An Auction Mechanism for a Cloud Spot Market

Adel Nadjaran Toosi,
Kurt Van Mechelen, and
Rajkumar Buyya

December 2, 2014

Abstract

Dynamic forms of resource pricing have recently been introduced by cloud providers that offer Infrastructure as a Service (IaaS) capabilities, in order to maximize profit and balance resource supply and demand. The design of a mechanism that efficiently prices perishable cloud resources in line with a provider's profit maximization goal remains an open research challenge however. In this paper, we propose an adaptation of the Consensus Revenue Estimate auction mechanism to the setting of a multi-unit online auction for cloud resources. The mechanism is envy-free, has a high probability of being truthful, and generates a near optimal profit for the provider. We combine the proposed auction design with a scheme for dynamically calculating reserve prices based on data center Power Usage Effectiveness (PUE) and electricity costs. Our simulation-based evaluation of the mechanism demonstrates its effectiveness under a broad variety of market conditions. In particular, we show how it improves on the classical uniform price auction and investigate the value of prior knowledge on the execution time of virtual machines, for maximizing profit.

1 Introduction

The increased adoption and maturity of cloud computing offerings has been accompanied by a growing role and significance of pricing mechanisms for trading computational resources. Especially Infrastructure as a Service (IaaS) cloud providers that offer computational services in the form of Virtual Machine (VM) instances with specific resource characteristics, have gradually expanded their pricing plans in order to maximize their profits and further attract demand. Currently, the most widely used model remains a fixed *pay-as-you-go* pricing plan wherein the consumer is charged the amount of time a VM instance was used at a fixed rate. However, the fact that computational resources sold by a cloud provider can be characterized as a *non-storable* or *perishable* commodity[1],

[1]Note that resources tied to a VM are qualified as non-storable (perishable), as a non-used hour of CPU time or memory space can never be reclaimed and therefore wastes data center capacity.

combined with the fact that demand for computational resources is non-uniform over time, motivates the use of dynamic forms of pricing in order to optimize revenue [1]. Through price adjustment based on actual (and possibly forecasted) supply and demand conditions, consumers can be incentivized to acquire spare capacity or shift demand from on-peak to off-peak hours. Consequently, both profit and consumer satisfaction can be increased.

Market-based pricing mechanisms that solicit reports (*bids*) from consumers and subsequently use an *allocation rule* and *pricing rule* to compute the allocation of resources to consumers and their associated prices respectively, are well fit to realize such dynamic forms of pricing. Recently, they have received significant attention for selling underutilized capacity in cloud infrastructures [2]. Well-designed auction mechanisms can be particularly effective since they: 1) incentivize users to bid in a truthful manner (i.e., report the price they are willing to pay for resources), 2) ensure resources are allocated to those who value them the most, and 3) correctly price resources in line with supply and demand conditions by creating competition among buyers.

Amazon Web Services (AWS) has adopted an auction-like approach to expand its pricing plans with Spot Instances for the Amazon Elastic Compute Cloud (EC2). In this scheme, consumers communicate their bids for a VM instance hour to AWS. Subsequently, AWS reports a market-wide *spot price* at which VM instance use is charged, while terminating any instances that are executing under a bid price that is lower than the market price. Although Amazon is not the only provider to offer dynamic pricing, it is currently the only IaaS provider that publicly offers an auction-like mechanism for selling IaaS resources. Nevertheless, attempts for creating such mechanisms have already been reported by other companies [3] and have also received attention by academia [4–7].

AWS has revealed no detailed information regarding their auction mechanism and the calculation of the spot price. At present, the design of an efficient, fair, and profit-maximizing auction mechanism for pricing cloud computing resources is an open research challenge, and of great interest to cloud providers.

In this chapter, we design such an auction mechanism aimed at generating additional profit from the spare capacity of non-storable resources available in cloud data centers. We refer to the marketplace in which this mechanism is used to sell VMs as the cloud *spot market* (Fig. 1).

The spare capacity that can be offered by an IaaS cloud provider in the spot market is usually much larger than the demand[2]. Therefore a provider is potentially able to accept all consumer requests. In this context, popular auction mechanisms such as the second-price Vickrey [10] auction may fail to generate a reasonable revenue for the provider. In general, when supply exceeds demand, bidders are less motivated to bid competitively, which can prevent providers to collect an optimal revenue. Providers therefore require an auction mechanism that can maximize revenue while incentivizing bidders to reveal their true value.

[2]This can be explained by the promise of Clouds providing infinite capacity of resources [8] and recent reports that suggest the overall utilization in large data centers is lower than 30% most of the time [9].

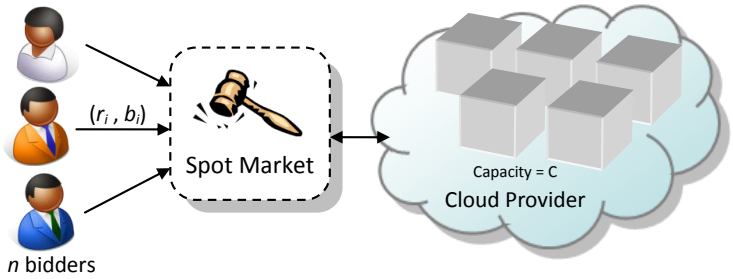


Figure 1: Spot market and auction mechanism

Hence, we restrict our focus to *truthful* auction designs. An auction mechanism is truthful if for each bidder i and any choice of order values by all other bidders, bidder i's *dominant strategy* is to report her private information with respect to her order truthfully. A strategy is dominant if a bidder cannot increase the pay-off derived from participating in the mechanism, by diverging from it.

If perfect knowledge about the distribution from which the bidders valuations are drawn is available, such a truthful auction mechanism can be designed [4]. Unfortunately, this is not always the case and pricing depends heavily on the accuracy of the underlying market analysis. Such analysis also needs to be updated frequently in order to adapt to changes in the market. Moreover, since customers of cloud services are distributed globally and experience different latency for the same service, assuming that the valuations for all bidders are drawn i.i.d. might be invalid.

This chapter focuses on designing a truthful auction mechanism for a cloud spot market aimed at maximizing the cloud provider's profit. The cloud spot market context influences our auction design in the sense that the design needs to: support multi-unit bids, operate in an online recurrent manner, result in a single market-wide price and fair outcomes, operate under a limitation of the maximal quantity a consumer can request, operate without prior knowledge on the distribution of bidders' valuations, and finally, allow for reserve prices to be set during oversupply conditions. The chapter's key contributions are:

- The design and application of a multi-unit, online recurrent auction mechanism within the context of IaaS resource trading. The mechanism extends the off-line single-round auction with a single-unit demand model of the consensus revenue estimate (CORE) mechanism proposed by Goldberg and Hartline [11], to a two-dimensional bid domain. The proposed auction mechanism is envy-free, truthful with high probability and generates near optimal profit for the provider. It adopts a greedy approach for maximizing provider profits in the online setting. It is initially designed for the unlimited supply case, and is subsequently extended to the limited supply case.

- The evaluation of the proposed mechanism with respect to revenue generation, truthfulness, and bid rejection rates. Extensive simulation results are presented that demonstrate that it achieves near optimality w.r.t. maxi-

mizing revenue without requiring prior knowledge on the order distributions. It is also shown to achieve low bid rejection rates, mitigating the *bidder drop problem* in online mechanisms [1]. We compare the proposed mechanism to a clairvoyant and non-clairvoyant variant of the Optimal Single Price Auction and to the Uniform Price Auction.

- A clairvoyant optimal auction mechanism (HTA-OPT) that uses dynamic programming to calculate the set of accepted bids. HTA-OPT serves as a benchmark that is used to quantify the efficiency loss caused by the lack of information on the amount of time a bidder wants to run a VM, when applying the allocation rule in a single auction round.

- The presentation of a method for dynamically computing a *reserve price*, based on a coarse grained data center power usage model that can be used by the provider within the proposed auction mechanism. The resulting prices are shown to correspond to actual minimal spot prices observed on the EC2 spot market.

The remainder of this chapter is organized as follows: After reviewing related work in Section 2, we introduce required terminology and notations in Section 3. Sections 4, 5 and 6 discuss respectively the competitiveness, truthfulness and envy-freeness properties for our auction design. Section 7 describes the proposed auction mechanism, while Section 8 focuses on the limited supply setting and the computation of the reserve price in that setting. Section 9 describes the online version of the proposed auction mechanism and mechanisms used in the comparative analysis. Our experimental evaluation of the mechanism can be found in Section 10. We compare its performance to the Optimal Single Price Auction and the Uniform Price Auction, and investigate the impact of perfect knowledge on the execution time of a VM. We also provide simulation results concerning the probability that any bidder can benefit from an untruthful reporting of the number of VM instances required. Our conclusions follow in Section 11.

2 Related Work

The use of an auction-like mechanism to sell spare capacity in cloud data centers was pioneered in late 2009 by Amazon. In Amazon's spot market, customers bid the maximum hourly price they are willing to pay to obtain a VM instance[3]. All instances incur a uniform charge, the *spot market price*. According to Amazon, this price is set dynamically based on the relationship of supply and demand over time. A unique feature of spot instances is that the provider has the right to terminate them when their associated bid falls below the spot market price. As a result, the resulting quality of service (QoS) may be lower compared to *on-demand* and *reserved* instances, depending on the bid made. Current spot

[3] `http://aws.amazon.com/ec2/spot-instances/`

market data shows customers can acquire VMs at price reductions between 50% to 93% compared to on-demand instances.

Amazon has revealed little information on the pricing and allocation rules of their pricing mechanism. Ben-Yehuda et al. [5] examined the price history of the EC2 spot market through a reverse engineering process, and found that the mechanism was not completely driven by demand and supply. Their analysis suggests that spot prices are usually drawn from a tight, fixed price interval, and reflect a random non-disclosed reserve price. In this chapter, we propose an auction mechanism with transparent allocation and pricing rules, while sharing similar properties with the EC2 spot market.

Several authors have presented strategies for customers to utilize Amazon spot instances (cost-)effectively [12–16]. However, as of yet a limited amount of work has been conducted that focuses on the design of auction mechanisms to the benefit of cloud providers, and the associated algorithms for allocating resources and capacity planning to maximize the provider's revenue. The problem of dynamically allocating resources to different spot markets in order to maximize a cloud provider's revenue has been investigated by Zhang et al. [7]. Danak and Manno [6] present a uniform-price auction for resource allocation that suits the dynamic nature of grid systems. Mihailescu and Teo [17] investigate Amazon EC2's spot market as a case in a federated cloud environment. They argue that spot pricing used by Amazon is truthful only in a market with a single provider, and show that rational users can increase their utility by being untruthful in a federated cloud environment. Recently, Zaman et al. have investigated the applicability of combinatorial auction mechanisms for allocation and pricing of VM instances in cloud computing [18].

Wang et al. [4] proposed an optimal recurrent auction for a spot market based on the seminal work of Myerson [19]. The mechanism was designed in the context of optimally segmenting the provider's data center capacity between on-demand and spot market requests. Their work differs from ours since they adopt a Bayesian approach wherein it is assumed that the customers' private values are drawn from a known distribution. They also propose a truthful dynamic auction [20] that periodically computes the number of instances to be auctioned off in order to maximize providers revenue. Unlike EC2 spot marketplace, their approach offers guaranteed services (i.e., instances are never be terminated by the provider) and constant price over time (i.e. as the price is set for the user, it remains constant as long as the user holds the instance). Their auction charges each user a different price and does not generate a market-wide single price. Moreover, their auction mechanism requires a priori known distribution of valuations and near future demand prediction.

