Towards Optimal Capacity Segmentation with Hybrid Cloud Pricing

Wei Wang, Baochun Li, and Ben Liang
Department of Electrical and Computer Engineering
University of Toronto
Toronto, ON M5S 3G4, Canada
weiwang@eecg.toronto.edu, bli@eecg.toronto.edu, liang@comm.utoronto.ca

***Abstract*—Cloud resources are usually priced in multiple markets with different service guarantees. For example, Amazon EC2 prices virtual instances under three pricing schemes — the subscription option (a.k.a., Reserved Instances), the pay-as-you-go offer (a.k.a., On-Demand Instances), and an auction-like spot market (a.k.a., Spot Instances) — simultaneously. There arises a new problem of *capacity segmentation*: how can a provider allocate resources to different categories of pricing schemes, so that the total revenue is maximized? In this paper, we consider an EC2-like pricing scheme with traditional pay-as-you-go pricing augmented by an auction market, where bidders periodically bid for resources and can use the instances for as long as they wish, until the clearing price exceeds their bids. We show that optimal periodic auctions must follow the design of *m*+1-price auction with seller's reservation price. Theoretical analysis also suggests the connections between periodic auctions and EC2 spot market. Furthermore, we formulate the optimal capacity segmentation strategy as a Markov decision process over some demand prediction window. To mitigate the high computational complexity of the conventional dynamic programming solution, we develop a near-optimal solution that has significantly lower complexity and is shown to asymptotically approach the optimal revenue.**

I. Introduction

Cloud computing transforms a large part of the IT industry by fulfilling the long-held ambitious vision of computing as a utility. Users pay to access computing resources delivered over the Internet, just as they pay to use water and electricity. Like other utility providers, many cloud providers charge their customers in a regular *pay-as-you-go* manner [1], [2], [3], [4], [5], [6], [7]. A provider sets a static or infrequently updated per-unit price, and users pay for only what they use.

Along with the pay-as-you-go offer, there are two additional pricing schemes widely adopted in cloud markets: the *subscription option* [2] and the *spot market* [1]. In the former scheme, a user pays a one-time subscription fee to reserve one unit of resource for a certain period of time. A user can use the reserved resource whenever it wants during the subscription period, under a significantly discounted usage price. The spot market, on the other hand, is an auction-like mechanism. Users periodically submit bids to the provider, who in turn posts a series of *spot prices*. Users gain resource access and can use the resources for as long as they wish, until the spot price rises above their bids, at which time they are rejected.

Instead of exclusively selling computing services via a single pricing channel, many providers use multiple pricing schemes simultaneously to charge for cloud services. For example, both pay-as-you-go pricing and subscription option are offered in [2], [4], [7]. Amazon EC2, on the other hand, leases virtual instances through all three pricing channels [1].

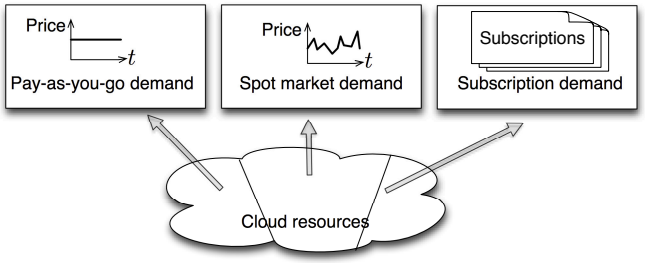


Fig. 1. The capacity segmentation problem: how do we allocate resources to each pricing model so that the revenue is maximized?

Compared with leasing cloud resources via a single pricing channel, multiple categories of pricing strategies are more attractive to a provider for two reasons. *First*, with a combined pricing structure, the deficiency of one business mode is compensated by another. For example, the demand uncertainty of pay-as-you-go pricing is compensated by risk-free income from subscription users bearing long-term usage commitment. *Second*, the use of multiple pricing categories expands the market demand by offering more flexible choices to accommodate different types of users. For example, price-sensitive users who cannot afford the pay-as-you-go price now have a chance to gain access to resources by bidding in the spot market.

The co-existence of multiple pricing channels, as illustrated in Fig. 1, raises a new, and challenging, question to a provider: with limited resources available, how do we allocate them to each pricing channel so that the overall revenue is maximized?

To answer this question, in this paper, we consider the problem of capacity segmentation with two pricing models applied in parallel, i.e., the regular pay-as-you-go offer and periodic auctions. The latter allows users to periodically bid for resources in a sequence of auctions. In each period, a uniform *take-it-or-leave-it* price is posted to clear the market. The winners can use the resources for as long as they wish, until the clearing price rises above their bids. The provider's

problem is to optimally allocate its resources to the two pricing channels, based on supply and demand, to maximize the obtained revenue.

We choose the aforementioned two pricing models as building blocks of the capacity segmentation problem for two reasons. *First,* due to the upfront usage commitment borne by users, subscription demand generates long-term risk-free income to the provider. In this sense, subscription requests are more preferred to providers [7] and are always fulfilled at the highest priority. We therefore focus only on the remaining two pricing models. *Second,* given that Amazon reveals no detailed information on how the spot price is determined [1], it is unclear how the spot market is operated. We hence turn to periodic auctions as they share similar pricing forms as the EC2 spot market[1](i.e., both are bid-based).

To the best of our knowledge, we are the first to consider such a capacity segmentation problem in cloud markets with hybrid pricing. We make two original contributions in this paper. *First,* we show that an optimal design for the auction channel must follow the form of the m+1-price auction with seller's reservation price. Contrary to the well-known result that, in general, bidders tend to underbid in a uniform-price auction (including the m+1-price auction), we show that, in cloud environments, however, the m+1-price auction is essentially *truthful* with *two-dimensional* bids. Interestingly, such truthfulness is also expected in the EC2 spot market. In this case, replacing Amazon's design with periodic auctions does not change user behaviours, resulting in the same market response.

Second, we formulate the optimal capacity segmentation problem as a Markov decision process (MDP). However, optimally solving this MDP with conventional dynamic programming is computationally prohibitive, especially for a large provider with enormous capacity. By utilizing some special bounding structure of the optimization problem, we further develop an approximate solution that significantly reduces the computational complexity from $O(C^3)$ to $O(C^2)$, with C being the capacity of the provider. We show that the proposed approximation is asymptotically optimal as the user demand becomes sufficiently high, which is naturally the case for a large cloud provider. This analytical result, together with extensive simulations, suggests that the approximation closely approaches the optimal solution.

The remainder of this paper is organized as follows. In Sec. II, we briefly survey the related work. Our model and notations are introduced in Sec. III. In Sec. IV, we characterize the revenue obtained in the auction market, and present an optimal design with maximum revenue. We also discuss its connections to the EC2 spot market. In Sec. V, we first present rationales for the use of multiple pricing models. We then show that achieving optimal segmentation is computationally prohibitive. For this reason, we present an asymptotically optimal solution where the computational complexity is significantly reduced by identifying some optimization structures in the problem. Extensive simulation studies are presented in Sec. VI. Sec. VII concludes the paper.

[1]We emphasize that periodic auctions are not equivalent to the EC2 spot market. While spot users are *price-takers* unaware of how spot prices are produced, auction bidders, on the other hand, have full knowledge on pricing details and can affect the clearing price via strategic bidding.

II. Related Work

Three pricing models are now widely adopted in cloud markets, i.e., the regular pay-as-you-go offer, the subscription option, and the newly invented spot market. The providers advertise that having multiple pricing schemes benefits cloud users by lowering their costs [1], [7]. Existing works also present some user strategies to switch between different pricing markets to cut the cost [8], [9].

