{d_2(\tau-1)}{d - d_1(\tau-1)} \leq \epsilon \frac{\alpha}{\alpha - 1} \frac{d_2(\tau-1)}{d}
$$

Substitute this bound on $1/f(z^{\tau-1}, \mathcal{S}_\tau)$ in Eqn. (13):

$$
\begin{aligned}
d_2(\omega) &\leq d_2(\omega-1) + \epsilon \frac{\alpha\Delta}{(\alpha-1)d} (p_\omega - p_{\omega-1}) d_2(\omega-1) \\
&= d_2(\omega-1)(1 + \epsilon \frac{\alpha\Delta}{(\alpha-1)d}(p_\omega - p_{\omega-1})) \\
&\leq d_2(\omega-1)\exp(\epsilon \frac{\alpha\Delta}{(\alpha-1)d}(p_\omega - p_{\omega-1})) \\
&\leq d_2(0)\exp(\epsilon \frac{\alpha\Delta}{(\alpha-1)d} p_\omega)
\end{aligned}
$$

the second inequality is due to $1 + x \leq e^x, \forall x \geq 0$.

Note that the stopping condition in this sub case is $d_2(\omega) \geq t\exp(\Lambda-1)$ and $d_2(0) = t$, as a result, we have that:

$$
\begin{aligned}
& t\exp(\Lambda-1) \leq t\exp(\epsilon \frac{\alpha\Delta}{(\alpha-1)d} p_\omega) \\
\Leftrightarrow \quad & \Lambda - 1 \leq \epsilon \frac{\alpha\Delta}{(\alpha-1)} \frac{p_\omega}{d} \Leftrightarrow \quad d/p_\omega \leq \epsilon \frac{\alpha\Delta}{(\alpha-1)(\Lambda-1)}
\end{aligned}
$$

Due to the weak duality theorem in linear programming and the relaxation of IP (5), the following inequality holds:

$$
DP(\mathcal{B})^*/p_\omega \leq d/p_\omega
$$

where $DP(\mathcal{B})^*$ is the value of the optimal solution to IP (5). This means d/p_ω plays as an upper bound of the approximation ratio.

Finally we obtain the approximation ratio:

$$
\epsilon \frac{\alpha\Delta}{(\alpha-1)(\Lambda-1)} = 1 + \epsilon \frac{\Lambda}{\Lambda - 1}(et^{1/(\Lambda-1)} - 1) = \alpha.
$$

□

REFERENCES

[1] *Amazon Elastic Compute Cloud*, http://aws.amazon.com/ec2/.

[2] *Linode*, https://www.linode.com/speedtest/.

[3] RightScale, "Social Gaming in the Cloud: A Technical White Paper," *White Paper*, 2010.

[4] M. Hajjat, X. Sun, Y.-W. E. Sung, D. A. Maltz, S. Rao, K. Sripanidkulchai, and M. Tawarmalani, "Cloudward Bound: Planning for Beneficial Migration of Enterprise Applications to the Cloud," in *Proc. of ACM SIGCOMM*, 2010.

[5] S. Zaman and D. Grosu, "Combinatorial Auction-Based Dynamic VM Provisioning and Allocation in Clouds," in *Proc. of IEEE CloudCom*, 2011.

[6] ——, "Combinatorial Auction-based Allocation of Virtual Machine Instances in Clouds," *Journal of Parallel and Distributed Computing*, vol. 73, no. 4, pp. 495 – 508, 2013.

[7] A. Gopinathan, Z. Li, and C. Wu, " Strategyproof Auctions for Balancing Social Welfare and Fairness in Secondary Spectrum Markets," in *Proc. IEEE INFOCOM*, 2011.

[8] R. Lavi and C. Swamy, "Truthful and Near-Optimal Mechanism Design via Linear Programming," in *Proc. of IEEE FOCS*, 2005.

[9] Y. Zhu, B. Li, and Z. Li, "Truthful Spectrum Auction Design for Secondary Networks," in *Proc. of IEEE INFOCOM*, 2012.

[10] *Google Cluster Data*, http://googleresearch.blogspot.ca/2011/11/more-google-cluster-data.html.

[11] W. Vickrey, "Counterspeculation, Auctions, and Competitive Sealed Tenders," *The Journal of finance*, vol. 16, no. 1, pp. 8–37, 1961.

[12] E. Clarke, "Multipart Pricing of Public Goods," *Public Choice*, vol. 11, no. 1, pp. 17–33, 1971.

[13] T. Groves, "Incentives in Teams," *Econometrica: Journal of the Econometric Society*, pp. 617–631, 1973.

[14] R. Carr and S. Vempala, "Randomized Metarounding," *Random Struct. Algorithms*, vol. 20, no. 3, pp. 343–352, May 2002.

[15] Z. Li, B. Li, and Y. Zhu, "Designing Truthful Spectrum Auctions for Multi-hop Secondary Networks," *IEEE Transactions on Mobile Computing*, vol. 12, 2013.

[16] H. Zhang, B. Li, H. Jiang, F. Liu, A. Vasilakos, and J. Liu, "A Framework for Truthful Online Auctions in Cloud Computing with Heterogeneous User Demands," in *Proc. of IEEE INFOCOM*, 2013.

[17] R. Buyya, R. Ranjan, and R. N. Calheiros, "InterCloud: Utility-oriented Federation of Cloud Computing Environments for Scaling of Application Services," in *Proc. of ICA3PP*, 2010.

[18] H. Li, C. Wu, Z. Li, and F. C. Lau, "Profit-Maximizing Virtual Machine Trading in a Federation of Selfish Clouds," in *Proc. of IEEE INFOCOM mini conference*, 2013.

[19] H. Kellerer, U. Pferschy, and D. Pisinger, *Knapsack Problems*. Springer, 2004.

[20] P. Briest, P. Krysta, and B. Vöcking, "Approximation Techniques for Utilitarian Mechanism Design," in *Proc. of ACM STOC*, 2005.

[21] V. Vazirani, *Approximation Algorithms*. Springer, 2001.

[22] M. Grötschel, L. Lovász, and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, 1988.