In contrast, we propose an auction mechanism designed to maximize profit based on a competitive auctioning framework proposed by Goldberg and Hartline [21]. The mechanism computes a uniform price outcome, and focuses on maximizing profit when the seller knows very little about the bidders valuations. In order to achieve truthfulness in this context, we rely on a *consensus estimation* technique [11].

Our work differs from that of Goldberg et al., in several aspects. First,

their analysis relies on the assumption that each customer is restricted to formulate unit demand, which is not the case for cloud consumers as they can ask and bid for multiple VM instances. Consequently, we revisit the definition and truthfulness analysis of the mechanism for the multi-unit case. Second, their auction mechanism is designed for off-line single-round scenarios. The context of a cloud spot market however requires an online auction where customers arrive over time and resources allocated by VM instances can be released and subsequently reused by other consumers. We adopt a greedy approach in realizing the online character of the auction, and investigate its performance compared to a clairvoyant optimal mechanism that relies on dynamic programming. Finally, the production cost of goods is not taken into account in their work. In the IaaS setting, taking this cost into account is important as a seller has the option to either shut down server capacity or sell the capacity at a given *reserve price*. We add such reserve pricing to the mechanism and introduce a coarse-grained cost model to determine that.

Lee and Szymanski [1] have proposed an auction mechanism for time sensitive e-services where services must be resold for future time periods repeatedly. They investigated the *bidder drop problem* in recurrent auctions that occurs when the least wealthy bidders tend to withdraw from the future auction rounds due to repeatedly losing the auction. Our proposed auction is not specifically designed to address this issue, however our evaluation shows that it rejects a lower number of requests compared to the Optimal Single Price auction while generating near optimal revenue.

3 Preliminaries and Notation

Consider a cloud provider with capacity C for a specific type of VM. That is, at a given time t up to C instances of the specific type can be hosted simultaneously. The provider runs a sealed-bid auction, $\mathcal{A}$, to sell this capacity. First, we assume the case that the provider's capacity far exceeds the total demand, in line with the cloud's promise of delivering an unlimited supply of resources. Subsequently, we generalize the results to a scenario in which supply is limited and lower than total demand.

Suppose there are n customers joining the auction at time t. Each bidder i ($1 \leq i \leq n$) requires q_i VM instances and has a private valuation v_i, denoting the maximum amount i is willing to pay for each VM instance per time slot (e.g., 1 hour). Customers submit an order (request), (r_i, b_i), where r_i represents the number of required VM instances and b_i the bid price. We denote by $\mathbf{d}$ the vector of all submitted orders. The ith element of $\mathbf{d}$, d_i, is the order by customer i.

Given $\mathbf{d}$, the provider (auctioneer) computes an allocation vector, $\mathbf{x} = (x_1, x_2, ..., x_n)$, and a price vector, $\mathbf{p} = (p_1, p_2, ..., p_n)$. The ith component x_i of the allocation vector indicates whether bidder i receives the r_i VMs requested in its order (if $x_i = 1$) or not ($x_i = 0$). A bidder for which $x_i = 1$ is called a *winner* and pays the corresponding price p_i, otherwise, the bidder is called a

loser and does not make any payment to the mechanism. As we focus on single price auctions, all p_i are equal for all winning bidders and we therefore refer to the sale price as p. *Partial fulfillment* of requests, in which only a fraction of the number of VM instances requested is allocated to a winning bidder, is only considered in the case of limited supply and when $b_i = p$. We allow for partial fulfillment for those orders in line with the behavior of the EC2 spot market.

Note that bidders are individually rational users that try to maximize their utility. Therefore, as long as it is deemed beneficial, a customer will strategically misreport her bid or the required number of VMs i.e., $b_i \neq v_i$ or $r_i \neq q_i$, where v_i and q_i are private information known only to customer i. We define customer i's utility at time t for one time slot of VM usage as follows:

$$u_i(r_i, b_i) = \begin{cases} (q_i v_i - r_i p_i) x_i \,, & \text{if } b_i \geq p_i \text{ and } r_i \geq q_i; \\ 0 \,, & \text{otherwise.} \end{cases} \tag{1}$$

The values of r_i and v_i for each customer are drawn from distributions that are unknown to the provider. Customer i's optimal bidding strategy must be defined so that it maximizes i's utility over all time slots. However, assuming that customers are not aware of the future and have no time-dependent valuation for resources, we define the utility function in (1) based on a single time slot. Winners in an auction round are awarded their requested VM instances and automatically attend the next round of the auction until they cancel their requests on their own account or they lose the auction. In the latter case, VMs held by an outbid customer are terminated by the provider without any prior notice.

The *holding time* of a VM is the specific amount of time a customer wants to run the VM. The VM's actual holding time might be smaller than the expected time if it is terminated by the provider instead of the owner. The holding time of a VM by the customer is not known to the provider (or to the mechanism) in advance. Therefore, in our model, a provider acts in a greedy manner to maximize revenue according to the arriving requests and the current existing requests in each round of auction. This can be modeled as a single round auction which is recurrently conducted by the provider as new requests arrive or current requests are terminated. In section 10.4, we compare the performance of this greedy strategy to the optimal strategy that has prior knowledge on the VM holding time. From this point onwards, we limit our discussion only to a single round of the auction. In Section 9, we introduce the recurrent version of the mechanism.

4 Competitive Framework

The revenue generated by auction $\mathcal{A}$ in a time slot equals:

$$\mathcal{A}(\mathbf{d}) = \sum_i r_i p_i \,. \tag{2}$$

The problem of maximizing revenue in an auction for cloud resources can be solved optimally if the seller knows the distribution from which the bidders' valuations are drawn i.i.d. [4]. In conventional economics this is called *Bayesian Optimal Mechanism Design* [19, 22]. However, we assume that the distributions from which the bidder's private information are drawn are unknown to the provider. Therefore, we base our approach on the competitive mechanism design proposed by Goldberg et al. [21]. We will compare the revenue attained by our mechanism to that of the *Optimal Single Price auction* for the unlimited capacity case.

Definition 1. The *Optimal Single Price auction*, $\mathcal{F}$, is defined as follows: Let $\mathbf{d}$ be an order vector. Without loss of generality, suppose the components of $\mathbf{d}$ are sorted in descending order by bid values. So, $(r_i,\, b_i)$ is the ith largest bid in $\mathbf{d}$ regardless of r_i. The auction $\mathcal{F}$ on input $\mathbf{d}$ determines the value k such that $b_k \sum_{i=1}^{k} r_i$ is maximized. We denote by $\sigma_k(\mathbf{d})$ the sum of the number of requested instances in the sorted vector of orders from the first order to kth order ($\sigma_k(\mathbf{d}) = \sum_{i=1}^{k} r_i$). All bidders with $b_i \geq b_k$ win at price b_k and all remaining bidders lose. Thus, the revenue of $\mathcal{F}$ on input $\mathbf{d}$ is

$$\mathcal{F}(\mathbf{d}) = \max_i \;\; b_i \sigma_i(\mathbf{d})\,. \tag{3}$$

If more than one value of i maximizes $b_i\sigma_i(\mathbf{d})$, choosing the price point that results in a lower transacted volume is preferable considering the cost of accommodating VM instances (e.g., electricity cost). From this point forward, we assume $\mathbf{d}$ is sorted decreasingly by bids values (b_i), unless otherwise mentioned.

We are interested in an auction mechanism that is competitive with $\mathcal{F}$ on every possible input; however, if a single bidder's utility dominates the total utility of the other bidders, no auction can compete with $\mathcal{F}$ as shown by Goldberg et al. [21]. We do not consider this to be an issue in our setting, because the cloud environment can be viewed as a mass-market where the number of winners of the optimal single price auction is typically large. In a mass-market, removing one order does consequently not change the maximum extractable profit significantly.

Definition 2. (*Mass-market*): Let $\mathcal{F}(\mathbf{d})$ be the revenue of $\mathcal{F}$ and $h_b(\mathbf{d})$ denote the maximum value of b in $\mathbf{d}$, then $\mathcal{F}(\mathbf{d}) \gg h_b(\mathbf{d})$ in mass-markets, which implies that $\mathcal{F}$ sells $m \gg 1$ units.

We say that auction $\mathcal{A}$ is competitive if there exists a constant β such that $\mathcal{A}(\mathbf{d}) \geq \mathcal{F}(\mathbf{d})/\beta$. For a randomized mechanism[4], the previous equation for competitiveness becomes:

$$\mathbf{E}[\mathcal{A}(\mathbf{d})] \geq \frac{\mathcal{F}(\mathbf{d})}{\beta}\,.$$

Assuming the fact that $\mathcal{F}$ sells at least m units, we define $\beta(m)$-competitiveness for a mass-market as below:

[4]The mechanism's allocation and/or pricing rule procedure has a randomized component.

Definition 3. Auction $\mathcal{A}$ is $\beta(m)$-competitive for a mass-market if for all order vectors $\mathbf{d}$ such that $\mathcal{F}$ sells at least m units, we have:

$$\mathbf{E}[\mathcal{A}(\mathbf{d})] \geq \frac{\mathcal{F}(\mathbf{d})}{\beta(m)}\,. \tag{4}$$

5 Truthfulness

Let $\mathbf{d}_{-i}$ denote the vector of orders $\mathbf{d}$ with $(r_i,\, b_i)$ removed that is:

$$\mathbf{d}_{-i} = ((r_1, b_1), \ldots, (r_{i-1}, b_{i-1}), (r_{i+1}, b_{i+1}), \ldots, (r_n, b_n))\,,$$

and further introduce the notation $\mathcal{F}((r_i, b_i)\,, \mathbf{d_{-i}}) = \mathcal{F}(\mathbf{d})$.

Proposition 1. $\mathcal{F}$ is not truthful.

Proof. Suppose $\mathcal{F}$ is truthful, then utility for each bidder i is maximized if $b_i = v_i$ and $r_i = q_i$ for any choice of $\mathbf{d}_{-i}$.

Consider $\mathbf{d}$ as any arbitrary vector of orders, assume $\mathcal{F}(\mathbf{d})$ is the maximum revenue by $\mathcal{F}$ and b_k is the sale price. Suppose $\mathcal{F}_2(\mathbf{d})$ is the second largest revenue which can be obtained by $\mathcal{F}$ and we limit $\mathbf{d}$ to those vectors such that $\mathcal{F}(\mathbf{d}) > \mathcal{F}_2(\mathbf{d})$. Given a fixed $\mathbf{d}_{-k}$, r_k, q_k, bidder k is able to reduce her bid from b_k to b'_k and still be the winner as long as $\mathcal{F}((r_k, b'_k)\,, \mathbf{d}_{-k}) > \mathcal{F}_2(\mathbf{d})$. As a result, fixing other variables and considering that bidder k is a winner ($x_k = 1$), bidder k is able to increase her utility from $q_k v_k - r_k b_k$ to $q_k v_k - r_k b'_k$. So there exists a $\mathbf{d}_{-i}$ and a bidder i such that u_i can be increased by misreporting i's true value, i.e., $b_i \neq v_i$. This contradicts the supposition that $\mathcal{F}$ is truthful. A similar proof can be constructed for the number of requested instances, which we omit here for space considerations. □

In order to create a truthful auction, an intuitive idea is to design the mechanism in a way that a bidder believes that her own order does not affect the price she pays. This is called an order-independent auction since the price the bidder is offered in the auction is independent of the bidder's bid value [21]. An order-independent auction can be viewed as a function that maps $\mathbf{d}_{-i}$ to a price for each bidder.