However, little has been addressed from the cloud provider's perspective, on how their resources should be allocated to different markets to maximize revenue. Relevant research works in the literature include [10], which investigates the resource allocation problem in either static pricing or variable pricing, by solving a static optimization program, and [11], which presents a dynamic auction-based model for resource allocations in the grid system. None of these works considers the coexistence of multiple different pricing markets.

We believe the key to solving the capacity segmentation problem lies in understanding the EC2 spot market. However, since Amazon reveals no detailed pricing information, it remains unclear how the spot price is determined. Despite Amazon's claim that the price is calculated based on market demand and supply [1], some recent works conjecture that the price is in fact artificially set via some random process [9], [12]. In this paper, we consider periodic uniform-price auctions, as they share similar pricing forms as the EC2 spot market. Unlike general uniform-price auctions discussed in economics literature [13], [14], in the cloud environment, *partial fulfillment* is not accepted: a bidder is either rejected or having all requested instances [1] being fulfilled. For this reason, our design avoids the well-known effect of "demand reduction" observed in general uniform-price auctions — that bidders have an incentive to bid lower than their true values [13], [14] — and is proved to be truthful with two-dimensional bids.

We note that there exist some works in the literature of economics that discuss a similar resource allocation problem in the retail market, where two pricing channels are used to sell products, the auction market and the regular pay-as-you-go pricing [15], [16]. However, neither the model nor the analysis applies to the cloud environment. *First,* their models are based on sales markets, where resources are sold to customers and will never be reclaimed and made available to others. In contrast, cloud instances are leased to users and can be reused by others once the resources are released by previous owners. *Second,* their analysis relies on a strong assumption that each customer is restricted to ask for only one unit of product, which is clearly not the case for cloud users. We note that [15] further assumes that the seller capacity is infinite.

It is worth mentioning that optimal periodic auctions have also been studied in the retail market in [17], [18], but the same

problems mentioned above render those works inapplicable in cloud markets.

III. System Model

Suppose a cloud provider has allocated a fixed capacity C for a certain type of virtual instances, i.e., at any given time, up to C instances of that type can be hosted. All these instances are leased in two pricing channels, an auction market and a pay-as-you-go market, simultaneously. We take discrete time horizons indexed by $t = 1, 2, \ldots$ in the following analysis.

A. User Model

Pay-as-you-go users. The pay-as-you-go market offers guaranteed services. Users can run their instances for as long as they wish, and are charged what they used based on a constant regular price p_r. In particular, denote by $t_{i,j}$ the running time of instance j hosted for user i. User i is then charged $p_r t_{i,j}$ for using that instance. To make the analysis tractable, we take a technical assumption that $t_{i,j}$'s are i.i.d. exponential. In discrete settings, this implies that $t_{i,j}$ follows the geometric distribution with p.m.f. $P(t_{i,j} = k) = q(1-q)^{k-1}$, where q is the probability that a currently running instance will be terminated by its user in the next period. Therefore, the expected overall payment for using one instance is $\mathbf{E}[p_r t_{i,j}] = p_r \mathbf{E}[t_{i,j}] = p_r/q$. Although the i.i.d. exponential instance running time is a simple model and practical user behaviours may not follow this pattern, taking this technical assumption allows tractable analysis and has been shown to give interesting insights into practical systems. We also note that such exponential resource usage time is widely adopted in economics literatures to analyze rental markets [19], [20].

Because p_r is constant, pay-as-you-go users have no purchasing strategy as the auction bidders have. We assume there are R_r^t instance requests received at time t, and if the available capacity allocated to the pay-as-you-go market is below R_r^t, some users do not receive their requested instances. The exact mechanism for user admission (e.g., first-come, first-served) is unimportant to the problem under consideration since the same price p_r is charged for each instance.

Users in the auction market. Instances purchased in the auction market offer no service guarantees and will be terminated by the provider whenever the bid has been exceeded by the clearing price. Suppose at time t, there are N_a^t bidders joining the auction. Each bidder i ($1 \le i \le N_a^t$) wishes to access n_i instances and has a maximum affordable price v_i, also known as the *reservation price*, for using one instance at one period. User i then submits a *two-dimensional bid* (r_i^t, b_i^t) requesting r_i^t instances with a bid price b_i^t. Note that user i could strategically misreport its bid (i.e., $b_i^t \neq v_i$ or $r_i^t \neq n_i$) as long as it is beneficial for her to do so.

After all bids are collected, the cloud provider runs the auction and charges a take-it-or-leave-it clearing price p_a^t to all winners: each user i with $b_i^t > p_a^t$ (resp. $b_i^t < p_a^t$) either has its new requests fulfilled (resp. rejected) or has its running instances continued (resp. terminated). Those with $b_i^t = p_a^t$ may or may not be accepted depending on the specific auction mechanism. The value of p_a^t is calculated based on some specified mechanism that is publicly known to all bidders. We therefore define user i's utility at time t as follows:

$$u_i^t(r_i^t, b_i^t) = \begin{cases} n_i v_i - r_i^t p_a^t\,, & \text{if } b_i^t > p_a^t \text{ and } r_i^t \ge n_i; \\ 0\,, & \text{otherwise.} \end{cases} \quad (1)$$

Here, both n_i and v_i are private information known only to user i, and are distributed with joint p.d.f. $f_{n,v}$ and c.d.f. $F_{n,v}$ on the support $[1, \bar{n}] \times [0, \bar{v}]$. The user i's problem is to find the optimal bid such that the utility is maximized, i.e., $\max_{r_i^t, b_i^t} u_i^t(r_i^t, b_i^t)$.

It is worth mentioning that the auction described above is substantially different from the uniform-price auction considered in the literature of economics [13], [14], as bidders in the later mechanism accept partial fulfillment and have different utility functions other than (1).

B. The Problem of Optimal Capacity Segmentation

The cloud provider aims to optimally allocate its available capacity to both the pay-as-you-go and auction markets, to maximize its obtained revenue. Let the available capacity at time t be C^t. In addition to knowing the exact number of requests in the current time slot t, we assume that the provider may predict the demand in the near future: it knows the distributions of N_a^τ (the user number in the auction market) and R_r^τ (the total requests in the pay-as-you-go market) for $\tau \le T = t + w$, with w being some *prediction window*. Note that forecasting future demand has already been addressed in some literature [10], [21].

Given C^t at time t, denote by $\Gamma^t(C^t)$ the maximum expected aggregate revenue obtained from t to T. Let $\gamma_a(c)$ and $\gamma_r(c)$ be the revenues of allocating c instances to the auction and the regular pay-as-you-go markets, respectively. The problem of optimal capacity segmentation is to find the optimal capacity allocations to the two markets such that the revenue collected within the prediction window is maximized. This can be expressed in the following recursive form:

$$\Gamma^t(C^t) = \mathbf{E}\Bigg[\max_{0 \le C_a^t \le C^t} \big\{\gamma_a(C_a^t) + \gamma_r(C^t - C_a^t) + \mathbf{E}_{C^{t+1}}\big[\Gamma^{t+1}(C^{t+1})\big]\big\}\Bigg], \quad (2)$$

where C_a^t is the capacity allocated to the auction market, and the boundary conditions are $\Gamma^{T+1}(c) = 0$ for all $c = 0, 1, \ldots, C$.

Since the pay-as-you-go price is infrequently updated [1], in this work we consider only the shorter time-scale problem of capacity segmentation given a fixed p_r. Then we have

$$\gamma_r(c) = \begin{cases} p_r c/q\,, & \text{if } c \le R_r^t; \\ p_r R_r^t/q\,, & \text{otherwise.} \end{cases} \quad (3)$$

Note that a discussion on how to optimize p_r can be conducted based on the proposed revenue maximizing method, but it additionally requires knowledge of the yet unknown supply-demand relation and hence is left open for future research.