Definition 4. The *order-independent auction* offers a price p_i to bidder i computed by the function f according to the order vector $\mathbf{d}_{-i}$, i.e., $p_i = f(\mathbf{d}_{-i})$. If bidder i's bid is greater or equal to p_i ($b_i \geq p_i$), the bidder wins the auction ($x_i = 1$) and pays p_i; otherwise the bidder loses the auction ($x_i = 0$) and pays zero.

Lemma 1. The *order-independent auction* is truthful.

Proof. Following Definition 4, bidder i's order does not affect the price she ends up paying, so the bidder is not able to increase her utility by changing her order. As a result, the bidder has no incentive to misreport her bid or quantity levels as this does not change the amount she pays. □

Following [21], we introduce the *optimal order-independent auction*. To define it, first we define the notion of the optimal single sale price for a set of orders.

Definition 5. Let $\mathbf{d}$ be a sorted vector of orders by descending values of bids. Denote $opt(\mathbf{d})$ the optimal single sale price for $\mathbf{d}$ that maximizes the revenue for the auctioneer, i.e.,

$$opt(\mathbf{d}) = \underset{b_i}{\operatorname{argmax}} \; b_i \sigma_i(\mathbf{d}) \,. \tag{5}$$

Now we can define the *optimal order-independent auction*, which is a truthful auction, as follows:

Definition 6. The *optimal order-independent auction* is defined by the order-independent function f such that $f(\mathbf{d}_{-i}) = opt(\mathbf{d}_{-i})$.

Unfortunately, even though the optimal order-independent auction is truthful, it has two main characteristics that make it unsuitable for our purposes. Firstly, it is not single price and secondly, a bidder j might lose the auction while bidder i with $b_i < b_j$ wins and is charged $p_i < b_j$. In this case the auction's outcome is not fair and the losing bidder envies the winning bidder's outcome. This might happen as the sale price for bidder i is computed based on $\mathbf{d}_{-i}$ which is different for each bidder. Proof of Lemma 2 provides examples of these outcomes.

6 Envy-freeness

In an *envy-free auction* no bidder can increase its utility by adopting another bidder's outcome. For our case, an envy-free auction requires a single sale price. All bidders willing to pay this price are provided with VM instances and charged at that price uniformly.

In this work, it will be irrelevant how bids that equal the sale price are treated, however, we assume that they are always provided with VM instances if the provider's capacity allows for it. Note that, according to the utility function in Equation 1, the utility value (u_i) is always zero for those bidders with true bid values (v_i) equal to p, irrespective of them winning or losing. Therefore, those bidders are assumed to have no preference over the two possible outcomes.

Lemma 2. The *optimal order-independent auction* is not envy-free.

Proof. It suffices to construct an example showing that the optimal order-independent auction is not single price. Consider three bidders with the following orders $d_1 = (1, \$8)$, $d_2 = (2, \$7)$, and $d_3 = (4, \$2)$. In order to calculate the sale price for each bidder i, first we obtain $\mathbf{d_{-i}}$ by removing bidder i's order from $\mathbf{d}$. Then $opt(d_{-i})$ is computed according to (5). Performing the above process for all bidders, we obtain the outcome for each bidder as follows. Bidder one and two win the auction and pay \$7 and \$2 respectively, while bidder three

loses the auction and pays zero. This shows that optimal order-independent auction is not single price.

In addition, the order-independent auction is not fair as there are situations in which a bidder might lose the auction while another bidder with a lower bid price wins the auction. Consider four bidders with orders $d_1 = (2, \$13)$, $d_2 = (5, \$3)$, $d_3 = (1, \$2)$ and $d_4 = (20, \$1)$. Bidder one and three win the auction and both pay bidder four's bid price, i.e., \$1 per instance, while bidder two with a bid price higher than bidder three ($\$3 > \$2$) loses the auction. □

Goldberg and Hartline [23] showed that no truthful, envy-free auction can be constant competitive and they provided the lower bound of $log(n)/log\ (log(n))$ with n the number of bidders. In order to obtain a constant competitive auction mechanism, we relax the assumption of truthfulness and extend the proposed Consensus Revenue Estimate (CORE) auction [23] for our case. The proposed auction is envy-free but is only truthful with *high probability*.

Definition 7. An auction is truthful with probability $1 - \epsilon$ if the probability that any bidder can benefit from an untruthful bid is at most ϵ. If ϵ is inverse polynomial in some specified parameters of the auction (such as the number of items or bidders) then we say the mechanism is *truthful with high probability*.

In the following section, we show that the proposed auction mechanism is truthful with high probability with respect to the bid price dimension. We also provide simulation results concerning the probability that any bidder can benefit from an untruthful reporting of the number of VM instances required.

7 Extended Consensus Revenue Estimate Auction

Recall that the optimal order-independent auction in Section 5 is truthful since it is order-independent. Due to the fact that it is not single price, and therefore not envy-free, it is not suitable for our problem context. The question therefore arises as to how a single price can be computed for an order-independent auction while attaining the revenue of the optimal auction, that is, $\mathcal{F}(\mathbf{d})$. It is clear that $\mathcal{F}(\mathbf{d})$ cannot be computed from $\mathbf{d}_{-i}$ and consequently, a function f that generates the optimal sale price based on $\mathbf{d}_{-i}$ cannot be built. Therefore, we are interested in a mechanism that provides us with a sufficiently accurate estimate of $\mathcal{F}(\mathbf{d})$ that is constant on $\mathbf{d}_{-i}$ for all i (i.e., it achieves *consensus*). If $\mathcal{F}(\mathbf{d}_{-i})$ is limited by a constant fraction of $\mathcal{F}(\mathbf{d})$, it is possible to pick a good estimate of $\mathcal{F}(\mathbf{d})$ such that it achieves consensus with high probability [23]. In the remainder of this section, we will outline how this estimate is computed.

In mass-markets such as clouds, $\mathcal{F}(\mathbf{d})$ is much larger than the highest bid. Let $h_b(\mathbf{d})$ denote the maximum bid value in $\mathbf{d}$, then $\mathcal{F}(\mathbf{d}) \geq \alpha h_b(\mathbf{d})$ in mass-markets, which implies that $\mathcal{F}$ sells at least α units.

Let m ($m \geq \alpha$) be the number of sold units in $\mathcal{F}$. If m is sufficiently large and the maximum number of units that can be requested by a customer is

limited, removing an order does not change $\mathcal{F}(\mathbf{d})$ considerably. We show this in Lemma 3.

Enforcing a restriction on the maximum number of VM instances that can be simultaneously acquired by a customer is reasonable and done by public cloud providers such as Amazon[5]. Such restriction reduces the chance of system stability being threatened by very large unpredicted requests. In addition, it reduces the risk of starvation for customers with small requests in the presence of wealthy customers.

Lemma 3. Let r denote the supremum of the number of requested units in $\mathbf{d}$, i.e., $r_i \leq r$ for all bidders, $1 \leq i \leq n$. If m, the number of sold units in $\mathcal{F}$, is sufficiently large, then for any i,

$$\frac{m-r}{m}\mathcal{F}(\mathbf{d}) \leq \mathcal{F}(\mathbf{d}_{-i}) \leq \mathcal{F}(\mathbf{d})\,. \tag{6}$$

Proof. Without loss of generality, suppose $\mathbf{d}$ is sorted in descending order of bids (b_i), i.e., $b_1 \geq b_2 \geq ... \geq b_n$. Suppose k is the rank of the bidder in $\mathbf{d}$ whose bid maximizes $b_i\sigma_i(\mathbf{d})$, i.e., $\mathcal{F}(\mathbf{d}) = b_k\sigma_k(\mathbf{d})$. By removing order i from $\mathbf{d}$, the maximum reduction in $\mathcal{F}(\mathbf{d})$ is r_ib_k (when $i \leq k$), and the minimum reduction is zero (when $i > k$). Therefore,

$$\mathcal{F}(\mathbf{d}) - r_ib_k \leq \mathcal{F}(\mathbf{d}_{-i}) \leq \mathcal{F}(\mathbf{d})\,.$$

$$m = \sum_{j=1}^{k} r_j \Rightarrow b_k = \frac{\mathcal{F}(\mathbf{d})}{m}\,,$$

$$r_i \leq r \Rightarrow r_ib_k \leq r\frac{\mathcal{F}(\mathbf{d})}{m} \Rightarrow$$

$$\frac{m-r}{m}\mathcal{F}(\mathbf{d}) \leq \mathcal{F}(\mathbf{d}_{-i}) \leq \mathcal{F}(\mathbf{d})\,.$$

□

We introduce ρ for $\frac{m}{m-r}$. In mass-markets, $\frac{1}{\rho}\mathcal{F}(\mathbf{d}) \leq \mathcal{F}(\mathbf{d}_{-i}) \leq \mathcal{F}(\mathbf{d})$, meaning that $\mathcal{F}(\mathbf{d}_{-i})$ is at least a constant fraction of $\mathcal{F}(\mathbf{d})$.

The Extended Consensus Revenue Estimate Auction (Ex-CORE) combines two general ideas as its name implies: *consensus estimation* and *revenue extraction*. For consensus estimation, it picks a function that estimates $\mathcal{F}(.)$ with high quality and achieves consensus with high probability. A function that works well in our case is g, defined as:

$g(\mathcal{F}(.)) = \mathcal{F}(.)$ rounded down to the nearest c^{l+u}

where $c > \rho$ is a constant chosen as to maximize the quality of the estimation, u is a uniform random value on $[0, 1]$, and l is the largest integer so that $c^{l+u} \leq \mathcal{F}(.)$.

[5] `http://aws.amazon.com/ec2/faqs/\#How_many_Spot_Instances_can_I_request`

Lemma 4. [11] For $c > \rho$ and any $\mathbf{d}$ with $\frac{1}{\rho}\mathcal{F}(\mathbf{d}) \leq \mathcal{F}(\mathbf{d}_{-i}) \leq \mathcal{F}(\mathbf{d})$, the probability that g outputs a value which is constant on all $\mathbf{d}_{-i}$ (i.e., achieves consensus) is $1 - \log_c \rho$.

Lemma 5. [11] If *payoff* for g, γ_g, is defined as:

$$\gamma_g(\mathcal{F}(.)) = \begin{cases} g(\mathcal{F}(.))\,, & \text{if } g \text{ achieves consensus;} \\ 0\,, & \text{otherwise.} \end{cases} \tag{7}$$

then for all $\mathcal{F}(.)$, we have:

$$\mathbf{E}[\gamma_g(\mathcal{F}(.))] = \frac{\mathcal{F}(.)}{ln(c)}\left(\frac{1}{\rho} - \frac{1}{c}\right). \tag{8}$$

Let us now discuss how to choose the value of c. We are interested in the expected payoff to be large relative to $\mathcal{F}(.)$, i.e., $\mathbf{E}[\gamma_g(\mathcal{F}(.))]/\mathcal{F}(.)$ is large over different values of $\mathcal{F}(.)$. For a fixed value of ρ, we can choose the value of c that maximizes $\frac{1}{ln(c)}\left(\frac{1}{\rho} - \frac{1}{c}\right)$. This function is differentiable on $c \in (1, \infty)$ and it has an absolute maximum on that interval. Therefore, by taking the derivative of it w.r.t. c and setting it to zero, we have:

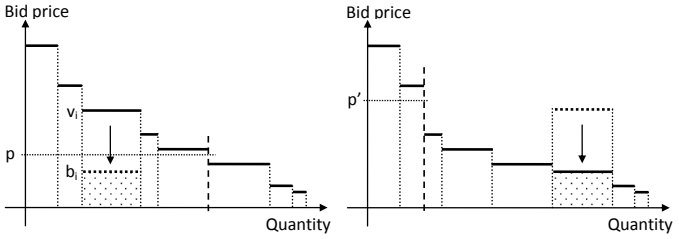
$$\begin{gathered} \frac{\partial \mathbf{E}[\gamma_g(\mathcal{F}(.))]/\mathcal{F}(.)}{\partial c} = 0 \Rightarrow \\ \frac{\rho\, ln(c) + \rho - c}{\rho\, c^2\, ln^2(c)} = 0, \;\; \rho > 1, \;\; c > \rho \Rightarrow \\ \rho\, ln(c) + \rho - c = 0 \end{gathered} \tag{9}$$

Note that (9) does not have an exact solution and needs to be solved by numerical methods.