To determine the value of C^{t+1} in (2), we note that at time t, there are C_a^t instances allocated to the auction market and $C - C_a^t$ instances held for the pay-as-you-go users. Suppose that right before $t+1$, X of them are terminated by pay-as-you-go users and are returned to the system. As a result, there are $C^{t+1} = C_a^t + X$ instances being available for new requests at the beginning of $t+1$. From the assumption of the exponential running time as explained in Sec. III-A, it is easy to see that X follows a binomial distribution with $P(X = k) = B(C - C_a^t, k, q)$, where $B(n,k,q) = \binom{n}{k} q^k (1-q)^{n-k}$. We re-write (2) as

$$\Gamma^t(C^t) = \mathbf{E}\Bigg[\max_{0 \le C_a^t \le C^t} \big\{\gamma_a(C_a^t) + \gamma_r(C^t - C_a^t) + \mathbf{E}_X\big[\Gamma^{t+1}(C_a^t + X)\big]\big\}\Bigg]. \quad (4)$$

Note that the capacity segmentation problem is essentially formulated as a Markov decision process. The cloud provider's problem is to solve (4) to find the optimal capacity segmentation point C_a^{t*}.

It is worth mentioning that the capacity segmentation problem stated in (4) is non-trivial: Neither market is always more profitable than the other. In the auction market, there may be high-bid requests from users starving for cloud resources, which can drive the auction price above the regular pay-as-you-go price (i.e., $p_a^t > p_r$), making the auction market more profitable for a provider. Such phenomenon has indeed been observed in the real world: In EC2 pricing, the spot price occasionally exceeds the regular price [1]. A cloud provider has to dynamically segment its capacity to maximize its revenue.

For now, problem (4) is still not well defined. One question remains: how to design the optimal auction market to maximize the revenue $\gamma_a(C_a^t)$ given C_a^t instances are allocated? We answer this question in the following section.

IV. Optimal Auction Design

This section addresses the question raised above. Given an allocated capacity C_a^t, what is the optimal design for the auction market described in Sec. III? We investigate the structure of the optimal auction and characterize its revenue $\gamma_a(C_a^t)$. We also discuss its connections to Amazon EC2 Spot Instances.

A. Preliminaries

An auction mechanism $\mathcal{M}$ is said to be *truthful* if for every bidder, no matter how others behave, the optimal bidding strategy is always to submit its true bids. In our problem, this means that for every i, $u_i^t(n_i, v_i) \ge u_i^t(r_i^t, b_i^t)$ for any (r_i^t, b_i^t).

By the *Revelation Principle* [22], it suffices to focus only on truthful auction designs when revenue is of interest. Lemma 1 characterizes the revenue of any truthful auctions by extending the *Revenue Equivalence Theorem* [23] to the two-dimensional domain. The proof is similar to [23] and is given in our technical report [24].

Lemma 1: Let $\mathbf{v} = (v_i)$ and $\mathbf{n} = (n_i)$. Denote by $\gamma_{\mathcal{M}}$ the revenue of a mechanism $\mathcal{M}$ with two-dimensional bids. Then for any *truthful* $\mathcal{M}$, we have

$$\mathbf{E}_{\mathbf{n},\mathbf{v}}[\gamma_{\mathcal{M}}] = \mathbf{E}_{\mathbf{n},\mathbf{v}}\left[\sum_{i=1}^{N} n_i \phi(v_i) x_i(\mathbf{n},\mathbf{v})\right]. \quad (5)$$

Here, $\phi(v_i) = v_i - \frac{1 - F_v(v_i|n_i)}{f_v(v_i|n_i)}$, and $x_i(\mathbf{n},\mathbf{v})$ takes the value 0 or 1 depending on whether user i loses or wins, respectively.

Lemma 1 greatly simplifies the revenue analysis. It essentially indicates that the expected revenue of a truthful mechanism only depends on who is to win (i.e., the values of x_i's), not what they pay (i.e., the value of p_a^t). Besides, by (5), one could explain the value of $\phi(v_i)$ as the marginal revenue generated by auctioning one instance to user i. This is also user i's expected payment of using one instance. Therefore, the expected auction revenue is calculated as the sum of payments collected from all winning users.

We now characterize the revenue of the auction market described in Sec. III. Without loss of generality, suppose bidders are sorted in a decreasing order of their bidding prices, i.e., $v_1 \ge v_2 \ge \cdots \ge v_{N_a^t}$. We have the following proposition.

Proposition 1: Suppose $\mathcal{M}$ is a truthful auction offering a uniform take-it-or-leave-it price. Let m be the number of winning bidders[2]. Then the expected revenue of $\mathcal{M}$ is characterized as follows:

$$\mathbf{E}_{\mathbf{n},\mathbf{v}}[\gamma_{\mathcal{M}}] = \mathbf{E}_{\mathbf{n},\mathbf{v}}\left[\sum_{i=1}^{m} n_i \phi(v_i)\right]. \quad (6)$$

Proof: Since the auction market offers a uniform take-it-or-leave-it price p_a^t, every winning bidder i must have $v_i \ge p_a^t$. In this case, the top m bidders win the auction, i.e., $x_i = 1$ for $i = 1, 2, \ldots, m$. Substituting this to (5) and applying Lemma 1, we see that the statement holds. ∎

Proposition 1 essentially indicates that maximizing the auction revenue is equivalent to maximizing the RHS of (6), subject to the capacity constraint:

$$\begin{aligned} \max_{m \le N_a^t} \quad & \sum_{i=1}^{m} n_i \phi(v_i) \\ \text{s.t.} \quad & \sum_{i=1}^{m} n_i \le C_a^t. \end{aligned} \quad (7)$$

For mathematical convenience, we take the standard *regularity assumption* that $\phi(\cdot)$ is increasing. This is not a restrictive assumption, as it generally holds for most distributions [23] and is widely adopted in the literature [13], [23], [25].

B. Optimal Auction Market

Problem (7) can be optimally solved by a greedy algorithm: Sequentially accept bidders' requests, from the top valued (i.e., the highest $\phi(v_i)$) to the bottom, until there is no longer capacity for more. It suffices to assume that all requests are positively valued ($\phi(v_i) > 0$), as those with $\phi(v_i) \le 0$ will

[2]The value of m depends on $\mathbf{n}$ and $\mathbf{v}$.

never be fulfilled. The optimal auction market, described in Algorithm 1, is designed based on the above process.

Algorithm 1 Optimal Auction Market with Capacity C_a^t

1. **if** $\sum_{i=1}^{N_a^t} r_i^t \leq C_a^t$ **then**
2. All bidders win ($m = N_a^t$), with $p_a^t = \phi^{-1}(0)$
3. **else**
4. Top m bidders win, with $p_a^t = b_{m+1}^t$, where $\sum_{i=1}^{m} r_i^t \leq C_a^t < \sum_{i=1}^{m+1} r_i^t$
5. **end if**

We note that Algorithm 1 adopts the similar design of the canonical *m*+1-price auction, with a difference that a seller has a reservation price $\phi^{-1}(0)$. Though *m*+1-price auction is truthful for the case where each bidder requests no more than one unit of the auctioned good [26], it is well known that in general, the truthfulness no longer holds when bidders have multi-unit demands [13], [14]. However, we show that for the specific problem considered in this paper, *m*+1-price auction is two-dimensionally truthful in both n_i and v_i. To see this, we require the following lemma.

Lemma 2 (Monotonicity): For every bidder i, fix all others' submissions. Denote by $p_a^t(b_i^t, r_i^t)$ the clearing price when i bids (b_i^t, r_i^t). Then for all b_i^t (resp. r_i^t), $p_a^t(b_i^t, r_i^t)$ is increasing w.r.t. r_i^t (resp. b_i^t).

Proof: We explain the pictorial proof through Figs. 2a and 2b. First we show that p_a^t is increasing w.r.t. r_i^t, i.e., $p_a^t(b_i^t, r_i^t) \leq p_a^t(b_i^t, r_i^t + \Delta)$ for all $\Delta > 0$. It suffices to consider the following two cases.

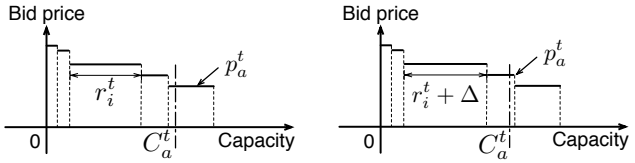
Case 1: Bidder i loses by requesting r_i^t instances. Note that increasing a bidder's request does not change its ranking (as they are sorted based on their bid prices). It is easy to verify that having this bidder requesting more instances, say, $r_i^t + \Delta$ instances, results in the same clearing price, i.e., $p_a^t(b_i^t, r_i^t) = p_a^t(b_i^t, r_i^t + \Delta)$.

Case 2: Bidder i wins by requesting r_i^t instances. Suppose it now increases its request by Δ instances. Fig. 2a illustrates the changes of clearing price p_a^t, from which we see that having a winning bidder requesting more instances essentially raises the clearing price, i.e., $p_a^t(b_i^t, r_i^t) \leq p_a^t(b_i^t, r_i^t + \Delta)$.

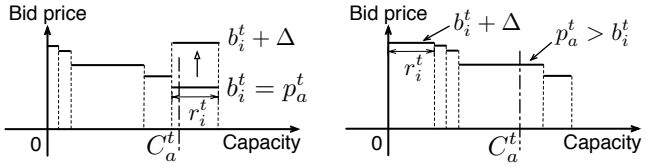
We next prove that p_a^t is also increases w.r.t. b_i^t, i.e., $p_a^t(b_i^t, r_i^t) \leq p_a^t(b_i^t + \Delta, r_i^t)$ for all $\Delta > 0$. Still, it suffices to consider two cases below.

Case 1: Bidder i loses by bidding b_i^t. Suppose it now raises its bid by Δ. We consider two cases. (1) Bidder i wins by bidding $b_i^t + \Delta$. This is shown in Fig. 2b, from which we see that the clearing price is raised. (2) Bidder i loses by bidding $b_i^t + \Delta$. In this case, if bidder i's new bid is used as the new clearing price (i.e., $p_a^t = b_i^t + \Delta$), then this new price must be higher than the original one (because its value is updated). Otherwise, the clearing price p_a^t remains unchanged. In summary, $p_a^t(b_i^t, r_i^t) \leq p_a^t(b_i^t + \Delta, r_i^t)$ for all $\Delta > 0$.

Case 2: Bidder i wins by bidding b_i^t. In this case, raising the bid price has no effect on the clearing price — the later



(a) Increasing r_i^t raises the clearing price.



(b) Increasing b_i^t raises the clearing price.

Fig. 2. Effect of demand r_i^t and bid price b_i^t on the clearing price p_a^t.

remains unchanged with the value equal to the $m+1$-th highest bid, i.e., $p_a^t(b_i^t, r_i^t) = p_a^t(b_i^t + \Delta, r_i^t)$. ■

Lemma 2 reflects the basic economic principle: With the same level of the supply, the market price rises as the bidders' demand increases. This monotonicity eliminates the incentive of *overbooking instances*, as stated in Lemma 3.

Lemma 3: For every bidder i, there is no advantage to overbook instances, i.e., given b_i^t, $u_i^t(r_i^t, b_i^t) \leq u_i^t(n_i, b_i^t)$ for all $r_i^t > n_i$.

Proof: It suffices to consider the case where bidder i wins by submitting (r_i^t, b_i^t), with $r_i^t > n_i$. In this case,

$$b_i^t \geq p_a^t(r_i^t, b_i^t) \geq p_a^t(n_i, b_i^t)\,, \tag{8}$$

where the first inequality holds since i wins by bidding (r_i^t, b_i^t), while the second inequality is derived from Lemma 2. This implies that i also wins by bidding (n_i, b_i^t). As a result,

$$\begin{aligned} u_i^t(r_i^t, b_i^t) &= n_i v_i - r_i^t p_i^t(r_i^t, b_i^t) \\ &\leq n_i v_i - n_i p_i^t(n_i, b_i^t) \\ &= u_i^t(n_i, b_i^t)\,. \end{aligned}$$

This concludes the proof. ■

Since no user has the incentive to request fewer instances than actually needed (otherwise, $u_i^t(r_i^t, b_i^t) = 0$ because $r_i^t < n_i$), Lemma 3 essentially indicates that users should always truthfully report their n_i value. This leads to the truthfulness statement as follows.

Proposition 2: Algorithm 1 is two-dimensionally truthful, i.e., $u_i^t(n_i, v_i) \geq u_i^t(r_i^t, b_i^t)$ for all (r_i^t, b_i^t), $i = 1, 2, \ldots, N_a^t$.

Proof: We consider all possible outcomes of bidding truthfully or untruthfully. Since every bidder i chooses to truthfully report n_i, any untruthful submission is of the form (n_i, b_i^t) where $b_i^t \neq v_i$. Suppose the untruthful submission leads to the bidder losing, then the propositional statement trivially holds for b_i^t. Therefore, in the following we only need to consider the case where the untruthful submission leads to the bidder winning.

Suppose bidding truthfully leads to the bidder winning, then it is easy to verify that $p_a^t(n_i, b_i^t) = p_a^t(n_i, v_i) = p$, where

p is either $\phi^{-1}(0)$ or b_{m+1}, depending on whether there is sufficient capacity to accommodate all requests. As a result, we see $u_i^t(n_i, v_i) = u_i^t(n_i, b_i^t)$.

On the other hand, if bidding truthfully leads to the bidder losing, then $p_a^t(n_i, v_i) \geq v_i$. Since by changing the submission to (n_i, b_i^t) user i wins, we must have $b_i^t > v_i$. By Lemma 2, $p_a^t(n_i, b_i^t) \geq p_a^t(n_i, v_i) \geq v_i$. Hence, $u_i^t(n_i, b_i^t) = n_i(v_i - p_a^t(n_i, b_i)) \leq 0 = u_i^t(n_i, v_i)$. This concludes the proof. ∎

Intuitively, given that all users report n_i truthfully as dictated by Lemma 3, the market can be viewed as a second price auction in terms of the bid price only, which is well-known to be truthful. We point out that the two-dimensional truthfulness of this special case of *m*+1-price auction in our problem is due to the specific characteristics of cloud markets that partial fulfillment is not allowed.

The revenue optimality of Algorithm 1 follows naturally from the proved truthfulness (i.e., Proposition 2):

Proposition 3: Among all mechanisms offering a uniform take-it-or-leave-it price, Algorithm 1 is optimal in terms of revenue maximization.