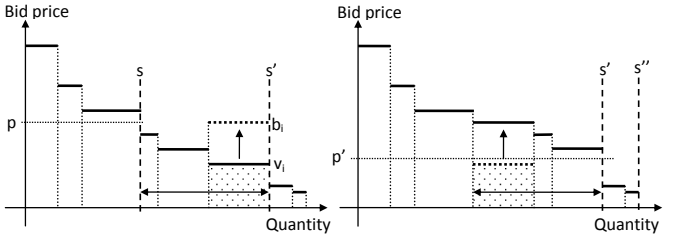
The second component of Ex-CORE, a revenue extraction mechanism, extracts a target revenue from the set of bidders if this is possible. The algorithm is based on the cost sharing mechanism proposed by Moulin and Shenkar [24]. Given an order vector $\mathbf{d}$ sorted in descending order of bids and a target revenue R, the revenue extractor function $e_R(\mathbf{d})$ finds the largest k such that $R/\sigma_k(\mathbf{d}) \geq b_k$. In other words, it finds the k bidders with the highest bid values that allow for the extraction of R. R is then shared among these k bidders based on the number of requested instances by each bidder, that is, each of these bidders are charged $R/\sigma_k(\mathbf{d})$. If no subset of bidders can share R, the auction has no winners.

Lemma 6. Given a target revenue R, the revenue extraction mechanism is truthful for the price dimension but not for the quantity dimension.

Proof. Without loss of generality, we consider $\mathbf{d}$ as sorted. The revenue extraction mechanism is truthful if $u_i(q_i, v_i) \geq u_i(r_i, b_i)$ for all values of b_i and r_i and for every bidder i, $1 \leq i \leq n$. First, we show that given a fixed r_i any untruthful submission of the bid price, i.e., $b_i \neq v_i$ decreases bidder's i utility. It suffices to consider the following two cases:



(a) Reporting $b_i < v_i$ increases the price to $p' > v_i$.



(b) Reporting $b_i > v_i$ decreases the price to $p' > v_i$.

Figure 2: Effect of misreporting true value on the sale price. Truthful submission leads to (a) winning and (b) losing.

Case 1: Suppose the truthful submission ($v_i = b_i$) leads to bidder i winning the auction, it is easy to verify that reporting $b_i > v_i$ only decreases the rank of bidder i in $\mathbf{d}$, assuming $\mathbf{d}$ remains unchanged except for bidder i. Therefore, it does not change the sale price and as a result, bidder i's utility also remains unchanged.

If bidder i reports $b_i < v_i$, as long as $b_i \geq p$ (p is the sale price), p remains unchanged. Hence, bidder i's utility does not increase or decrease. However, as soon as $b_i < p$, bidder i loses the auction, the sale price rises, and the bidder's utility drops to zero. This is illustrated in Fig. 2(a). Consequently, submitting $b_i < v_i$ might not improve bidder i's utility and might reduce it to zero.

Case 2: Suppose the truthful submission ($v_i = b_i$) leads to the bidder losing the auction, then reporting $b_i < v_i$ would clearly not change the zero utility of the bidder.

If reporting $b_i = v_i$ leads to bidder i losing the auction, it follows that $p > v_i$. Assume $p = R/s$, where s is the sum of the number of requested units by largest group of k bidders with highest bid values that can at least generate a revenue of R. Consider $s' = \sigma_i(\mathbf{d})$, as a result $s' > s$, since we know b_i is a losing bid.

Suppose bidder i reports her bid $b_i > v_i$, we argue that new sale price p' is always larger than v_i ($p' > v_i$). That is, increasing b_i might increase s up to s' at most. This is shown in Fig. 2(b).

Using reductio ad absurdum, assume by increasing b_i, s can be increased to a value $s'' > s'$. Hence, we know that there is a bidder j whose bid price, b_j, is larger than R/s'' ($R/s'' \leq b_j$). We know that $i < j$ and $b_j \leq v_i$, because $s'' > s'$ requires j to be placed after i in the sorted vector. If $R \leq s''b_j$ after

increasing b_i, then $R \leq s''b_j$ before increasing as well, because bidder i is placed in the lower rank either bidding at $b_i = v_i$ or $b_i > v_i$ in the sorted vector of orders. That is, bidder i is a winner in either of cases. This contradicts our initial assumption that reporting $b_i = v_i$ leads to bidder i losing the auction. So, $s'' \leq s' \Rightarrow p' > v_i$.

Hence, bidding $b_i > v_i$ leads to negative utility for bidder i and bidder i would be worse off.

Second, we provide an example that demonstrates that the revenue extraction mechanism is not truthful for the quantity dimension. That is, bidders are able to increase their utility by misreporting their required number of instances. Assume $R = \$7$ and an order vector $\mathbf{d} = \{(1, \$8), (5, \$1)\}$. Bidder one is charged $7/1 = 7$ and bidder two loses according to the revenue extraction mechanism. The utility for bidder one is then computed as follows: $1 \times 8 - 1 \times 7 = 1$. Now, consider that bidder one misreports 2 as the required number of instances. Then, the largest group of bidders able to share R includes the orders of both bidders. Therefore the price for bidder one is $7/7 = 1$ and its utility is computed as: $1 \times 8 - 2 \times 1 = 6$.

□

Definition 8. *Extended Consensus Revenue Estimate Auction (Ex-CORE)*: For constant c, and a random value u, uniformly chosen from $[0, 1]$, find $g(.)$ as $\mathcal{F}(.)$ rounded down to nearest c^{l+u} for integer l. The sale price by Ex-CORE is then defined as $p = e_R(\mathbf{d})$ where $R = g(\mathcal{F}(\mathbf{d}))$.

Lemma 7. For order vector $\mathbf{d}$, constant c and a choice of u, if $g(\mathcal{F}(\mathbf{d}_{-i})) = R$ for all i, $1 \leq i \leq n$, i.e., it is a consensus, then the Ex-CORE auction is truthful with respect to bid prices.

Proof. It suffices to show that if $g(\mathcal{F}(\mathbf{d}_{-i})) = R$ for all i, no bidder can increase her utility by bidding any value other than their true bid value. Note that If $g(\mathcal{F}(\mathbf{d}_{-i})) = R$ for all i then $g(\mathcal{F}(\mathbf{d})) = R$. Now consider that bidder i submits an order (r_i, b_i) where $b_i \neq v_i$ resulting in $\mathbf{d}'$ ($\mathbf{d}'$ is identical to $\mathbf{d}$ except for bidder i's bid price).

As long as $g(\mathcal{F}(\mathbf{d}')) = g(\mathcal{F}(\mathbf{d})) = R$, bidder i is not able to benefit out of misreporting. Because the sale price p is computed as $p = e_R(\mathbf{d})$, and according to Lemma 6, the revenue extraction mechanism is truthful. Therefore, bidder i's utility cannot be improved by misreporting v_i; thus bidder i's best strategy is to bid at v_i.

The proof is in fact very straightforward. For every user i, since $\mathcal{F}(\mathbf{d}_{-i}) = \mathcal{F}(\mathbf{d})$, changing bid b_i to b'_i will lead to a new order vector $\mathbf{d}'$ the same as the original $\mathbf{d}$ except component i. As a result, $\mathbf{d}'_{-i} = \mathbf{d}_{-i}$. Hence $g(\mathcal{F}(\mathbf{d}'_{-i})) = g(\mathcal{F}(\mathbf{d}_{-i})) = g(\mathcal{F}(\mathbf{d})) = R$. This essentially implies that user i will be given exactly the same price as before. Consequently, the sale price cannot be decreased and bidder i's utility cannot be increased.

□

Proposition 2. The Extended Consensus Revenue Estimate Auction (Ex-CORE) is envy-free, truthful with probability $1 - \log_c \rho$ for the bid price dimension, and $\frac{1}{ln(c)}\left(\frac{1}{\rho} - \frac{1}{c}\right)$-competitive for mass markets.

Proof. Definition 8 and Lemmas 4, 5 and 7 are enough to prove the proposition. □

7.1 Discussion

The Ex-CORE auction is not two-dimensionally truthful because the revenue extraction mechanism is not truthful for the quantity dimension (Lemma 6). In the cloud spot market however, no customer has an incentive to request fewer instances than needed (as $u_i(r_i, b_i) = 0$ whenever $r_i < q_i$). Our detailed investigation of the proposed mechanism shows that bidders are able to increase their utility in some cases by requesting a higher number of instances than what they actually require. Devising a two-dimensional truthful mechanism for this highly complex strategy space remains as a future work. Nevertheless, we believe that the proposed mechanism retains its practical value due to several key reasons.

First, users who misreport the required number of instances end up paying for a higher number of instances. In order to increase utility, the increment in r_i must cause a sufficient reduction in the market price to compensate for the surplus cost a bidder pays for the additional instances. Considering that the bidder is not aware of the other orders, there is always a risk of decrease in utility by misreporting.

Second, $\mathcal{F}(\mathbf{d})$ is monotonically increasing w.r.t r_i (the rationale is intuitive) and Ex-CORE calculates the sale price based on the estimation of $\mathcal{F}(\mathbf{d})$. Given that r (the maximum number of requested instances) is a constant and $m \to \infty$ in a cloud mass-market, in expectation R (the estimated value of $\mathcal{F}(\mathbf{d})$) rises as the bidder increases demand. The revenue extraction mechanism computes the price by $R/\sigma_k(\mathbf{d})$. Therefore the risk of increasing the market price increases by misreporting the number of required instances, as the numerator of the fraction (R) is increasing while there is no certainty about the decrease or increase in the denominator ($\sigma_k(\mathbf{d})$).

Last but not the least, r is constrained by a limit. As the number of sold instances in the cloud market is usually high, the effect on the market price of a bidder misreporting demand is typically low given the assumption of non-collusive behavior of bidders.

In Section 10, we demonstrate through simulation that in markets of sufficient size, an individual bidder indeed has a very low probability of gaining utility by misreporting VM demand.

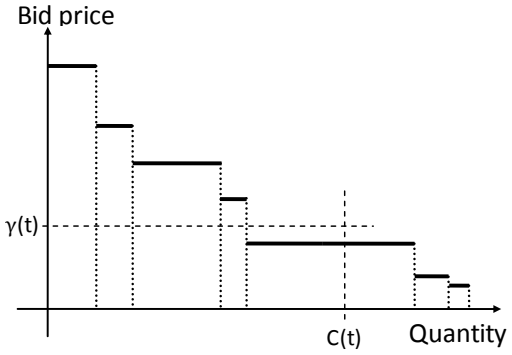


Figure 3: Supply limited by capacity and reserve price at time t

8 Limited Supply and Reserve Price

Up to this point, we have considered an unlimited capacity setting. In reality, however, situations arise wherein a cloud provider needs to reject requests due to lack of supply. We modify the auction mechanisms to take into account that $C(t)$, the number of VM instances available for sale at time t, can be lower than the demand.

As the provider wishes to maximize revenue, it can select a set of high-value bidders such that the total amount of requested VMs by this set is smaller or equal to $C(t)$. This set of bidders subsequently participates in the auction mechanism for the unlimited supply case, while the remaining bids are rejected. Fig. 3 depicts how supply is limited by $C(t)$. This method allows us to extend our discussion into the bounded supply case. In order to be envy-free in the bounded supply case, we need to ensure that none of the bidders win at a price lower than the highest losing bid. Therefore, we ensure that $p = max(b_{lost}, p)$, with b_{lost} the highest losing bid.

If a bidder accepts partial fulfillment of an order, the fraction of required instances that fits in the provider's available capacity can be allocated. When multiple winning consumers are subject to such partial delivery, ties can be broken randomly.

8.1 Reserve Price

If profit instead of revenue is of concern, the provider needs to take its costs for delivering a VM instance into account. Let $\gamma(t)$, the *reserve price* at time t, be the lowest possible price that the provider accepts for one slot of usage of a VM instance, at time t; orders with bids below this level are ignored by the auction. In this section, we propose a method for a provider to compute $\gamma(t)$. Fig. 3 depicts how the order vector is shaped by $\gamma(t)$.

The reserve price for most perishable goods and services is considerably low

at their expiration time. For instance, the reserve price for flight seats is theoretically negligible; as soon as boarding is closed on a particular flight, all the unsold seats on that flight are completely wasted. Thus, selling a remaining seat at a reasonable low price is often a better option compared to wasting the seat capacity without generating any revenue. However, there is a fundamental difference between cloud resources and other perishable goods and services. A significant part of the service cost in cloud data centers is related to power consumption of physical servers. The cost of power drawn by servers and associated cooling systems is comparable to the amortized capital investments for purchasing the servers themselves [25]. Thus, when considering the perishable nature of VM services, taking into account the *marginal cost* of instantiating a VM is important in this case[6].

The overall cost of the data center, $C_{overall}$, can be divided into capital and operational costs, $C_{overall} = C_{cap} + C_{opr}$. The parameter C_{cap} includes upfront investments and all one-time expenses that are depreciated over the lifetime of the data center, e.g., those related to the purchase of land, buildings, construction, buying physical servers and software, installing power delivery and cooling infrastructures etc. C_{opr} includes electricity costs, staff salaries and ISP costs. Operational costs can further be categorized as being *fixed* or *variable*, $C_{opr} = C_{opr_{fixed}} + C_{opr_{var}}$. The parameter $C_{opr_{fixed}}$ includes costs that remain identical no matter the data center is operating at full capacity or not, e.g., staff salaries. However, components of $C_{opr_{var}}$ may increase or decrease depending on data center utilization, e.g., electricity costs.

The provider is not able to avoid the incurrence of C_{cap} and $C_{opr_{fixed}}$, whereas $C_{opr_{var}}$ can be avoided to a large extent. $C_{opr_{var}}$ over any specific time period is dominated by the cost of power consumption, C_{pwr}, and can be strongly approximated by it ($C_{opr_{var}} \approx C_{pwr}$).

Cloud providers are able to measure instant power consumption in the data center. Knowing the power consumption and electricity prices, C_{pwr} can be easily calculated. We argue that the cloud provider should define the reserve price in a way that accommodating a VM with a specific bid must at least generate sufficient revenue to offset the contribution of VM to $C_{opr_{var}}$. Assuming all VMs are of the same type, $\gamma(t)$ can therefore be derived as follows:

$$\gamma(t) = C_{pwr}/VM_n(t), \tag{10}$$

where $VM_n(t)$ is the number of running VMs in the data center at time t, and C_{pwr} is the cost of power consumption at that time. Knowing the electricity price, φ, and total data center power consumption, $Power_{total}$, C_{pwr} can be computed as $C_{pwr} = Power_{total} \times \varphi$. As $\gamma(t)$ is primarily affected by factors such as IT load, electricity price, data center outside air temperature and humidity [26], it should vary dynamically.

Because we resort to simulation for the experimental performance evaluation of our proposed solution, we require a model for C_{pwr}. Detailed modeling of data

[6]In economics, the marginal cost is the change in total cost that arises as a result of one additional unit of production.

center power usage however is difficult because of the complexity and diversity of the infrastructure [26]. Consequently, we propose abstract model based on the concept of Power Usage Effectiveness (*PUE*).

8.2 Power Usage Efficiency Model

PUE is a measure of how efficiently a data center consumes its power. It is computed as the ratio of total data center power consumption, $Power_{total}$, to IT load power, $Power_{IT}$, i.e., power consumed by servers, storage and network equipment:

$$PUE = Power_{total}/Power_{IT}. \tag{11}$$

PUE measures the power overhead consumed in supporting the IT load. The overhead is caused by cooling and humidification systems (e.g., chiller), power distribution (e.g., PDU), power conditioning system (e.g., UPS), and lighting. Ideally $PUE = 1$. Inefficient data centers have a *PUE* of 2.0 to 3.0, while *PUE* scores lower than 1.14 are advertised by leading companies such as Facebook and Google [27]. *PUE* reported in this way is usually an average value over a specific period of time (e.g., one year), whereas instant *PUE* is not a constant value. The efficiency of the data center varies over time by changes in the data center conditions.

One of the most important conditions is the outside ambient temperature [28], as the energy required to remove heat generated within the data center grows with it [26]. To some degree, outside air humidity affects cooling power as well, but we do not consider it in this work in order to limit model complexity.

A second important condition that changes over time and affects *PUE* is the IT load. This follows from the fact that the efficiency of power conditioning system and cooling equipments increases under higher load [29]. We represent IT load by the percentage of ON physical servers in the data center (referred to as *data center utilization*). We model *PUE* as function of load and outside ambient temperature, i.e., $PUE = f(load, temp)$. In order to simplify the model, we assume that every server in the data center consumes its peak load power if it is ON; and none otherwise. $Power_{IT}$, is therefore computed according to (12).

$$Power_{IT} = N_{Srv-ON} \times Power_{Srv}, \tag{12}$$

where N_{Srv-ON} is the number of non-idle servers in the data center, and $Power_{srv}$ is the peak power consumption by servers. The contribution of networking equipment in (12) is not taken into account as it is small and its power draw does not vary significantly with data center load [26].

In this study, we assume that the provider commits to provide the actual amount of resources required by a VM, regardless of the actual resource usage pattern of the applications it executes. Moreover, we assume the cloud provider periodically packs the data center's workload into a minimum number of servers, powering off any inactive ones.

Algorithm 1 The Online Ex-CORE Auction

Input: $\mathbf{d}$, p_{cur}, p_{optprv} ▷ $\mathbf{d}$ is the list of orders, sorted in descending order of bids, p_{cur} is current market price, p_{optprv} is the optimal single price in the previous round.

Output: p ▷ Sale Price

1: $p_{opt} \leftarrow opt(\mathbf{d})$
2: **if** $p_{opt} = p_{optprv}$ **then**
3: **return** p_{cur}
4: **end if**
5: $r \leftarrow$ the largest r_i in $\mathbf{d}$
6: $m \leftarrow \underset{\sigma_i(\mathbf{d})}{\text{argmax}}\ b_i\sigma_i(\mathbf{d})$
7: **if** $m \leq r$ **then**
8: **return** p_{opt} ▷ single optimal price
9: **else**
10: $\rho \leftarrow \frac{m}{m-r}$
11: Find c in $\rho\, ln(c) + \rho - c = 0$
12: $u \leftarrow rnd(0,1)$ ▷ chosen uniformly random on [0,1]
13: $l \leftarrow \lfloor log_c(\mathcal{F}(\mathbf{d})) - u \rfloor$
14: $R \leftarrow c^{(l+u)}$
15: $j \leftarrow$ the largest k such that $\frac{R}{\sigma_k(\mathbf{d})} \geq b_k$
16: **return** $\frac{R}{\sigma_j(\mathbf{d})}$
17: **end if**

9 Auction Mechanisms and Benchmarks

In this section we review the different auction mechanisms that are included in our experimental evaluation.

Optimal Single Price Auction (OPT): The extractable revenue in a single-round, single-price auction is at most $\mathcal{F}(\mathbf{d})$ which can be achieved by an optimal price choice. Since we are interested in maximizing the provider's revenue, we use the Optimal Single Price Auction (OPT) described in Definition 1 as a benchmark. In the online version of OPT, the auction is executed upon every arrival of an order or termination of an instance.

Online Extended Consensus Revenue Estimate Auction (Online Ex-CORE): Details of the Ex-CORE auction can be found in Section 7. Our online version of Ex-CORE (outlined in Algorithm 1) records the optimal sale price computed by OPT in the previous round, and updates the sale price using the Ex-CORE algorithm. Only when the optimal sale price calculated in the current round differs from the one in the previous round of the auction, a new price is computed (lines 1-4). This prevents the market to be exposed to a high number of price fluctuations due to randomness in the Ex-CORE algorithm. Note that it does not violate a possibly existing consensus established in the previous round of the Ex-CORE auction, as arriving or leaving orders have not changed the optimal price.

Lines 5 and 6 compute r, the maximum number of requested units in the

order list, and m, the maximum number of units sold by OPT. As our mechanism is designed to work for mass-market scenarios it requires m to be larger than r ($m \gg r$). In the rare event when this condition would not hold, the algorithm returns the price computed by OPT.

On line 10, ρ is computed, followed by the computation of the optimal value for c, for which we use Newton-Raphson. Subsequently, c is used to generate an estimation of $\mathcal{F}(.)$ that achieves the consensus with high probability (lines 12-14). Finally, the estimated value is converted to the market clearing price through the revenue extraction mechanism.

Holding Time Aware Optimal Auction (HTA-OPT): Due to a lack of prior knowledge on the holding time of VMs, the online version of the Ex-CORE auction operates in a greedy manner, as it attempts to maximize revenue given the newly arriving order and the existing orders at a given time. In order to quantify the efficiency loss caused by this lack of information, we use HTA-OPT as a benchmark algorithm that uses prior knowledge on VM holding times. HTA-OPT takes into account the fact that an order with a long holding time and a low bid can potentially generate more revenue than a short order with a high bid.

Algorithm 2 calculates the optimal sale price using dynamic programming. The price is computed based on the maximum possible revenue that can be generated by current orders in the system and the corresponding remaining time of these orders. The main reasoning is that if the algorithm sets the price at a specific bid price, all orders with bid prices lower than that price are not available for the next time slot. Assuming bidders are charged on an hourly basis of VM usage, we express duration similarly in an hourly basis. Each partial hour is considered as a full hour (e.g., 2.5 hours is considered as 3 hours of usage).

Algorithm 2 has the following input arguments: the list of orders $\mathbf{d}$, sorted in descending order of bids, an order index i set to the number of orders in the first call of the function, a time slot index t set to 1 for the first call, and a boolean argument $firstCall$ indicating that it is the first call to the function. Lines 5-15 initialize the revenue array rev such that each element in rev is set to the revenue that can be generated in that time slot, provided that the price is set to a corresponding bid price. Line 16 ends the recursion when the termination conditions are reached.

In lines 24-29, the algorithm chooses the most profitable path given two choices for dealing with order i at time t. This is done by recursively computing the total revenue in case the market price is kept below the order's bid at time t ($ans1$), and computing the revenue in case the decision is made to let the market price exceed the order's bid at t ($ans2$).

The most profitable decision path is stored in the dp array with the aggregated revenue for the checked paths. Finally, we find the highest possible revenue within the first column of dp, and return the corresponding price. We break ties by favoring the market price with the lowest transaction volume.

Uniform Price Auction: In the uniform price auction, the provider serves the highest bidder first, allocating the requested number of instances. This is

followed by an allocation for the second highest bidder and so forth until supply is exhausted or there are no more orders. All bidders are charged the lowest winning bid.