Proof: Since Algorithm 1 is truthful, all bidders bid $r_i^t = n_i$ and $b_i^t = v_i$. In this case, Algorithm 1 optimally solves problem (7). By Proposition 1, this implies that Algorithm 1 maximizes the revenue among all truthful auctions offering uniform clearing prices. Due to the Revelation Principle [22], imposing the truthfulness to the auction design does not hurt the revenue. We therefore conclude that the statement generally holds. ∎

C. Optimal Revenue

To derive the revenue obtained from Algorithm 1, one has to deal with two cases, with or without sufficient capacity to accommodate all profitable requests. To combine both cases in our subsequent discussion, we artificially insert a *virtual bidder* to the market, who requests an infinite amount of instances at a price $\phi^{-1}(0)$. Inserting a virtual bidder has no effect on the auction result, but it significantly simplifies the revenue expression. Based on Algorithm 1, $p_a^t = v_{m+1}$, and $\gamma_a(C_a^t) = v_{m+1} \sum_{i=1}^m n_i$, where $\sum_{i=1}^m n_i \leq C_a^t < \sum_{i=1}^{m+1} n_i$. By Proposition 1, we have $\mathbf{E}[\gamma_a(C_a^t)] = \mathbf{E}[v_{m+1} \sum_{i=1}^m n_i] = \mathbf{E}[\sum_{i=1}^m n_i \phi(v_i)]$. Therefore, in expectation, it is equivalent to writing

$$\gamma_a(c) = \sum_{i=1}^{m} n_i \phi(v_i)\,, \tag{9}$$

where $\sum_{i=1}^m n_i \leq c < \sum_{i=1}^{m+1} n_i$. In this sense, $n_i\phi(v_i)$ can be viewed as the *marginal revenue* generated by accepting the requests of bidder i.

D. Connections to EC2 Spot Market

It is interesting to see some connections between the auction market discussed in this paper and the *spot market* adopted by Amazon EC2 Spot Instances [1]. Similar to the auction market, spot users periodically submit bids (r_i^t, b_i^t) to Amazon, requesting r_i^t instances at a price b_i^t. A uniform *spot price* p_s^t is periodically posted by Amazon to charge the winners, i.e., those who bid higher than the spot price ($b_i^t > p_a^t$). All winners can use the instances as long as the price does not rise above their bids.

Though similar in description, the pricing of Spot Instances is by no means an auction market. Since Amazon has revealed no detailed information regarding how the spot price p_s^t is calculated, there is no way for spot users to know what p_s^t is going to be, even with the complete information of demand (i.e., other users' bids) and supply (i.e., the amount of instances offered in the spot market). This is not the case in a real auction, where the mechanism details are publicly known to every participant.

We now investigate the optimal bidding strategy for Spot Instances. Without pricing details, a valid approach for spot users is to view p_s^t as a random variable, with p.d.f. f_s^t and c.d.f. F_s^t learned from the price history published by Amazon [1]. Suppose the utility defined for user i is similar to (1) with the clearing price p_a^t replaced by the spot price p_s^t, i.e.,

$$u_i^t(r_i^t, b_i^t) = \begin{cases} n_i v_i - r_i^t p_s^t\,, & \text{if } b_i^t > p_s^t \text{ and } r_i^t \geq n_i; \\ 0\,, & \text{otherwise.} \end{cases}$$

The user's problem is to find the optimal bid so that its expected utility is maximized, i.e., $\max_{r_i^t, b_i^t} \mathbf{E}_{p_s^t} u_i^t(r_i^t, b_i^t)$.

Proposition 4: In the spot market, the optimal bid for user i is to truthfully submit (n_i, v_i).

Proof: Let A be the event that i wins by bidding b_i^t. Denote by I_X the indicator function of event X. We have

$$\begin{aligned} \mathbf{E}[u_i^t(r_i^t, b_i^t)] &= P(A)(n_i v_i - r_i^t \mathbf{E}[p_s^t | A]) I_{r_i^t \geq n_i} \\ &= F_s^t(b_i^t) \left[n_i v_i - r_i^t \int_{\underline{v}}^{b_i^t} x f_s^t(x) \mathrm{d}x \right] I_{r_i^t \geq n_i}. \end{aligned} \tag{10}$$

It is easy to see that bidding $r_i^t = n_i$ dominates all other strategies for every b_i^t. Now substituting it back to (10) and applying the first-order optimality conditions, we see that the optimal bid price is v_i. This concludes the proof. ∎

Intuitively, without knowing how the spot price reacts to different submissions, no user has the incentive to strategize over its bid. Therefore, by replacing a spot market with an auction market, the provider would expect the same user behaviour. In other words, the two markets are equivalent in terms of the market reaction. Considering that both are of similar pricing structures (i.e., both are bid-based), we believe that the auction market offers a good simulation to the spot market, and the analysis of the former sheds light on the latter.

V. Optimal Capacity Segmentation

Having characterized the revenue for the auction market, we are now ready to investigate the market segmentation problem stated in (4). Before delving into the detailed technical discussions, we justify the motivation of having two co-existing markets by taking a look at the simplest scenario where no future information is available, i.e., the prediction window w is 0.

A. Motivations for Joint Markets

When $w = 0$, (4) is reduced to a one-shot optimization problem, i.e.,

$$\Gamma^t(C^t) = \mathbf{E}\Big[\max_{0\le C_a^t\le C^t}\big\{\gamma_a(C_a^t)+\gamma_r(C^t-C_a^t)\big\}\Big]. \quad (11)$$

We are not interested in solving the problem above as an $O(C)$ solution trivially exists (i.e., search all possible C_a^t's to find the optimal segmentation). Instead, we show that this simple scenario illustrates two motivating factors behind pricing via joint markets.

First, with the auction market, low-valuation users whose $v_i < p_r$ are offered a chance for access to cloud instances. Therefore, having two markets expands the potential demand and increases the overall revenue.

Second, users with low tolerance to interruptions are offered an option to increase their *request priority*, as stated below.

Proposition 5: To maximize revenue, the provider always tries to fulfill the requests of those auction bidders whose $v_i \ge p_r/q$ before it accepts any pay-as-you-go requests.

Proof: Suppose the provider has sufficient capacity to fulfill bidder i's requests n_i. By Proposition 1, the marginal revenue of accommodating bidder i is $n_i\phi(v_i)$. Now if the provider changes its mind and allocates these n_i instances to pay-as-you-go users, then the marginal revenue would be at most $n_i p_r/q$. Note that this will not happen if $\phi(v_i) \ge p_r/q$, as bidder i's requests bring more marginal revenue to the provider. We therefore conclude the proof by noticing that $v_i \ge \phi(v_i) \ge p_r/q$. ∎

In other words, for high-valuation users, the auction market is actually offering *guaranteed services* with higher fulfillment priority. Only low-valuation users bear the risk of being interrupted.

All above justify the motivation for using multiple markets: It benefits both the provider and the users. However, though Proposition 5 reveals some basic criteria in allocating resources, it alone is unable to guarantee the optimal revenue. In fact, optimal capacity segmentation is a complicated problem. The following discussions are targeted for a general setting where short prediction is available, i.e., $w > 0$.

B. Complexity of Optimal Capacity Segmentation

Since (4) describes an MDP problem, a standard solution is *numerical dynamic programming* via backward induction. It proceeds by first simulating market demand in the last stage T based on the predicted demand and calculating the optimal segmentation made in that stage. Using this result, it then determines how to segment the capacity in stage $T-1$, based on the predicted market demand at that time. This process continues backwards until the optimal segmentation C_a^{t*} made in the current stage is obtained. In each stage τ, for each possible C^τ and each demand realization (i.e., the auction requests $(\mathbf{n},\mathbf{v})$ and pay-as-you-go requests R_r^τ), compute (4), which takes $O(C^2)$ operations. Since the computation is taken over all $C^\tau = 0,1,\ldots,C$, the complexity of one-stage calculation is $O(C^3)$. By noting that only short prediction is possible and w is usually small, we see the overall computational complexity of the above process is $O(C^3)$.

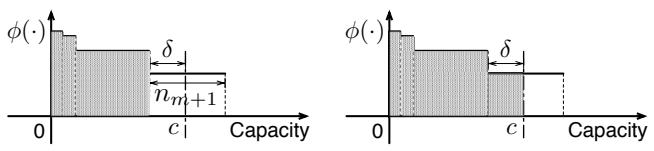


Fig. 3. The pictorial illustration of the exact auction revenue $\gamma_a(c)$ and its upper bound $\bar{\gamma}_a(c)$, shown as the shaded area in the left and right figures, respectively.