10 Performance Evaluation

Our evaluation of the proposed auction framework includes three parts. In the first, we simulate Ex-CORE in a single-round, unlimited supply setting using several order distributions. In the second part, the impact of misreporting the number of required instances on the utility obtained by an individual bidder is explored. The last part evaluates the auction framework under bounded supply. Auctions then occur recurrently by arriving and finishing orders, and the marginal cost of VM production changes dynamically over time.

10.1 Order Generation

Due to the lack of real-world data on bidder valuations and order sizing, we need to resort to a synthetic generation of orders. In line with [21], we adopt the following four distributions for the generation of bids:

1. **uniform (l, h)**: Bid prices are drawn from a uniform distribution bounded by l and h.
2. **normal (μ, σ)**: Bid prices are drawn from a normal distribution with mean μ and standard deviation σ. Bids less than or equal to zero are discarded and a new bid is drawn from the distribution. This causes the normal distribution to be skewed as zero and negative bid values are not permitted.
3. **Zipf (h, θ)**: Bid prices are drawn from a Zipf distribution with parameters h as the highest bid price and parameter θ. This distribution is a generalization of the Pareto principle that 80% of the total bid value originates from 20% of the bidders.
4. **bipolar (l, h)**: Bid prices are generated by randomly choosing either l or h with equal probability.

For requested number of instances in each order, we consider three types of distributions :

1. **constant(ζ)**: The number of instances for all orders equal $\zeta \leq r$ where r is the supremum on the number of requested units.
2. **uniform (l, h)**: The number of instances for an order is drawn from a uniform distribution between $l = 1$ and $h = r$.
3. **normal (μ, σ)**: The number of instances for an order is drawn from a normal distribution as a discrete value with mean μ and standard deviation σ. Values smaller than 1 or larger than r are discarded.

Algorithm 2 Holding Time-Aware Optimal Auction

Input: $\mathbf{d}$, i, t, $firstCall$
Output: p ▷ Sale Price

$maxDuration \leftarrow \max_i(\text{DURATION}(d_i))$ ▷ The duration function returns the time remaining from the holding time of the orders in unit of hours.
$dp[|\mathbf{d}|][maxDuration] \leftarrow \{-1\}$
$rev[|\mathbf{d}|][maxDuration] \leftarrow \{0\}$ ▷ Create dp and rev arrays with $|\mathbf{d}|$ (size of $\mathbf{d}$) rows and $maxDuration$ columns, and initialize all cells with -1 and 0 respectively.
function HTA-OPT($\mathbf{d}, i, t, firstCall$)
    if $firstCall$ **then**
        for $j \leftarrow 1$ to $maxDuration$ **do**
            $prvCount \leftarrow 0$
            for $k \leftarrow 1$ to $|\mathbf{d}|$ **do**
                if $j \leq \text{DURATION}(d_k)$ **then**
                    $rev[k][j] = d_k \times (r_k + prvCount)$
                    $prvCount \leftarrow prvCount + r_k$
                end if
            end for
        end for
    end if
    if $i = 0$ or $t > \text{DURATION}(d_i)$ **then**
        return 0;
    end if
    if $dp[i][t] = -1$ **then**
        $ans1 \leftarrow 0, ans2 \leftarrow 0$
        if $t \leq \text{DURATION}(d_i)$ **then**
            $ans1 \leftarrow \text{HTA-OPT}(\mathbf{d}, i, t+1, false) + rev[i][t]$
        end if
        if $i \geq 1$ **then**
            $ans2 \leftarrow \text{HTA-OPT}(\mathbf{d}, i-1, t, false)$
        end if
        if $ans1 > ans2$ **then** $dp[i][t] \leftarrow ans1$
        else $dp[i][t] \leftarrow ans2$
        end if
    end if
    if $firstCall$ **then**
        $k \leftarrow \text{argmax}_i(dp[i][0])$ ▷ In case of ties, pick lowest i, i.e., higher price and selling less instances
        return b_k
    end if
    return $dp[i][t]$
end function

10.2 Single Round Evaluation

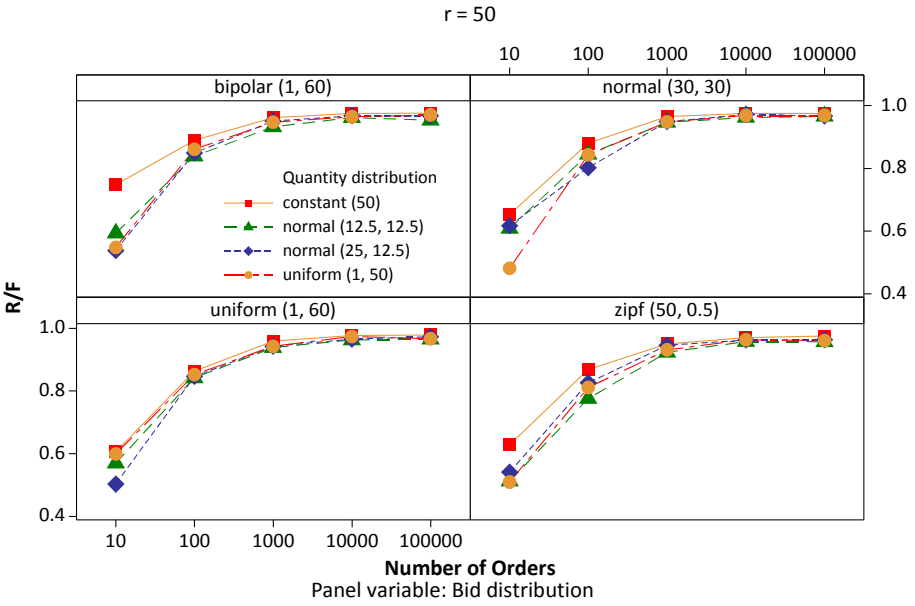


Figure 4: Ratio of gained revenue by the Ex-CORE auction to optimal auction under different distribution of orders.

We investigate the generated revenue for different combinations of distributions for the bid price and the number of requested units per order. The number of orders varies between 10 and 100000 and the ratio of generated revenue by Ex-CORE (R) to $\mathcal{F}$ is reported ($R/\mathcal{F}$). Each experiment is carried out 30 times and the mean value of $R/\mathcal{F}$ is reported. Fig. 4 shows the simulation results when $r = 50$. As the number of orders increases, $R/\mathcal{F}$ approaches 1 regardless of the distribution used for order generation as we expected. Although there is a small difference between the revenue obtained by Ex-CORE for different distributions as shown in Fig. 4, the distribution of orders does not have a significant effect on the generated revenue, especially when the number of orders in the market is large. Fig. 5 shows separate box plots of $R/\mathcal{F}$ for different order distributions when the number of orders equals 100. Statistical analysis certifies that the performance does not change significantly under different order distributions. By design, Ex-CORE does not require a priori knowledge about the order distribution. Therefore, the provider does not need to rely on frequent investigation and monitoring of changes in the market conditions in order to maximize its revenue.

A sensitivity analysis with respect to r showed that its value does significantly impact the results, as long as r is sufficiently small compared to the total demand volume, and the total supply volume in the market is sufficiently large. This can be easily justified by Lemma 3. For brevity, we omit a discussion on

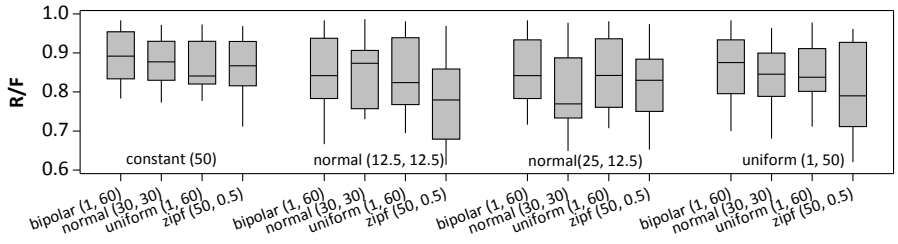


Figure 5: Ratio of gained revenue by the Ex-CORE auction to optimal auction under different distribution of orders when number of orders is 100.
these experiments.

10.3 Evaluation of Misreporting Quantity

As the Ex-CORE mechanism is not two-dimensionally truthful, we investigate the potential for a bidder to gain utility by misreporting the number of required units in her order. First, we generate list of orders with the same settings used for experiment 1 and assume each generated order to be truthful. The utility obtained by every bidder is subsequently calculated according to (1).

Assuming there is no collusion among bidders, we increase the requested number of instances (r_i) for an individual bidder i up to the maximum number of instances that can be requested (r) by the step size of one, while keeping the orders of the other bidders unchanged. For every stepwise increase, we calculate the bidder's utility and compare it with the utility attained under a truthful report. We then compute the probability of a bidder increasing its utility through a misreport of quantity, by dividing the number of cases in which utility increased to the total number of steps.

The experiment is repeated for every bidder with the same random number seed for each step and is subsequently carried out 30 times with different seeds. Fig. 6 shows the mean and box plot of the mean probability of increase in the utility by misreporting the quantity after 30 runs under different order distributions. The constant distribution of the requested units is removed from the set of quantity distributions, as there is no opportunity to change r_i. As one can observe in the figure, the probability of gaining utility through misreports converges to zero as the market size grows under all order distributions. Moreover, there is no predictable pattern for the user to increase the utility value due to the implicit random component of Ex-CORE and the lack of knowledge about other bidders. Fig. 7 shows the maximum increase of the utility value among all bidders, achieved through misreporting quantity. As can be seen in the figure, the maximum possible gain in the utility value for all bidders through misreporting number of instances converges to zero as the market size grows.

10.4 Online Auction Framework Evaluation

We evaluate the profit of the online Ex-CORE auction through simulation. We consider the case where capacity (C) is bounded fixed throughout the simulation

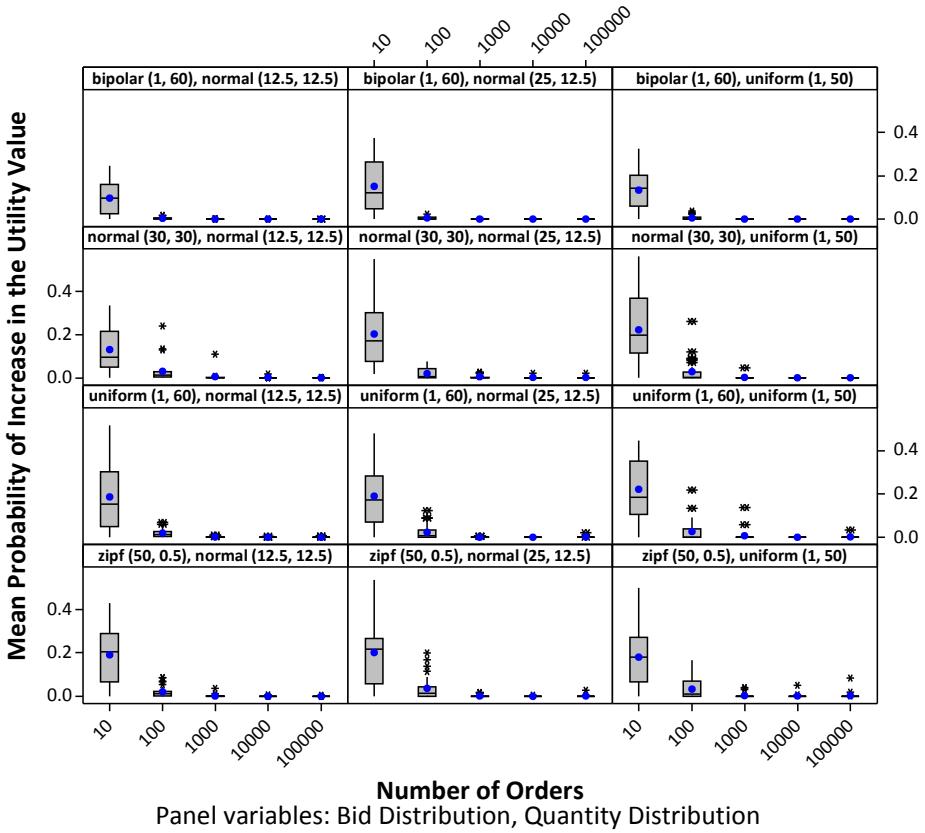


Figure 6: Mean probability of increase in the utility value for bidders by Ex-CORE under different distribution of orders when $r = 50$. A blue circle denotes the mean value.

at $C = 8 \times 10^4$. In real-world scenarios, a provider may offer several pricing plans (e.g., on-demand, reserved pricing plans) or different types of VMs (e.g., small, medium, large). Under such circumstances, the capacity allocated for a specific VM type in the auction market can be dynamically adjusted in order to maximize profit [4, 7]. We consider the auction operates for just one type of VM. Similar auctions can be run separately for the provider's VM types.

We simulate the market for 24 hours. Customers submit their orders independently following a Poisson process with λ set at the total number of requests in the whole simulation divided by 24. As the distribution of the bid prices and the quantity of requested units do not significantly impact the revenue results, we use uniform distributions for both.

Bid prices in dollars are drawn from a uniform distribution on [0, 0.06]. The maximum bid price is derived from the Amazon EC2 price of on-demand small instances in the US east region[7]. Considering the lower QoS for VMs in the spot market and the truthfulness properties of the mechanism, bidding higher

[7] http://aws.amazon.com/ec2/pricing/

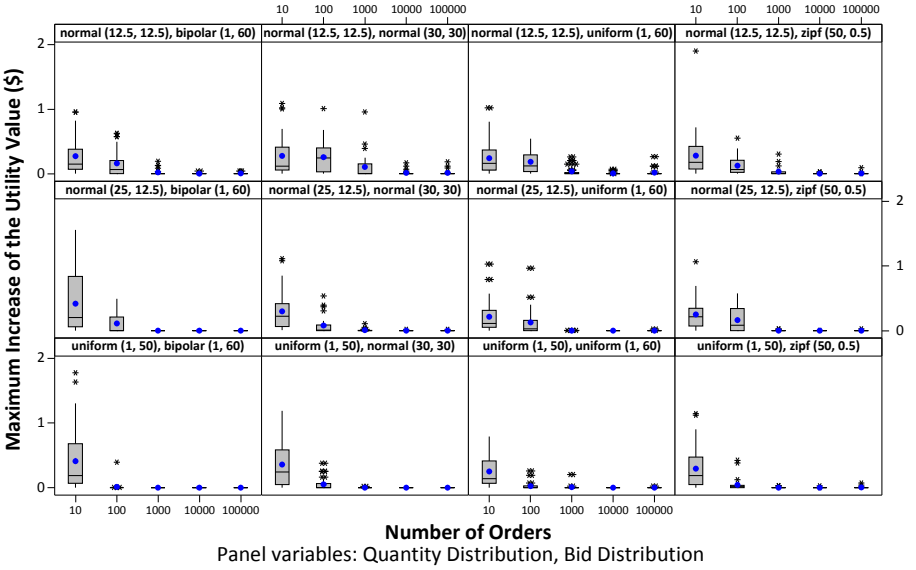


Figure 7: Maximum increase of the utility value among all bidders, achieved through misreporting quantity in Ex-CORE auction under different distribution of orders when $r = 50$. Mean value is denoted by blue disc.

than the on-demand price seems unreasonable. However, in real-world scenarios there might be orders with a bid higher than the on-demand price, as observed on the EC2 spot market. This is not of concern in our model.

The requested number of instances per order for each bidder is modeled by i.i.d. random variables uniformly distributed on [1, 50]. Amazon EC2 similarly imposes a limit of 100 VM instances per region that can be acquired by a customer in the spot market[8].

Following Mills et al. [30], the holding time of the VM instances by users is modeled by i.i.d Pareto distributed random variables, with shape parameter 1 and location parameter 1. Each generated random value represents the time in hours that VM instances remain in the system. If the order's bid price is lower than the current market price, the order remains in the queue for a maximum time period of half an hour. The order is considered in every auction round during this period. If the order is not serviced at the end of this period, it is labeled as rejected. The VMs that are instantiated following the acceptance of an order can be terminated at any time if the market price exceeds the order's bid. Upon termination, these VMs are not charged for their last partial hour. VMs that are terminated by their owner are charged for a discrete number of hours, with a partial hour of usage accounted for as a full hour.

To model the marginal cost of running VMs, we assume that the data center is populated with BL460c G6 blade servers that host a quad-core Intel Xeon E5504 2.0 GHz processor. The peak power usage per blade server is rated at 400

[8] `http://aws.amazon.com/ec2/spot-instances/`

W. Using the Amazon EC2 small instances type, each server is able to host up to 8 VMs. Two sets of electricity prices are considered for the data center, one for "on-peak" hours from 7am to 9pm and another for "off-peak" hours from 9pm to 7am. Following work by [31], we adopt a peak price of 0.108$/KWh and an off-peak price reduction of 50%. We compute *PUE* based on data center load and outside air temperature. Taking models by Goiri et al. [9] and Rasmussen [28] into account, Fig. 8 illustrates our modeling of the PUE as a function of outside temperature. Drastic jumps in *PUE* occur due to chiller activation when outside temperature crosses the $20°C$ mark [9]. We consider a relatively warm day with minimum temperature of $14°C$ and maximum temperature of $33°C$. We estimate hourly temperatures throughout the day based on a method by Gaylon et al. [32].

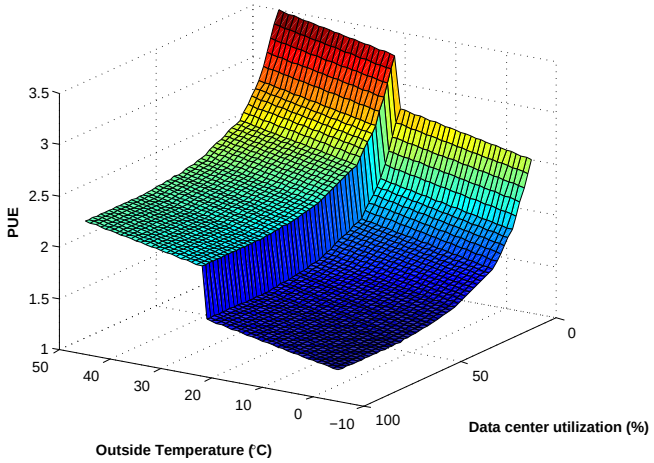
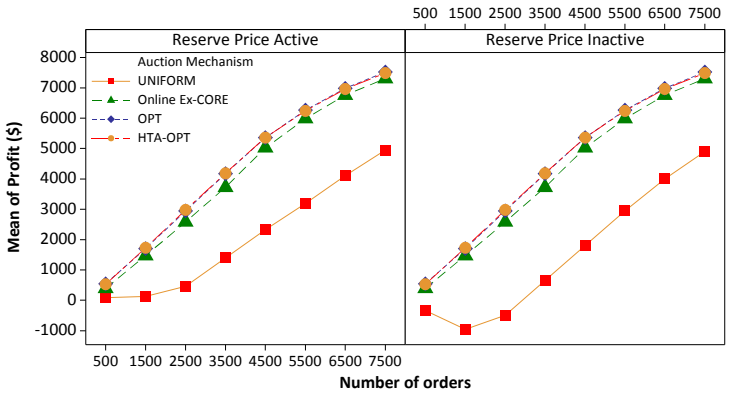


Figure 8: *PUE* as related to load and outside temperature.

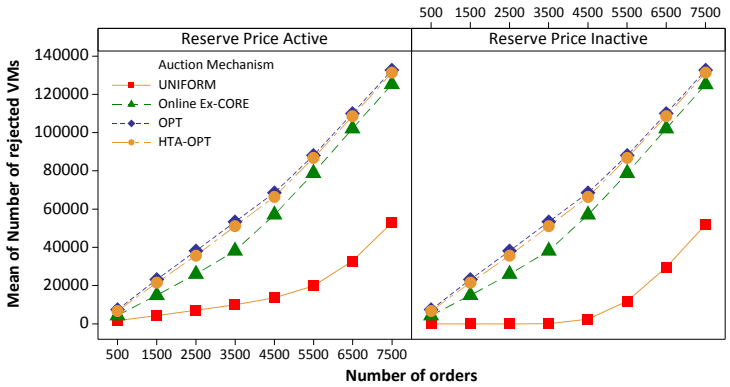
The green dashed line in Fig. 10 depicts the reserve price generated based on our model in a sample simulation run. Our investigation on the historical price data of spot instances for the past 90 days prior to 14th of November 2013 shows that the spot market price never goes below $0.007 for the small instances in the US-east region. The value complies with our computed reserve price for that instance type when the physical server characteristics, as well as the electricity prices and outside air temperature parameters are based on realistic data for an Amazon data center in the US-East region. The same holds true for the modeled and observed minimum spot price for the other instance types in the `m1` instance class, as they are based on hardware with similar power draw characteristics.

10.4.1 Experimental Results

We evaluate the online Ex-CORE auction by comparing profit and number of rejected VM requests to the other auction mechanisms outlined in Section 9.



(a)



(b)

Figure 9: (a) Average profit gained and (b) number of rejected VM instances with different auction mechanisms.

The computed profit is the total generated revenue minus the cost of electricity. The capital cost and all other fixed cost are not considered, as they are identical for all mechanisms. Each experiment is carried out 30 times and the mean value is reported. The results are illustrated in Fig. 9(a) and Fig. 9(b), where the number of orders in a 24-hour simulation is increased from 500 to 7500, and scenarios with or without the adoption of a reserve price are shown.

Fig. 9(a) shows that gained profit by all mechanisms increases with the number of orders. The OPT, HTA-OPT and online Ex-CORE auctions generate comparable profits, while there is a big gap between uniform price auction and the other mechanisms. When supply is higher than demand and there is no competition among bidders, all orders are accepted by the uniform price auction; and consequently the uniform price auction performs poorly under such circumstances. This supports the idea that the traditional auction mechanisms such as the Vickrey Auction [10] or Uniform price auction are not suitable for the cloud spot market in which supply is often higher than demand.

The benefit of using the online Ex-CORE auction is that, in spite of a small difference in generated profit compared to OPT and HTA-OPT (6% lower on average), it accommodates a considerably higher number of VMs (17% and 14% less rejections on average respectively). This reduces the impact of the bidder drop problem, introduced by Lee and Szymanski [1] that can be caused by frequent rejection of customers with low valuations.

As illustrated in Fig. 9(a) and Fig. 9(b), the reserve price only affects the outcome of the uniform price auction. Considering the range and distribution of bid values used in the simulation, the market price generated by online Ex-CORE, OPT and HTA-OPT is always higher than the reserve price. To exemplify further, Fig. 10 provides the reserve price and the market price generated by online Ex-CORE in a sample simulation run. This, however, does not mean that the reserve price is of no importance and can be ignored in real-world scenarios. In order to show the impact of the reserve price, different highest submitted bid prices are used to decrease the average market price as shown in Fig. 11. As can be seen in the figure, when the highest submitted bid price is low and therefore the market price is lower on average, the absence of reserve price can lead to loss or lower profit due to execution of VMs at a price below their variable cost.