For large providers with high capacities, finding exact solutions to (4) is computationally intractable. As a typical example, when $C = 10^5$, the computation above requires $O(10^{15})$ operations, which is prohibitive when decisions need to be made in real time.

C. An Asymptotically Optimal Solution

The segmentation decision needs to be made quickly after the user demand has arrived. In practice, this is often more important than pursuing exact optimality. Hence, we next propose an asymptotically optimal solution to (4) that significantly reduces the computational complexity.

In the auction market, the bidders' requests do not always fit exactly within the allocated capacity. By (9), if c instances are allocated to the auction market, then there would be $\delta = c - \sum_{i=1}^m n_i$ instances leftover as these computing resources are insufficient to accommodate the request of bidder $m+1$, i.e., $\delta < n_{m+1}$. Fig. 3 illustrates this scenario. However, if bidder $m+1$ accepts partial fulfillment, then those δ instances would generate $\delta\phi(v_{m+1})$ additional revenue to the provider. Let $\bar{\gamma}_a(\cdot)$ be the revenue obtained as if partial fulfillment were accepted. We have

$$\begin{aligned}\bar{\gamma}_a(c) &= \gamma_a(c) + \delta\phi(v_{m+1}) \\ &= \sum_{i=1}^m n_i\phi(v_i) + \delta\phi(v_{m+1}). \end{aligned} \quad (12)$$

Clearly, $\bar{\gamma}_a$ is an *upper bound* of γ_a. Fig. 3 gives a pictorial illustration of $\gamma_a(c)$ and $\bar{\gamma}_a(c)$.

The following lemma indicates that the upper bound $\bar{\gamma}_a$ offers a close approximation to γ_a, provided that the capacity allocated to the auction market is enormous compared with a single bidder's requests. Its proof is given in [24].

Lemma 4: If $c \ge \alpha\bar{n}$ for some $\alpha \ge 1$, then $\gamma_a(c) \ge (1-\frac{1}{\alpha})\bar{\gamma}_a(c)$.

Now, instead of calculating the optimal capacity segmentation, the provider solves an approximate problem by replacing γ_a with its upper bound $\bar{\gamma}_a$ in the original problem (4), i.e.,

$$\begin{aligned}\bar{\Gamma}^t(C^t) = \mathbf{E}\Big[\max_{0\le C_a^t\le C^t}\big\{&\bar{\gamma}_a(C_a^t)+\gamma_r(C^t-C_a^t) \\ &+\mathbf{E}_X\big[\bar{\Gamma}^{t+1}(C_a^t+X)\big]\big\}\Big]. \end{aligned} \quad (13)$$

The boundary conditions are $\bar{\Gamma}^{T+1}(c) = 0$ for all $c = 0, 1, \ldots, C$.

Let $\tilde{C}_a^t$ be the optimal solution to (13). Then $\tilde{C}_a^t$ is used as an approximate, sub-optimal solution to the original problem (4). In particular, the provider allocates $\tilde{C}_a^t$ instances to the auction market, generating revenue

$$\tilde{\Gamma}^t(C^t) = \mathbf{E}\Big[\gamma_a(\tilde{C}_a^t) + \gamma_r(C^t - \tilde{C}_a^t) + \mathbf{E}_X\big[\tilde{\Gamma}^{t+1}(\tilde{C}_a^t + X)\big]\Big]. \quad (14)$$

We adopt the above sub-optimal allocation $\tilde{C}_a^t$ as the approximate solution to (4) for two reasons: 1) It closely approaches the optimal revenue, and 2) it contains some optimization structures that significantly reduce the computational complexity.

To see the first argument, we show that as long as the auction market has high demand, the proposed approximation realizes the optimal revenue almost for sure. The following proposition formalizes this statement. Its proof is given in Appendix.

Proposition 6: The approximate allocation is *asymptotically optimal* as the number of bidders tends to infinity. Formally, $\tilde{\Gamma}^t(C^t) \to \Gamma^t(C^t)$ w.p. 1 for all t and C^t if $N_a^t \to \infty$ for all t.

We believe that the optimality condition $N_a^t \to \infty$ is not a restrictive requirement in practice for a large cloud provider, as it always has a huge amount of users requesting virtual instances. In fact, even for cases where there are not too many users bidding in periodic auctions, the proposed approximation still generates near-optimal revenue for a cloud provider. We later verify this point via extensive simulations in Sec. VI.

We now show that (13) has an important optimization structure that leads to an efficient solution within $O(C^2)$. First, we see that $\bar{\gamma}_a(\cdot)$ is concave, as stated below.

Lemma 5: Given $\mathbf{n}$ and $\mathbf{v}$, $\bar{\gamma}_a(c)$ defined in (12) is concave. That is, $\nabla\bar{\gamma}_a(c) = \bar{\gamma}_a(c) - \bar{\gamma}_a(c-1)$ is decreasing w.r.t. c.

Lemma 5 suggests the concavity of $\bar{\Gamma}^t(\cdot)$ as follows.

Lemma 6: For every $\tau = t, \ldots, T$, $\bar{\Gamma}^\tau(C^\tau)$ is increasing and concave for all $C^\tau = 0, 1, \ldots, C$.

This concavity finally leads to an interesting structure described in the following proposition.

Proposition 7: For every realization $\mathbf{n}$ and $\mathbf{v}$ at time $\tau = t, t+1, \ldots, T$, let $\tilde{C}_a^\tau(C^\tau)$ be the optimal solution to (13). For all $C^\tau = 0, 1, \ldots, C$, we have

$$\tilde{C}_a^\tau(C^\tau + 1) - 1 \le \tilde{C}_a^\tau(C^\tau) \le \tilde{C}_a^\tau(C^\tau + 1). \quad (15)$$

Due to space constraints, the detailed proofs of Lemmas 5 and 6, as well as Proposition 7, are all given in [24].

Proposition 7 plays a key role in reducing computational complexity: It indicates that previously calculated results can be *reused* in subsequent computations. We therefore carry out dynamic programming from the last stage T and proceed backwards to t. Within each stage τ, $\bar{\Gamma}^\tau(C^\tau)$ is sequentially computed as $C^\tau = C, C-1, \ldots, 0$. When computing $\tilde{C}_a^\tau(C^\tau)$, instead of exhaustively searching the entire solution space from 0 to C, one only needs to try two possible values, $\tilde{C}_a^\tau(C^\tau + 1)$ and $\tilde{C}_a^\tau(C^\tau + 1) - 1$, and the one resulting in higher revenue is selected as $\tilde{C}_a^\tau$. The entire computation only takes $O(C^2)$ operations.

In terms of computational efficiency, the approximate solution significantly outperforms the optimal one, as the total capacity of a provider is usually enormous in practice. As an example, when $C = 10^5$, the approximation is 10^5 times faster than the exact solution.

TABLE I
REVENUE GAP BETWEEN THE APPROXIMATION AND THE UPPER BOUND.

	$\lambda = 100$	$\lambda = 200$	$\lambda = 500$
Revenue gap	0.81%	0.52%	0.38%

VI. Simulation Results

We evaluate the revenue performance of the proposed approximate solution via extensive simulations. We adopt a typical scenario where $C = 10^5$. That is, the provider is able to host up to 10^5 virtual instances of a certain type simultaneously. We simulate the markets for 100 time periods. In each period t, cloud users arrive into the system following a Poisson process with intensity λ, which are then randomly split into the pay-as-you-go and auction markets with equal probability. Our evaluation adopts three demand patterns — low, medium, and high, with λ being 100, 200, and 500, respectively. For the pay-as-you-go market, each user's demand is modeled by a random variable uniformly distributed in $[1, 1000]$, the price p_r is normalized to 1, and the instance return probability q is taken as 0.5. For the auction market, each bidder i's request n_i is modeled by an i.i.d. random variable uniformly distributed in $[1, 1000]$, and its affordable price v_i is i.i.d. exponential with mean $\mathbf{E}[v_i] = 0.5p_r = 0.5$. We enable short predictions and set the prediction window $w = 5$. Each result below has been averaged over 1000 runs.

A. Revenue Performance

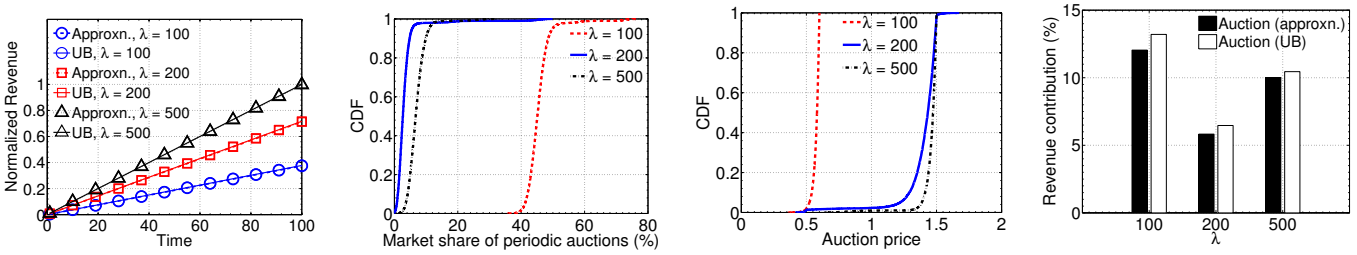
We first evaluate the proposed near-optimal segmentation scheme by comparing its revenue (i.e., $\tilde{\Gamma}^t$ defined in (14)) against the theoretical upper bound (i.e., $\bar{\Gamma}^t$ defined in (13)). The results are illustrated in Fig. 4a, where all data is normalized by the maximum upper-bound revenue.

Fig. 4a shows that our approximation design closely approaches the optimal solution. Even compared with the theoretical revenue upper bound $\bar{\Gamma}^t$, the gap is almost negligible, less than 1% in all cases, as summarized in Table I.

Also note that the revenue gap is diminishing as the market demand increases. This conforms to the asymptotic optimality concluded from Proposition 6: The higher the market demand is, the more close the approximate solution approaches the optimal revenue.

B. Capacity Segmentation and Auction Prices

We now analyze how the capacity is allocated to the two markets under the near-optimal segmentation strategy. Fig. 4b



(a) Normalized revenue vs. time.

(b) CDF of the market share of periodic auctions.

(c) CDF of the auction price.

(d) Revenue contribution of periodic auctions.

Fig. 4. Performance evaluation of the approximate capacity segmentation algorithm, where "UB" stands for upper bound while "approxn." is short for approximation.

illustrates the CDF of the *market share* of periodic auctions in all three demand patterns. Here, the market share is defined as the ratio, between the capacity allocated to the auction market and the entire capacity that the provider has. It is worth mentioning that the allocated capacity might not be fully used to accommodate auction bidders, even for the case where the auction demand exceeds the supply. The provider would strategically reserve some instances by rejecting low-bid requests, since accepting them lowers the clearing price, which may decrease the revenue.

As illustrated in Fig. 4b, when demand is low (i.e., $\lambda = 100$), about half of the capacity is allocated to the auction market, leading to a 50% market share. Fig. 4c shows the corresponding clearing price that is around the mean bid $\mathbf{E}[v_i]$ of auction bidders. In this case, since cloud instances are over-provisioned, some of them are auctioned at a discounted price to increase the revenue. It is worth mentioning that though auction bidders enjoy using the resources at a lower price, they bear the risks that the services might be interrupted.

As demand increases, the market share drops, while the auction price rises. For the case where $\lambda = 200$, Fig. 4b shows that almost all instances are hosted to accommodate pay-as-you-go requests, with less than 10% capacity allocated to auction markets. This is essentially due to the simulation settings that instances are less valued in periodic auctions than they are in pay-as-you-go market, as the mean bid is only half of the pay-as-you-go price (i.e., $\mathbf{E}[v_i] = 0.5p_r$). In this case, pay-as-you-go requests are considered more profitable than auction bids. Only a few high-value bids are accepted by the provider, resulting in a higher clearing price in the auction channel as illustrated in Fig. 4c.

It is interesting to observe that, when demand keeps increasing, the market share of periodic auctions rebounds, which is shown in Fig. 4b with $\lambda = 500$. In this case, the entire market demand significantly exceeds the provider's capacity. As a result, more high-bid requests are received from the auction market. Since these requests are more profitable than those in the pay-as-you-go market, the provider fulfills them by allocating more resources to the auction market. The clearing price is also observed to rise in Fig. 4c.

All discussions above show that augmenting pay-as-you-go pricing with periodic auctions essentially increases the provider's ability to respond to demand uncertainties. Periodic auctions help to fulfill some leftover revenue when resources are over-provisioned in the pay-as-you-go market. On the other hand, it extracts more revenue by charging high prices to high-bid requests when demand exceeds supply.

C. Comparisons Between Pay-as-You-Go and Auctions

The two markets do not make equal revenue contributions. As presented in Fig. 4d, the pay-as-you-go market contributes more than 85% revenue to the provider in all three demand patterns. Note that the pay-as-you-go market takes up only 66% of the overall demand[3]. Therefore, it provides a disproportionately large share of revenue. Similar observations are made when different demand ratios between the two markets are considered.

By offering guaranteed services with a static price, instances in the pay-as-you-go market often demand a higher premium than those in the auction market. For this reason, pay-as-you-go requests are usually more profitable than most auction bids, and are accepted at a higher priority for revenue maximization. Table II further validates this point, where the request acceptance rates are listed for all three demand patterns. We see that pay-as-you-go requests are generally accepted with a considerably higher probability than auction bids. However, this does not mean that auction bidders are always secondary customers. As stated in Proposition 5, those who bid sufficiently high will always be accommodated first. In our simulation, these are the top 5% bidders. As illustrated in Table II, their requests are least affected by the specific demand pattern. Therefore, the auction market offers an option to the users to increase the priority of their requests.

VII. Conclusions

In this paper, we investigate the problem of optimal capacity segmentation in an EC2-like cloud market with the regular pay-as-you-go pricing augmented by periodic auctions. To this end, we analytically characterize the revenue of uniform-price auctions, and present an optimal design with maximum revenue. Contrary to the well-known result that uniform-price

[3]New demand arrivals are equal for both markets, but each new pay-as-you-go instance requires twice the capacity of each new auction instance since $q = 0.5$.

TABLE II
AVERAGE REQUEST ACCEPTANCE RATES

	Pay-as-you-go users	**Auction users**
$\lambda = 100$	100%	36.9%
$\lambda = 200$	88.9%	5.7%
$\lambda = 500$	63.1%	5.2%

auctions have suffered from the "demand reduction" in general, our design achieves truthfulness in cloud environments where partial fulfillment is unacceptable to users. We further connect our design to the EC2 spot market, showing that the two are equivalent in terms of their market response. Based on the established analysis for the auction channel, we formulate the capacity segmentation problem as a Markov decision process. Realizing that the exact solution is computationally prohibitive in practical settings, we present an asymptotically optimal solution that reduces the computational complexity from $O(C^3)$ to $O(C^2)$, which is significant for cloud providers with large capacities. All our theoretical results are further validated by extensive simulation studies.

APPENDIX

This section proves the asymptotic optimality of the approximate solution proposed in Sec. V. We need the following technical lemma.