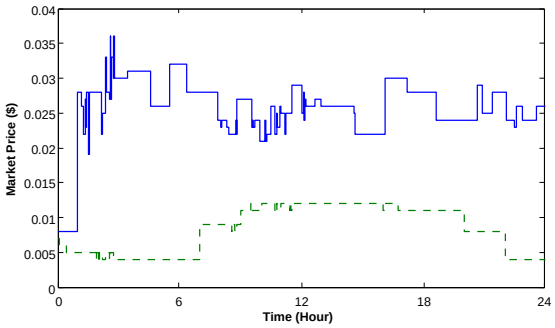


Figure 10: Reserve price (green dashed line) and spot market price generated by online Ex-CORE (blue solid line) in a sample simulation run when the number of orders is 1500.

As we are interested in the importance of a priori knowledge on the holding time of VMs by customers, the profit and the number of rejected VMs by OPT and HTA-OPT are investigated further. The results of a paired T-Test comparing the profit performance of OPT with HTA-OPT when the number of orders is 4500 and the holding time of VMs is distributed i.i.d. based on a Pareto distribution with both shape and location parameters equal to one are shown in Table 1. Given a null hypothesis of no statistically significant difference in mean profit by OPT and HTA-OPT, the p-value is relatively high (p-value = 0.846), suggesting that there is no strong evidence that the null hypothesis is false, i.e. there is no credible evidence that OPT and HTA-OPT on average generate different profit. However, there is a statistically significant difference in the mean number of rejected VMs. HTA-OPT rejected 2152 less VMs on average as it results in outcomes with a lower market price. Considering the

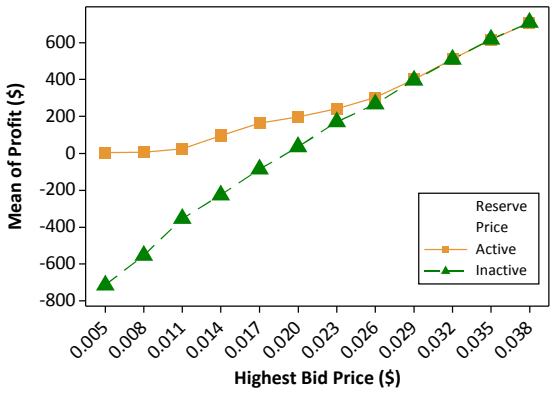


Figure 11: Average profit gained by Ex-CORE when the number of orders is 1500.

Table 1: Paired T-Test with 95% Confidence Interval (CI) for comparison of difference in mean of profit and number of rejected VMs generated by OPT and HTA-OPT (OPT − HTA-OPT) when the number of orders is 4500.

	OPT	**HTA-OPT**	**Difference (95% CI)**	**P-value**
Profit	5358.7	5361.6	-2.9 (-33.6, 27.7)	=0.846
Rejected	68575	66423	2152 (1192, 3111)	<0.001

reported 95% Confidence Interval (CI), we can state that knowing the holding time of VMs by itself does not change the amount of profit a provider generates as it is not aware of upcoming orders' bid prices.

11 Summary and Conclusion

With the rapid adoption of cloud computing environments, balancing supply and demand for cloud resources through dynamic forms of pricing is quickly gaining importance. In this chapter, we presented an envy-free auction that is truthful with high probability and that generates a near optimal profit for the cloud provider. The auction operates under conditions similar to the EC2 spot market. The truthfulness of the mechanism frees bidders from understanding its intricacies, thereby lowering the complexity of participation and the options for strategic behavior. At the same time, the mechanism aims to achieve a maximal profit for the provider, and achieves envy-freeness through the use of a uniform price. The mechanism is a generalization and extension of the consensus revenue estimate (CORE) auction that enables its application in the cloud computing setting, which requires an online recurrent auction with multi-unit requests. In order to incorporate marginal costs of production in the resource trading process, we pair the auctioning scheme with a method that calculates dynamic reserve prices based on a cost model that incorporates data center *PUE*, load, and electricity cost.

An important benefit of the proposed auction design is that it achieves near optimality w.r.t. maximizing revenue without requiring prior knowledge on the bid distributions. Our evaluation demonstrates its performance in this regard under a variety of order distributions. The proposed mechanism is shown to significantly outperform the uniform price auction and to closely approximate the profit outcome of the revenue maximizing, but non-truthful, optimal single price auction in an online setting (within 6% in our experiments), while improving on the number of rejected VMs (up to 17% in our experiments). Finally, our results show that the generated revenue does not differ significantly from the revenue attained by a mechanism based on dynamic programming that relies on prior knowledge regarding the holding time of VMs.

References

[1] J.-S. Lee and B. Szymanski, "A novel auction mechanism for selling time-sensitive e-services," in *Proceedings of Seventh IEEE International Conference on E-Commerce Technology, (CEC'05)*, Hong Kong, Jul. 2005, pp. 75–82.

[2] M. Macías and J. Guitart, "A genetic model for pricing in cloud computing markets," in *Proceedings of the 2011 ACM Symposium on Applied Computing (SAC'11)*, Taichung, Taiwan, Mar. 2011, pp. 113–118.

[3] M. Stokely, J. Winget, E. Keyes, C. Grimes, and B. Yolken, "Using a market economy to provision compute resources across planet-wide clusters," in *Proceedings of IEEE International Symposium on Parallel Distributed Processing (IPDPS'09)*, Rome, Italy, May 2009, pp. 1–8.

[4] W. Wang, B. Li, and B. Liang, "Towards optimal capacity segmentation with hybrid cloud pricing," in *Proceedings of the 32nd IEEE International Conference on Distributed Computing Systems (ICDCS'12)*, Macau, China, Jun. 2012, pp. 425–434.

[5] O. A. Ben-Yehuda, M. Ben-Yehuda, A. Schuster, and D. Tsafrir, "Deconstructing amazon ec2 spot instance pricing," *ACM Transaction Economy Computing*, vol. 1, no. 3, pp. 16:1–16:20, Sept. 2013.

[6] A. Danak and S. Mannor, "Resource allocation with supply adjustment in distributed computing systems," in *Proceedings of the 30th International Conference on Distributed Computing Systems (ICDCS'10)*, Genoa, Italy, Jun. 2010, pp. 498–506.

[7] Q. Zhang, Q. Zhu, and R. Boutaba, "Dynamic resource allocation for spot markets in cloud computing environments," in *Proceedings of the Fourth IEEE International Conference on Utility and Cloud Computing (UCC'11)*, Melbourne, Australia, Dec. 2011, pp. 178–185.

[8] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[9] Í. Goiri, K. Le, J. Guitart, J. Torres, and R. Bianchini, "Intelligent placement of datacenters for internet services," in *Proceedings of the 31st IEEE International Conference on Distributed Computing Systems (ICDCS'11)*, Minneapolis, Minnesota, USA, Jun. 2011, pp. 131–142.

[10] W. Vickrey, "Counterspeculation, auctions, and competitive sealed tenders," *The Journal of finance*, vol. 16, no. 1, pp. 8–37, 1961.

[11] A. V. Goldberg and J. D. Hartline, "Competitiveness via consensus," in *Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms (SODA'03)*, Baltimore, Maryland, USA, Jan. 2003, pp. 215–222.

[12] B. Javadi, R. K. Thulasiram, and R. Buyya, "Statistical modeling of spot instance prices in public cloud environments," in *Proceedings of the Fourth IEEE International Conference on Utility and Cloud Computing (UCC'11)*, Melbourne, Dec. 2011, pp. 219–228.

[13] S. Yi, D. Kondo, and A. Andrzejak, "Reducing costs of Spot instances via checkpointing in the Amazon Elastic Compute Cloud," in *In Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing (Cloud '10)*, Washington, USA, 2010, pp. 236–243.

[14] W. Voorsluys and R. Buyya, "Reliable provisioning of spot instances for compute-intensive applications," in *Proceedings of 26th International Conference on Advanced Information Networking and Applications (AINA'12)*, Fukuoka, Japan, Mar. 2012, pp. 542–549.

[15] Y. Song, M. Zafer, and K.-W. Lee, "Optimal bidding in spot instance market," in *Proceedings of the 31st International Conference on Computer Communications (INFOCOM'12)*, Orlando, Florida, USA, Mar. 2012, pp. 190–198.

[16] N. Chohan, C. Castillo, M. Spreitzer, M. Steinder, A. Tantawi, and C. Krintz, "See spot run: using spot instances for mapreduce workflows," in *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*. USENIX Association, 2010.

[17] M. Mihailescu and Y.-M. Teo, "The impact of user rationality in federated clouds," in *Proceedings of the 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid'12)*, Ottawa, Canada, May 2012, pp. 620–627.

[18] S. Zaman and D. Grosu, "Combinatorial auction-based allocation of virtual machine instances in clouds," *Journal of Parallel and Distributed Computing*, vol. 73, no. 4, pp. 495–508, 2013.

[19] R. B. Myerson, "Optimal auction design," *Mathematics of operations research*, vol. 6, no. 1, pp. 58–73, 1981.

[20] W. Wang, B. Liang, and B. Li, "Revenue maximization with dynamic auctions in IaaS cloud markets," in *Proceedings of the 21st IEEE/ACM International Symposium on Quality of Service (IWQoS'13)*, 2013, pp. 1–6.

[21] A. V. Goldberg, J. D. Hartline, A. R. Karlin, M. Saks, and A. Wright, "Competitive auctions," *Games and Economic Behavior*, vol. 55, no. 2, pp. 242 – 269, 2006.

[22] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, *Algorithmic game theory*. Cambridge University Press, 2007.

[23] A. V. Goldberg and J. D. Hartline, "Envy-free auctions for digital goods," in *Proceedings of the 4th ACM conference on Electronic Commerce (EC'03)*, San Diego, CA, USA, Jun. 2003, pp. 29–35.

[24] H. Moulin and S. Shenker, "Strategyproof sharing of submodular costs: budget balance versus efficiency," *Economic Theory*, vol. 18, no. 3, pp. 511–533, 2001.

[25] M. K. Patterson, "The effect of data center temperature on energy efficiency," in *Proceedings of 11th Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITHERM'08)*, Orlando, Florida, USA, May 2008, pp. 1167–1174.

[26] S. Pelley, D. Meisner, T. F. Wenisch, and J. W. VanGilder, "Understanding and abstracting total data center power," in *Workshop on Energy-Efficient Design (WEED'09)*, 2009.

[27] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, "The cost of a cloud: research problems in data center networks," *SIGCOMM Computing Communication Review*, vol. 39, no. 1, pp. 68–73, 2008.

[28] N. Rasmussen, "Electrical efficiency measurement for data centers," *White Paper by Schneider Electric - Data Center Science Center*, vol. 154 revision 2, 2011.

[29] S. Greenberg, E. Mills, B. Tschudi, P. Rumsey, and B. Myat, "Best practices for data centers: Lessons learned from benchmarking 22 data centers," *ACEEE Summer Study on Energy Efficiency in Buildings in Asilomar, CA.*, vol. 3, pp. 76–87, 2006.

[30] K. Mills, J. Filliben, and C. Dabrowski, "Comparing VM-placement algorithms for on-demand clouds," in *Proceedings of Third International Conference on Cloud Computing Technology and Science (CloudCom'12)*, Taipei, Taiwan, Dec. 2011, pp. 91–98.

[31] K. Le, R. Bianchini, J. Zhang, Y. Jaluria, J. Meng, and T. D. Nguyen, "Reducing electricity cost through virtual machine placement in high performance computing clouds," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC'11)*, Seattle, USA, Nov. 2011, pp. 22:1–22:12.

[32] G. S. Campbell and J. M. Norman, *Introduction to Environmental Biophysics*. Springer Verlag, 1998.