Lemma 7: If $N_a^t \to \infty$ for all t, then for any given $\alpha < \infty$, $\tilde{\Gamma}^t(C^t) \geq (1 - \frac{1}{\alpha})\bar{\Gamma}^t(C^t)$ holds for all t and C^t.

Proof: Given an arbitrary $\alpha < \infty$, we prove by induction.

Basis: Show that the statement holds for $t = T + 1$. This is indeed the case due to the boundary conditions, i.e., $\tilde{\Gamma}^{T+1}(C^{T+1}) = \bar{\Gamma}^{T+1}(C^{T+1}) = 0$.

Inductive step: Suppose the statement holds for $t + 1$. We show it also holds for t. It suffices to consider the following two cases.

Case 1: $\tilde{C}_a^t \geq \alpha\bar{n}$. In this case, by Lemma 4, we have $\gamma_a(\tilde{C}_a^t) \geq (1 - \frac{1}{\alpha})\bar{\gamma}_a(\tilde{C}_a^t)$. Also, from the induction assumptions, we know $\tilde{\Gamma}^{t+1} \geq (1 - \frac{1}{\alpha})\bar{\Gamma}^t$ w.p. 1. Substituting these two inequalities back to (13) and (14) lead to the statement: $\tilde{\Gamma}^t(C^t) \geq (1 - \frac{1}{\alpha})\bar{\Gamma}^t(C^t)$ w.p. 1. (Note that $\tilde{C}_a^t$ is the optimal solution to (13).)

Case 2: $\tilde{C}_a^t < \alpha\bar{n}$. In this case, let Y be the number of top bidders with unit demand, i.e., $n_i = 1$ and $v_i = \bar{v}$. Suppose $P(n_i = 1, v_i = \bar{v}) = p > 0$. We know $Y \sim Bin(N_a^t, p)$. Let $\mathcal{N}(\mu, \sigma^2)$ be the normal distribution. For any finite k, we have

$$\begin{aligned}\lim_{N_a^t \to \infty} P(Y = k) &= \lim_{N_a^t \to \infty} B(N_a^t, k, p) \\ &= \lim_{N_a^t \to \infty} \mathcal{N}(N_a^t p, kp(1-p)) \\ &= 0 \ ,\end{aligned}$$

where the second equality holds because of the Central Limit Theorem. This essentially indicates that $P(Y \geq \tilde{C}_a^t) = 1$ for any finite $\tilde{C}_a^t$. Therefore, all $\tilde{C}_a^t$ instances can be allocated to top bidders with unit demand, which implies $\gamma_a(\tilde{C}_a^t) = \bar{\gamma}_a(\tilde{C}_a^t)$ w.p. 1. Now substituting this equality to (13) and (14) and applying the induction assumptions, we see the statement holds, i.e., $\tilde{\Gamma}^t(C^t) \geq (1 - \frac{1}{\alpha})\bar{\Gamma}^t(C^t)$ w.p. 1. ■

We are now ready to prove the asymptotic optimality of the approximate solution.

Proof of Proposition 6: By Lemma 7, for any given $\alpha < \infty$, $\tilde{\Gamma}^t(C^t) \geq (1 - \frac{1}{\alpha})\bar{\Gamma}^t(C^t) \geq (1 - \frac{1}{\alpha})\Gamma^t(C^t)$ w.p. 1. Also note that $\tilde{\Gamma}^t(C^t) \leq \Gamma^t(C^t)$ (because $\Gamma^t(C^t)$ is the optimal solution). We finally have

$$(1 - \frac{1}{\alpha})\Gamma^t(C^t) \leq \tilde{\Gamma}^t(C^t) \leq \Gamma^t(C^t) \tag{16}$$

held for any finite α w.p. 1. This essentially indicates that $\tilde{\Gamma}^t \to \Gamma^t$ w.p. 1. ■

REFERENCES

[1] Amazon EC2 Pricing, http://aws.amazon.com/ec2/pricing/.
[2] Windows Azure, http://www.microsoft.com/windowsazure.
[3] Google AppEngine, http://code.google.com/appengine.
[4] GoGrid Cloud Hosting, http://www.gogrid.com.
[5] RackSpace Cloud Hosting, http://www.rackspace.com/cloud.
[6] JoyentCloud, http://www.joyentcloud.com.
[7] ElasticHosts, http://www.elastichosts.com/.
[8] Y. Hong, M. Thottethodi, and J. Xue, "Dynamic server provisioning to minimize cost in an IaaS cloud," in *Proc. ACM SIGMETRICS*, 2011.
[9] K. Vermeersch, "A broker for cost-efficient qos aware resource allocation in EC2," Master's thesis, University of Antwerp, 2011.
[10] Q. Zhang, E. Gürses, R. Boutaba, and J. Xiao, "Dynamic resource allocation for spot markets in clouds," in *Proc. HOT-ICE*, 2011.
[11] A. Danak and S. Mannor, "Resource allocation with supply adjustment in distributed computing systems," in *Proc. IEEE ICDCS*, 2010.
[12] O. A. Ben-Yehuda, M. Ben-Yehuda, A. Schuster, and D. Tsafrir, "Deconstructing Amazon EC2 Spot Instance pricing," in *Proc. IEEE CloudCom*, 2011.
[13] P. Klemperer, *Auctions: Theory and Practice*. Princeton University Press, 2004.
[14] R. Engelbrecht-Wiggans, J. List, and D. Reiley, "Demand reduction in multi-unit auctions with varying numbers of bidders: Theory and field experiments," *Inernational Economic Review*, vol. 47, no. 1, pp. 203–31, 2006.
[15] H. Etzion, E. Pinker, and A. Seidmann, "Analyzing the simultaneous use of auctions and posted prices for online selling," *Manufacturing & Service Operations Management*, vol. 8, no. 1, pp. 68–91, 2006.
[16] R. Caldentey and G. Vulcano, "Online auction and list price revenue management," *Management Science*, vol. 53, no. 5, pp. 795–813, 2006.
[17] G. Vulcano, G. Van Ryzin, and C. Maglaras, "Optimal dynamic auctions for revenue management," *Management Science*, vol. 48, no. 11, pp. 1388–1407, 2002.
[18] J. Gallien, "Dynamic mechanism design for online commerce," *Operations Research*, vol. 54, no. 2, pp. 291–310, 2006.
[19] H. Fujiwara and K. Iwama, "Average-case competitive analyses for ski-rental problems," *Algorithmica*, vol. 42, no. 1, pp. 95–107, 2005.
[20] R. Randhawa and S. Kumar, "Usage restriction and subscription services: Operational benefits with rational users," *Manufacturing & Service Operations Management*, vol. 10, no. 3, pp. 429–447, 2008.
[21] E. Caron, F. Desprez, and A. Muresan, "Forecasting for grid and cloud computing on-demand resources based on pattern matching," in *Proc. IEEE CloudCom*, 2010.
[22] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, *Algorithmic Game Theory*. Cambridge University Press, 2007.
[23] R. B. Myerson, "Optimal auction design," *Mathematics of Operations Research*, vol. 6, no. 1, pp. 58–73, 1981.
[24] W. Wang, B. Li, and B. Liang, "Towards optimal capacity segmentation with hybrid cloud pricing," University of Toronto, Tech. Rep., 2011. [Online]. Available: http://iqua.ece.toronto.edu/~bli/papers/capseg.pdf
[25] J. Bulow and J. Roberts, "The simple economics of optimal auctions," *Journal of Political Economy*, vol. 97, no. 5, pp. 1060–1090, 1989.
[26] W. Vickrey, "Counterspeculation, auctions, and competitive sealed tenders," *The Journal of Finance*, vol. 16, no. 1, pp. 8–37, 1961.